Advances in Applied Mathematics 应用数学进展, 2024, 13(10),4555-4562
Published Online October 2024 in Hans. https://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2024.1310436

Hans汉斯

广义极小残差法中基于Arnoldi过程的多项式预处理方法

耿　硕

天津职业技术师范大学理学院，天津

收稿日期：2024年9月14日；录用日期：2024年10月7日；发布日期：2024年10月15日

摘　要

本文探讨了在求解大规模稀疏线性方程组时，多项式预处理技术在*GMRES*方法中的应用，提高了其计算效率和计算精度。我们分析了多项式预处理如何增加用于形成近似解的多项式的阶数。同时为了简化多项式预处理的过程，我们提出了基于*Arnoldi*过程的多项式预处理方法，通过直接利用*Arnoldi*基向量和递归系数来构造多项式 $p(A)b$，从而有效避免了对多项式系数的直接计算。通过数值算例验证了这种方法简单且高效，为多项式预处理在*GMRES*中的应用提供了新的视角。

关键词

***GMRES*算法，线性方程组，预处理，稀疏矩阵**

A Polynomial Preprocessing Method Based on the Arnoldi Process in the Generalized Minimal Residual Method

Shuo Geng

School of Science, Tianjin University of Technology and Education, Tianjin

Received: Sep. 14th, 2024; accepted: Oct. 7th, 2024; published: Oct. 15th, 2024

Abstract

In this paper, the application of polynomial preprocessing technology in the *GMRES* method is discussed when solving large-scale sparse linear equations, which improves its computational efficiency and computational accuracy. We analyze how polynomial preprocessing increases the order

文章引用: 耿硕. 广义极小残差法中基于 Arnoldi 过程的多项式预处理方法[J]. 应用数学进展, 2024, 13(10): 4555-4562.
DOI: 10.12677/aam.2024.1310436

of the polynomial used to form an approximate solution. At the same time, in order to simplify the process of polynomial preprocessing, we propose a polynomial preprocessing method based on the *Arnoldi* process, which directly uses the *Arnoldi* basis vector and recursive coefficients to construct the polynomial $p(A)b$, which effectively avoids the direct calculation of the polynomial coefficients. Numerical examples verify that this method is simple and efficient, which provides a new perspective for the application of polynomial preprocessing technology in the *GMRES* method.

Keywords

***GMRES* Algorithm, Systems of Linear Equations, Preprocessing, Sparse Matrices**

1. 引言

大规模稀疏线性方程组 $Ax=b$ 的求解是科学与工程计算中的核心问题之一。随着问题规模的增大和复杂性的提高，传统的直接求解方法往往面临计算量巨大和存储需求高的挑战。而迭代方法很好地解决了这一问题，迭代算法通过逐步逼近解的方式，对于大规模问题通常更加高效。它们不需要一次性计算整个矩阵的逆或分解，因此可以显著减少计算量和存储需求[1]。*Krylov* 子空间方法也是一种迭代算法，其特别适用于处理大规模稀疏矩阵问题。在求解过程中，它只需要存储和计算矩阵向量乘积，而不需要显式地存储整个矩阵，大大降低了存储需求并提高了计算效率。相比之下，一些传统的迭代算法可能需要更多的存储空间和计算资源来处理稀疏矩阵。同时，*Krylov* 子空间方法通过构建一个低维度的 *Krylov* 子空间来逼近原问题的解，实现了对原问题的降维处理。这种降维处理不仅降低了问题的复杂度，还使得算法更加灵活和高效[2]。*GMRES* 是一种 *Krylov* 子空间方法，因其存储需求低和计算效率高的特点而备受关注[3]。然而，对于矩阵 A 的谱特性复杂的问题，*GMRES* 的性能可能会受到限制[4]。

为了进一步提高 *GMRES* 的性能，多项式预处理成为了一个重要的研究方向[5]。多项式预处理通过在求解过程中引入一个多项式函数 $p(A)$，将原始线性方程组转化为 $p(A)Ax=p(A)b$。多项式预处理技术通过对方程组的系数矩阵进行预处理，降低其条件数，从而加速迭代过程的收敛速度。然而，构造合适的多项式 $p(A)$ 并非易事。切比雪夫多项式和最小二乘多项式是两种常见的方法[6]，但它们都依赖于对矩阵 A 的谱特性的准确估计，而这在许多实际问题中是做不到的。由此推测，求解多项式的复杂也可能阻碍了多项式预处理在 *GMRES* 中的广泛应用。因此，本文提出了一种基于 *Arnoldi* 过程的多项式预处理方法。该方法利用 *Arnoldi* 算法生成的基向量和递归系数，直接实现多项式 $p(A)b$，而无需显式计算多项式 p 的系数。这种方法不仅简化了多项式预处理的实施，还避免了系数计算精度不高所带来的问题。数值算例的结果也说明了该方法的有效性。

2. 广义极小残差法(*GMRES*)算法

考虑线性方程组

$$Ax=b,$$

其中矩阵 $A\in R^{n\times n}, b\in R^n$ 是已经给定的，而向量 $x\in R^n$ 是待求的未知向量。这里假定系数矩阵 A 是非奇异的大型稀疏矩阵，而且 $A\neq A^{\mathrm{T}}$。广义极小残差法是求 $x_k\in K_k(A,b)$，$K_k(A,b)=span\{b,Ab,A^2b,\cdots,A^{k-1}b\}$

使得

$$\|r_k\|_2 = \min\left\{\|b-Ax\|_2 : x_k \in K_k(A,b)\right\},$$

其中 $r_k = b - Ax_k$，即求 $x_k \in K_k(A,b)$，使得残差向量 r_k 的 2-范数最小。

由 *Arnoldi* 分解[7]可以得到

$$AQ_k = Q_k H_k + \beta_k q_{k+1} e_k^{\mathrm{T}} = Q_{k+1}\hat{H}_k,$$

其中 $Q_{k+1} = \left[Q_k, q_{k+1}\right] \in R^{n\times(k+1)}$ 满足 $Q_{k+1}^{\mathrm{T}} Q_{k+1} = I_{k+1}$，而矩阵

$$\hat{H}_k = \begin{bmatrix} H_k \\ \beta_k e_k^{\mathrm{T}} \end{bmatrix} \in R^{(k+1)\times k}$$

是上 *Hessenberg* 矩阵。对任意的 $x = Q_k y \in K_k(A,b)$，有

$$\begin{aligned} \|b-Ax\|_2 &= \|b - AQ_k y\|_2 \\ &= \left\|\beta_0 Q_{k+1} e_1 - Q_{k+1}\hat{H}_k y\right\|_2, \\ &= \left\|\beta_0 e_1 - \hat{H}_k y\right\|_2 \end{aligned}$$

其中 $\beta_0 = \|b\|_2$，则极小化问题 $\|r_k\|_2 = \min\left\{\|b-Ax\|_2 : x_k \in K_k(A,b)\right\}$ 等价于求 $y_k \in R^k$，使得

$$\left\|\beta_0 e_1 - \hat{H}_k y_k\right\|_2 = \min\left\{\left\|\beta_0 e_1 - \hat{H}_k y\right\|_2 : y \in R^k\right\}.$$

之后再利用 $\hat{H}_k$ 的 *QR* 分解来求解上述最小二乘问题。由于 $\hat{H}_k$ 是上 *Hessenberg* 矩阵，可以计算 k 个 *Givens* 旋转变换

$$G(i,j,\theta) = \begin{pmatrix} 1 & & & & & & & & \\ & 1 & & & & & & & \\ & & \ddots & & & & & & \\ & & & \cos\theta & & \sin\theta & & & i \\ & & & & 1 & & & & \\ & & & & & \ddots & & & \\ & & & & & & 1 & & \\ & & & -\sin\theta & & \cos\theta & & & j \\ & & & & & & & 1 & \\ & & & & & & & & \ddots \\ & & & & & & & & & 1 \\ & & & i & & j & & & \end{pmatrix}$$

使得

$$\left(G_k G_{k-1}\cdots G_2 G_1\right)\hat{H}_k = \begin{bmatrix} R_k \\ 0 \end{bmatrix},$$

其中 R_k 是非奇异的上三角矩阵。由此可得最小二乘问题 $\left\|\beta_0 e_1 - \hat{H}_k y_k\right\|_2 = \min\left\{\left\|\beta_0 e_1 - \hat{H}_k y\right\|_2 : y \in R^k\right\}$ 的解为

$$y_k = R_k^{-1} t_k,$$

其中 $t_k = \left(\tau_1, \tau_2, \cdots, \tau_k\right)^{\mathrm{T}}$，$\tau_1 = \beta_0 c_1$，$\tau_j = (-1)^{j-1}\beta_0 s_1 s_2 \cdots s_{j-1} c_j$，$j = 2,3,\cdots,k$，此时的残差向量的范数为

$$\left\|b-AQ_k y_k\right\|_2=\left\|\beta_0 e_1-\hat{H}_k y_k\right\|_2=\left|\beta_k\right|,$$

其中 $\beta_k=(-1)^k\beta_0 s_1 s_2\cdots s_k$。

在实际使用这一算法时，一般先选定一个不太大的正整数 m，用 *GMRES* 方法计算出 x_m，然后再以 x_m 为初始向量重新运行。这就是所谓的 *GMRES* (m)算法，具体算法如下。

***Algorithm*1**：重新启动的 *GMRES* 算法(*GMRES* (m))
Input：稀疏矩阵 A，右端项 b，单次循环迭代次数 m
Output：迭代近似解 x

Step1：给定迭代初始值以及终止条件

$x_m=0$,

$r_0=b-Ax_m$,

$\beta_0=\|r_0\|_2$,

$q_1=r_0/\beta_0$,

$\rho_m=\|b\|_2$,

$\dfrac{|\rho_m|}{\|b\|_2}<\varepsilon$.

Step2：得到迭代解 x_m

先由 *Arnoldi* 算法可以得到一个长度为 m 的 *Arnoldi* 分解

$$AQ_m=Q_{m+1}\hat{H}_m$$

再利用 *Givens* 旋转变换计算 $\hat{H}_k$ 的 *QR* 分解

$$\hat{H}_k=G^{\mathrm{T}}\begin{bmatrix}R_m\\0\end{bmatrix}$$

最后通过回代计算 x_m

$$x_m=x_m+Q_m y_m$$

3. 多项式预处理

我们在用 *GMRES* (m)子空间方法来解决线性方程组 $Ax=b$ 时，由于 A 的条件数较大收敛速度较慢，可以考虑使用多项式预处理方法降低其条件数，从而加速迭代过程的收敛速度。进而方程将会转化为

$$p(A)Ax=p(A)b \tag{1}$$

其中 p 是一个多项式用 *GMRES* (m)方法求解这个新的方程组时就会有着更快的收敛速度，此方法也可以与其他预处理方法结合使用。

设 p 是 d 次多项式，记 $s(A)\equiv p(A)A$，则 s 的次数为 $d+1$。设 *GMRES* (m)某一步迭代中的近似解为 $\hat{x}=\pi(A)r_0$，其中 π 是最高次为 $m-1$ 次的多项式。则有

$$\hat{x}=\pi(s(A))\,p(A)r_0 \tag{2}$$

因此，实际用于形成近似解的组合多项式 $(\pi\circ s)\,p$ 的次数为 $(m-1)*(d+1)+d=m*d+m-1$。用于求

近似解的最小化是在维度 m 的子空间上，而不是维度为 $m*(d+1)$ 的整个 *Krylov* 子空间上。

4. 基于 *Arnoldi* 过程的多项式预处理 *GMRES* 算法

接下来我们给出一种基于 *Arnoldi* 过程的多项式预处理方法，首先计算 *Arnoldi* 基向量，并找到将这些基向量组合在一起以产生最小残差解的系数。这些系数来自标准 *GMRES* 方法中的中间步骤。然后，为了实现 $p(A)b$，对于某个向量 b，必须应用相同的 *Arnoldi* 递归。就好像我们用起始向量 b 运行 *Arnoldi*，但我们没有计算用来向量正交化的标量，而是使用与找到多项式信息的原始 *Arnoldi* 迭代中使用的相同的递归系数。将其与生成的 *Arnoldi* 多项式组合表示多项式 p，最后应用于向量 b，得到 $p(A)b$。

Algorithm2：使用 *Arnoldi* 基向量实现 $p(A)b$

Input：稀疏矩阵 A，右端项 b，预处理多项式的最高次项的次数 d

Output：迭代近似解 y

Step1：p 的初始形式

$$v_1 = b/\|b\|_2,$$

$$AV_{d+1} = V_{d+2}H_{d+2,d+1}.$$

$$\min\|e_1 - H_{d+2,d+1}g\|_2.$$

Step2：对于任意向量 b，实现 $y = p(A)b$.

$$y = g_1 * b,$$

$$w_1 = b.$$

for $j = 1:d$

$$t = A * w_j,$$

$$t = t - \sum_{i=1}^{j} h_{ij} * w_i,$$

$$w_{j+1} = t/h_{j+1,j},$$

$$y = y + g_{j+1} * w_{j+1}.$$

End

可以考虑对应于 *Arnoldi* 迭代生成的 *Arnoldi* 向量的多项式。例如，第二个 *Arnoldi* 向量(V_{d+1} 的第二列)为 $v_2 = (1/h_{21})(Av_1 - h_{11}v_1)$，$A$ 的相应多项式为 $\tau(A) = (1/h_{21})(A - h_{11}I)$。当这些 *Arnoldi* 多项式应用于起始向量 v_1 时，它们会产生正交向量。在上面的算法中，我们在步骤 2 中将这些相同的多项式应用于另一个向量 b。但是，生成的 w_j 向量不是正交的。

计算 $p(A)b$ 的矩阵向量积的数量为 d (与之前相同)。向量运算的次数约为 $d^2/2 + 2.5d$。这里额外的计算有时将很重要。

5. 数值算例

为了展示出基于 *Arnoldi* 过程的多项式预处理方法的有效性，我们一共给出了四个例子，分别是当稀疏矩阵 A 为较低条件数、较高条件数、正定矩阵和不定矩阵四种不同性质的矩阵时，该算法均展现出了良好的效果。

例 1：对于线性方程组 $Ax=b$。矩阵 A 的大小为 50×50，对角线元素为 $\{1,2,\cdots,50\}$，次对角线元素为 $\{0.2,0.2,\cdots,0.2\}$，向量 $b=[1,1,\cdots,1]_{1\times 50}^{\mathrm{T}}$。

我们分别用 *GMRES* (10)方法和基于 *Arnoldi* 过程的多项式预处理方法(以下简称 *Arnoldi* 多项式法)计算方程组的解。结果显示如下：

方法	*GMRES* (10)	*Arnoldi* 多项式法
误差范数	8.8×10^{-7}	6.2177×10^{-15}
迭代时间/秒	0.2807	0.0068

从本例的结果可以看出，在面对较低条件数的稀疏矩阵 $A\left(cond\left(A\right)=50.3431\right)$ 时，*Arnoldi* 多项式法表现出了出色的收敛效果，无论是在迭代速度上还是在迭代精度上都要优于 *GMRES* (10)算法。同时我们再来看下一个例子。

例 2：对于线性方程组 $Ax=b$。矩阵 A 的大小为 500×500，对角线元素为 $\{0.1,0.2,\cdots,1,2,\cdots,491\}$，次对角线元素为 $\{0.2,0.2,\cdots,0.2\}$ 组成的，向量 $b=[1,1,\cdots,1]_{1\times 500}^{\mathrm{T}}$。

本例中稀疏矩阵 A 的条件数为 $cond\left(A\right)=8.3218\times 10^{3}$，我们依旧对两种算法进行对比，结果如下：

方法	*GMRES* (10)	*Arnoldi* 多项式法
误差范数(迭代 100 次)	0.0230	0.4787
迭代时间/秒	0.0705	0.0316
误差范数(迭代 200 次)	0.0025	2.5324×10^{-4}
迭代时间/秒	0.1144	0.0923

通过对比可以看出，在此例中，*Arnoldi* 多项式法在迭代初期的收敛速度相对于 *GMRES* (10)来说较慢，迭代精度较低，然而随着迭代的继续，在迭代后期 *Arnoldi* 多项式法的收敛速度要快于 *GMRES* (10)，迭代精度也有所提升。并且从迭代所耗的时间上来看 *Arnoldi* 多项式法也有着不错的效果。

例 3：对于线性方程组 $Ax=b$。矩阵 A 的大小为 2000×2000，对角线元素为 $\{0.1,0.2,\cdots,1,2,\cdots,1981\}$ 次对角线元素为 $\{0.2,0.2,\cdots,0.2\}$，向量 $b=[1,1,\cdots,1]_{1\times 2000}^{\mathrm{T}}$。

用一个更大规模的稀疏矩阵来比较两种算法，结果如下：

方法	*GMRES* (10)	*Arnoldi* 多项式法
误差范数(迭代 100 次)	0.0390	2.3049
迭代时间/秒	0.2721	0.4961
误差范数(迭代 200 次)	0.0120	0.4993
迭代时间/秒	0.5193	1.0246
误差范数(迭代 400 次)	0.0013	2.5907×10^{-4}
迭代时间/秒	1.0702	2.6187

在此例中，稀疏矩阵 A 正定且条件数为 $cond\left(A\right)=3.3744\times 10^{4}$，*Arnoldi* 多项式法虽然在迭代初期表现的不够出色，但随着迭代的进行，在迭代后期，在迭代精度上有着不错的迭代效果。仅仅牺牲了一点

迭代时间，而在迭代精度上却提升了一位计算精度。

例 4：对于线性方程组 $Ax=b$。矩阵 $A=SDS^{-1}$，$A,S,D\in R^{1000\times1000}$，$S=(1,0.9)$ 是双对角阵，1 是主对角元，0.9 是其上对角元，D 是对角阵，对角元为 $\{-10,-9,\cdots,-1,1,\cdots,990\}$，向量 $b=[1,1,\cdots,1]^{\mathrm{T}}_{1\times1000}$。

方法	*GMRES* (10)	*Arnoldi* 多项式法
误差范数(迭代 100 次)	0.0740	0.7254
迭代时间/秒	0.5422	0.1105
误差范数(迭代 200 次)	0.0480	0.2037
迭代时间/秒	0.6253	0.3020
误差范数(迭代 300 次)	0.0290	7.2663×10^{-6}
迭代时间/秒	0.8808	0.4780

由此例结果可以看出，由于矩阵 A 有负特征值的影响，*GMRES* (10)方法并没有产生良好的迭代效果，而 *Arnoldi* 多项式法却表现出了良好的迭代效果。

除上面四个例子展示的结果之外，我们还对算法的收敛稳定性进行了对比，由于在各个例子中的收敛稳定性均大致相同，所以只展示例 2 的结果图，如图 1 所示。

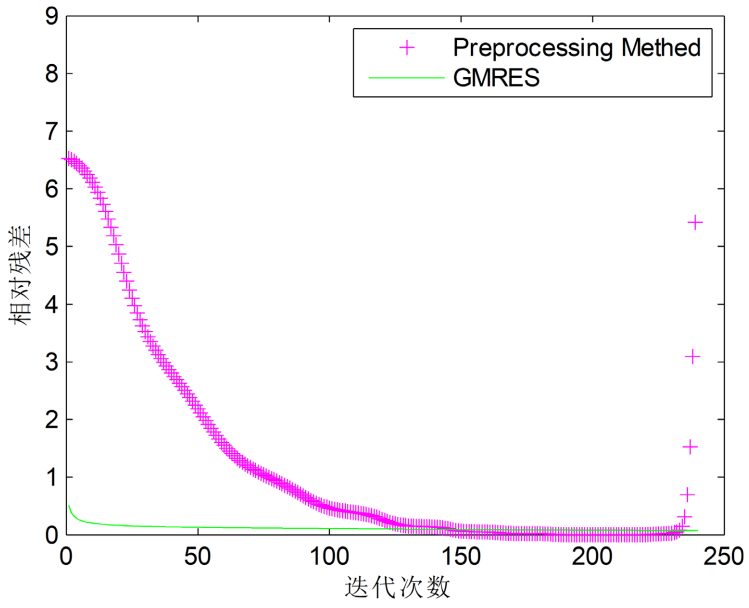

Figure 1. Comparison of the residuals of the *Arnoldi* polynomial method and the *GMRES* algorithm
图 1. *Arnoldi* 多项式法与 *GMRES* 算法的残差对比图

如上图所示，基于 *Arnoldi* 过程的多项式预处理方法虽然会随着迭代的进行，迭代精度逐渐超过 *GMRES* 算法，但是当迭代一直进行下去时，会出现收敛不稳定的现象。

6. 结论

我们提出了一种基于 *Arnoldi* 过程的多项式预处理技术，并将其运用于 *GMRES* 算法中，通过计算一些数值算例来与重启的 *GMRES* 算法进行对比，结果显示出该预处理方法在面对不同问题时，在计算精

度以及迭代时间上都有着不错的表现结果，验证了该方法的有效性及可行性。未来还可以对该 *Arnoldi* 多项式预处理方法进行进一步完善，就比如，使该方法能够具有更高的收敛精度，使其能够在面对更为复杂的问题时依然具有良好的收敛稳定性。

参考文献

[1] Saad, Y. (2003) Iterative Methods for Sparse Linear Systems. Society for Industrial and Applied Mathematics. https://epubs.siam.org/doi/book/10.1137/1.9780898718003

[2] Saad, Y. (1981) Krylov Subspace Methods for Solving Large Unsymmetric Linear Systems. *Mathematics of Computation*, **37**, 105-126. https://doi.org/10.1090/s0025-5718-1981-0616364-6

[3] Saad, Y. and Schultz, M.H. (1986) GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems. *SIAM Journal on Scientific and Statistical Computing*, **7**, 856-869. https://doi.org/10.1137/0907058

[4] Ghai, A., Lu, C. and Jiao, X. (2018) A Comparison of Preconditioned Krylov Subspace Methods for Large-Scale Nonsymmetric Linear Systems. *Numerical Linear Algebra with Applications*, **26**, e2215. https://doi.org/10.1002/nla.2215

[5] Joubert, W. (1994) A Robust Gmres-Based Adaptive Polynomial Preconditioning Algorithm for Nonsymmetric Linear Systems. *SIAM Journal on Scientific Computing*, **15**, 427-439. https://doi.org/10.1137/0915029

[6] Ashby, S.F., Manteuffel, T.A. and Otto, J.S. (1992) A Comparison of Adaptive Chebyshev and Least Squares Polynomial Preconditioning for Hermitian Positive Definite Linear Systems. *SIAM Journal on Scientific and Statistical Computing*, **13**, 1-29. https://doi.org/10.1137/0913001

[7] Arnoldi, W.E. (1951) The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem. *Quarterly of Applied Mathematics*, **9**, 17-29. https://doi.org/10.1090/qam/42792