Krylov 子空间迭代法

杜磊
dulei@dlut.edu.cn

大连理工大学 数学科学学院
创新园大厦 B1207

2019 年 11 月 1 日

内容提要

1. 求解线性方程组的 Krylov 子空间迭代法
2. 求解特征值问题的 Krylov 子空间迭代法

参考文献 (线性方程组)

S. Yousef
Iterative Methods for Sparse Linear Systems
SIAM, 2003. (2nd edition)

H.A. van der Vorst
Iterative Krylov Methods for Large Linear Systems
Cambridge University, 2003.

V. Simoncini & D.B. Szyld
Recent computational developments in Krylov subspace methods for linear systems
Numer. Linear Algebra Appl., 14(2007): 1–59.

C. Vuik
Krylov Subspace Solvers and Preconditioners
ESAIM: Proceedings and Surveys, 63(2018), 1–43.

S. Yousef
Iterative methods for linear systems of equations: A brief historical journey
arXiv:1908.01083, 2019.

Krylov 子空间

定义

设 $A \in \mathbb{R}^{n\times n}, r \in \mathbb{R}^n$, 则由 A 和 r 生成的 m 阶 Krylov 子空间为

$$\mathcal{K}_m(A, r) = \mathrm{span}\{r, Ar, A^2 r, \cdots, A^{m-1} r\}.$$

构造 Krylov 子空间法的两个关键问题:

- 如何基底基底?
- 如何构造近似解?

Arnoldi 过程

Algorithm 1 Arnoldi 过程 (Gram-Schmidt)

```
设定 v_1 = r/‖r‖_2;
for j = 1, 2, ..., m do
    计算 h_ij = (Av_j, v_i), i = 1, 2, ..., j;
    计算 w_j = Av_j − Σ_{i=1}^{j} h_ij v_i;
    h_{j+1,j} = ‖w_j‖_2;
    如果 h_{j+1,j} = 0, 循环停止.
    v_{j+1} = w_j / h_{j+1,j};
end for
```

引理

如果程序 (1) 前 m 步没有中断, 则 $v_1, v_2, \ldots, v_m$ 为 Krylov 子空间 $\mathcal{K}_m(A, r) = \mathrm{span}\{r, Ar, A^2r, \cdots, A^{m-1}r\}$ 的一组标准正交基底.

引理

记 $V_m = [v_1, v_2, \ldots, v_m]$,

$$H_m = \begin{bmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1m} \\ h_{21} & h_{22} & h_{23} & \cdots & h_{2m} \\ & h_{32} & h_{33} & \cdots & h_{3m} \\ & & \ddots & \ddots & \vdots \\ & & & h_{m-1,m} & h_{mm} \end{bmatrix}, \tilde{H}_m = \begin{bmatrix} H_m \\ h_{m+1,m} e_m^{\mathrm{T}} \end{bmatrix},$$

则有下列关系成立:

$$\begin{aligned} AV_m &= V_m H_m + h_{m+1,m} v_{m+1} e_m^{\mathrm{T}} = V_{m+1} \tilde{H}_m, \\ V_m^{\mathrm{T}} A V_m &= H_m. \end{aligned}$$

Arnoldi 过程

Algorithm 2 Arnoldi 过程 (Modified Gram-Schmidt)

设定 $v_1 = r/\|r\|_2$;
for $j = 1, 2, \ldots, m$ **do**
  计算 $w_j = Av_j$;
  for $i = 1, 2, \ldots, j$ **do**
    $h_{ij} = (w_j, v_i)$;
    $w_j = w_j - h_{ij}v_i$;
  end for
  $h_{j+1,j} = \|w_j\|_2$;
  如果 $h_{j+1,j} = 0$, 循环停止.
  $v_{j+1} = w_j/h_{j+1,j}$;
end for

如果 A 为对称矩阵，则 H_m 为对称三对角矩阵

引理

记 $V_m = [v_1, v_2, \ldots, v_m]$,

$$H_m = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{m-1} \\ & & \beta_{m-1} & \alpha_m \end{bmatrix}, \tilde{H}_m = \begin{bmatrix} H_m \\ \beta_m \end{bmatrix},$$

则有下列关系成立:

$$\begin{aligned} A V_m &= V_m H_m + \beta_m v_{m+1} e_m^{\mathrm{T}} = V_{m+1} \tilde{H}_m, \\ V_m^{\mathrm{T}} A V_m &= H_m. \end{aligned}$$

Algorithm 3 Lanczos 过程

设定 $v_0 = 0, \beta_0 = 0, v_1 = r/\|r\|_2$;
for $j = 1, 2, \ldots, m-1$ **do**
  计算 $z = Av_j$;
  $\alpha_j = v_j^{\mathrm{T}} z$;
  $z = z - \alpha_j v_j - \beta_{j-1} v_{j-1}$;
  $\beta_j = \|z\|_2$;
  如果 $\beta_j = 0$, **循环停止**.
  $v_{j+1} = z/\beta_j$;
end for

Algorithm 4 FOM 法

设定 $v_1 = r / \|r\|_2$;
for $j = 1, 2, \ldots$ **do**
  计算 $w_j = Av_j$;
  for $i = 1, 2, \ldots, j$ **do**
    $h_{ij} = (w_j, v_i)$;
    $w_j = w_j - h_{ij} v_i$;
  end for
  $h_{j+1,j} = \|w_j\|_2$;
  如果 $h_{j+1,j} = 0$, 使 $m = j$ 并跳出循环.
  $v_{j+1} = w_j / h_{j+1,j}$;
end for
$y_m = H_m^{-1}(\beta e_1)$ 和 $x_m = x_0 + V_m y_m$.

FOM 法

推论

由 FOM 法得到的近似解 x_m 的残向量可表示为

$$b - Ax_m = -h_{m+1,m} e_m^{\mathrm{T}} y_m v_{m+1}$$

进而知

$$\|b - Ax_m\|_2 = h_{m+1,m} |e_m^{\mathrm{T}} y_m|.$$

- 基于 FOM 的变形
  - 重启 FOM (FOM(m))
  - IOM, DIOM

The Generalized Minimum Residual 法

Algorithm 5 GMRES 法

计算 $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$, $v_1 = r_0/\beta$;
for $j = 1, 2, \ldots$ **do**
  计算 $w_j = Av_j$;
  for $i = 1, 2, \ldots, j$ **do**
    $h_{ij} = (w_j, v_i)$;
    $w_j = w_j - h_{ij}v_i$;
  end for
  $h_{j+1,j} = \|w_j\|_2$;
  如果 $h_{j+1,j} = 0$, 使 $m = j$ 并跳出循环.
  $v_{j+1} = w_j/h_{j+1,j}$;
end for
定义 $(m+1) \times m$Hessenberg 矩阵 $\bar{H}_m = \{h_{ij}\}$;
计算 $y_m = \arg\min_{y \in \mathbb{R}^m} \left\|\beta e_1 - \bar{H}_m y\right\|_2$ 和 $x_m = x_0 + V_m y_m$.

共轭梯度法

当 A 对称正定时，Lanczos 过程可得 $T_k = V_k^{\mathrm{T}} A V_k$，其中

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

由 $r_k \perp \mathcal{K}_k$，可得

$$x_k = x_0 + V_k z_k = x_0 + V_k T_k^{-1}(\beta e_1).$$

根据 T_k 对称正定性，T_k 存在 LDL^{T} 分解，即有 $T_k = L_k D_k L_k^{\mathrm{T}}$. 于是

$$x_k = x_0 + V_k T_k^{-1}(\beta e_1) = x_0 + V_k T_k^{-1}(\beta e_1) = (V_k L_k^{-\mathrm{T}})(\beta D_k^{-1} L_k^{-1} e_1).$$

共轭梯度法

记

$$\tilde{P}_k \doteq V_k L_k^{-\mathrm{T}} = [\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_k],$$
$$y_k \doteq \beta D_k^{-1} L_k^{-1} e_1 = [\eta_1, \eta_2, \ldots, \eta_k]^{\mathrm{T}}.$$

假设 T_{k+1} 的 LDL^{T} 分解为 $T_{k+1} = L_{k+1} D_{k+1} L_{k+1}^{\mathrm{T}}$，可证下面的地推公式成立:

引理

$$\tilde{P}_k \doteq V_{k+1} L_{k+1}^{-\mathrm{T}} = [\tilde{P}_k, \tilde{p}_{k+1}],$$
$$y_{k+1} \doteq \beta D_{k+1}^{-1} L_{k+1}^{-1} e_1 = [y_k^{\mathrm{T}}, \eta_{k+1}]^{\mathrm{T}}.$$

图: MINRES 算法

x_0 is an initial guess, $r_0 = b - Ax_0$,
set $g = (\|r_0\|, 0, \ldots, 0)^{\mathrm{T}}$, $v_1 = r_0/\|r_0\|$,
for $n = 1, 2, \ldots$ do:
(Lanczos process)

$$\alpha_n = (v_n, Av_n),$$
$$\tilde{v}_{n+1} = Av_n - \alpha_n v_n - \beta_{n-1} v_{n-1},$$
$$\beta_n = (\tilde{v}_{n+1}, \tilde{v}_{n+1})^{1/2},$$
$$v_{n+1} = \tilde{v}_{n+1}/\beta_n,$$

set $t_{n-1,n} = \beta_{n-1}$, $t_{n,n} = \alpha_n$, $t_{n+1,n} = \beta_n$,
(Givens rotations)
for $i = \max\{1, n-2\}, \ldots, n-1$ do:

$$\begin{pmatrix} t_{i,n} \\ t_{i+1,n} \end{pmatrix} = \begin{pmatrix} c_i & s_i \\ -\bar{s}_i & c_i \end{pmatrix} \begin{pmatrix} t_{i,n} \\ t_{i+1,n} \end{pmatrix},$$

end

$$c_n = \frac{|t_{n,n}|}{\sqrt{|t_{n,n}|^2 + |t_{n+1,n}|^2}},$$
$$\bar{s}_n = \frac{t_{n+1,n}}{t_{n,n}} c_n,$$
$$t_{n,n} = c_n t_{n,n} + s_n t_{n+1,n},$$
$$t_{n+1,n} = 0,$$
$$\begin{pmatrix} g_n \\ g_{n+1} \end{pmatrix} = \begin{pmatrix} c_n & s_n \\ -\bar{s}_n & c_n \end{pmatrix} \begin{pmatrix} g_n \\ 0 \end{pmatrix},$$

(Update x_n)

$$p_n = (v_n - t_{n-2,n} p_{n-2} - t_{n-1,n} p_{n-1})/t_{n,n},$$
$$x_n = x_{n-1} + g_n p_n,$$

(Check convergence)
if $|g_{n+1}|/\|b\| \le \epsilon$, then stop.
end

双共轭梯度法 (Bi-CG)

图: Bi-CG 算法

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, $\beta_{-1} = 0$,

$\boldsymbol{r}_0^*$ is an arbitrary vector, such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$, *e.g.*, $\boldsymbol{r}_0^* = \boldsymbol{r}_0$,

for $n = 0, 1, \ldots,$ until $\|\boldsymbol{r}_n\| \leq \epsilon \|\boldsymbol{b}\|$ do:

$$\boldsymbol{p}_n = \boldsymbol{r}_n + \beta_{n-1}\boldsymbol{p}_{n-1}, \quad \boldsymbol{p}_n^* = \boldsymbol{r}_n^* + \bar{\beta}_{n-1}\boldsymbol{p}_{n-1}^*,$$

$$\alpha_n = \frac{(\boldsymbol{r}_n^*, \boldsymbol{r}_n)}{(\boldsymbol{p}_n^*, A\boldsymbol{p}_n)},$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \alpha_n \boldsymbol{p}_n,$$

$$\boldsymbol{r}_{n+1} = \boldsymbol{r}_n - \alpha_n A\boldsymbol{p}_n, \quad \boldsymbol{r}_{n+1}^* = \boldsymbol{r}_n^* - \bar{\alpha}_n A^{\mathrm{H}} \boldsymbol{p}_n^*,$$

$$\beta_n = \frac{(\boldsymbol{r}_{n+1}^*, \boldsymbol{r}_{n+1})}{(\boldsymbol{r}_n^*, \boldsymbol{r}_n)}.$$

end

平方共轭梯度法 (CGS)

图: CGS 算法

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, $\beta_{-1} = 0$,

$\boldsymbol{r}_0^*$ is an arbitrary vector, such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$, *e.g.*, $\boldsymbol{r}_0^* = \boldsymbol{r}_0$,

for $n = 0, 1, \ldots,$ until $\|\boldsymbol{r}_n\| \leq \epsilon \|\boldsymbol{b}\|$ do:

$$\boldsymbol{p}_n = \boldsymbol{r}_n + \beta_{n-1}\boldsymbol{z}_{n-1},$$

$$\boldsymbol{u}_n = \boldsymbol{p}_n + \beta_{n-1}(\boldsymbol{z}_{n-1} + \beta_{n-1}\boldsymbol{u}_{n-1}),$$

$$\alpha_n = \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)}{(\boldsymbol{r}_0^*, A\boldsymbol{u}_n)},$$

$$\boldsymbol{z}_n = \boldsymbol{p}_n - \alpha_n A\boldsymbol{u}_n,$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \alpha_n(\boldsymbol{p}_n + \boldsymbol{z}_n),$$

$$\boldsymbol{r}_{n+1} = \boldsymbol{r}_n - \alpha_n A(\boldsymbol{p}_n + \boldsymbol{z}_n),$$

$$\beta_n = \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_{n+1})}{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)}.$$

end

稳定双共轭梯度法 (Bi-CGSTAB)

图: Bi-CGSTAB 算法

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, $\beta_{-1} = 0$,

$\boldsymbol{r}_0^*$ is an arbitrary vector, such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$, *e.g.*, $\boldsymbol{r}_0^* = \boldsymbol{r}_0$,

for $n = 0, 1, \ldots,$ until $\|\boldsymbol{r}_n\| \leq \epsilon \|\boldsymbol{b}\|$ do:

$$\boldsymbol{p}_n = \boldsymbol{r}_n + \beta_{n-1}(\boldsymbol{p}_{n-1} - \zeta_{n-1} A\boldsymbol{p}_{n-1}),$$

$$\alpha_n = \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)}{(\boldsymbol{r}_0^*, A\boldsymbol{p}_n)},$$

$$\boldsymbol{t}_n = \boldsymbol{r}_n - \alpha_n A\boldsymbol{p}_n,$$

$$\zeta_n = \frac{(A\boldsymbol{t}_n, \boldsymbol{t}_n)}{(A\boldsymbol{t}_n, A\boldsymbol{t}_n)},$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \alpha_n \boldsymbol{p}_n + \zeta_n \boldsymbol{t}_n,$$

$$\boldsymbol{r}_{n+1} = \boldsymbol{t}_n - \zeta_n A\boldsymbol{t}_n,$$

$$\beta_n = \frac{\alpha_n}{\zeta_n} \cdot \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_{n+1})}{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)}.$$

end

内容提要