# Note on Krylov subspace methods*

Zhengbo Zhou†

May 31, 2025

[1] Higham, N. J. (2023). Numerical linear algebra and matrix analysis. Unpublished manuscript.

## 1 Introduction

The $k$th Krylov subspace of $A \in \mathbb{C}^{n\times n}$ and a nonzero vector $b \in \mathbb{C}^n$ is defined by

$$\mathcal{K}_k(A, b) = \text{span}\{b, Ab, \ldots, A^{k-1}b\}.$$

In the Krylov subspace method for solving a linear system $Ax = b$, each iterate $x_k$ is chosen from the shifted Krylov subspace $x_0 + \mathcal{K}_k(A, r_0)$, where $r_0 = b - Ax_0$, that is,

$$x_k - x_0 \in \mathcal{K}_k(A, r_0). \tag{1.1}$$

We can rewrite (1.1) as

$$x_k - x_0 = \sum_{i=0}^{k-1} \alpha_i A^i r_0 =: q_{k-1}(A)r_0,$$

where $q_{k-1}$ is a polynomial of degree at most $k-1$. Multiply by $A$ gives,

$$Ax_k - Ax_0 = Aq_{k-1}(A)r_0,$$

using the definition $r_k := b - Ax_k$, we have

$$r_k - r_0 = Aq_{k-1}(A)r_0$$

or

$$r_k = q_k(A)r_0,$$

where $p_k$ is a polynomial of degree at most $k$ with $q_k(0) = 1$. The minimal residual method (MINRES) for symmetric matrix, and the generalized minimal residual method (GMRES) for general $A$. In both cases, $x_k \in x_0 + \mathcal{K}_k(A, r_0)$ to minimize $\|r_k\|_2^2$.

*Date: May 31, 2025, Manchester

†Department of Mathematics, University of Manchester, Manchester M13 9PL, United Kingdom. zhengbo.zhou@student.manchester.ac.uk

## 2 The Arnoldi Method

The Arnoldi process for $A \in \mathbb{C}^{n\times n}$ attempts to compute the Hessenberg reduction $Q^*AQ = H$, where $Q \in \mathbb{C}^{n\times n}$ is unitary and $H \in \mathbb{C}^{n\times n}$ is upper Hessenberg. Write $Q = [q_1, \ldots, q_n]$ and equating $k$th columns in $AQ = QH$ gives

$$Aq_k = \sum_{i=1}^{k+1} h_{ik} q_i, \quad k = 1 : n-1, \tag{2.1}$$

and for the $n$th column we have

$$Aq_n = \sum_{i=1}^{n} h_{in} q_i.$$

(2.1) may also be rewritten as

$$h_{k+1,k} q_{k+1} = Aq_k - \sum_{i=1}^{k} h_{ik} q_i, \qquad k = 1 : n-1. \tag{2.2}$$

Using the fact that the columns of $Q$ are orthonormal. By choosing an arbitrary $j = 1 : k$, we have

$$q_j^* q_{k+1} h_{k+1,k} = q_j^* Aq_k - \sum_{i=1}^{k} h_{ik} q_j^* q_i.$$

Notice that the latter term is only nonzero if $i$ is taken to be the same value as $j$. Therefore, by choosing $i = j$, we have

$$0 = q_j^* Aq_k - h_{jk} \Rightarrow h_{jk} = q_j^* Aq_k, \quad j = 1 : k.$$

Let us define $r_k := Aq_k - \sum_{i=1}^{k} h_{ik} q_i$, the right hand side of (2.2). Then we can write it as $h_{k+1,k} q_{k+1} = r_k$. If $r_k \neq 0$, then $q_{k+1} = r_k / h_{k+1,k}$, with $h_{k+1,k} = \|r_k\|_2$. If $r_k = 0$, the process will terminate within. Notice that, we have already obtain the algorithm implicitly. Here, $r_k$ can be computed prior to $q_k$, and the algorithm can proceed.

To illustrate the difference between the combined version and separate version as discussed in Algorithm 1, we provide a small numerical experiment. Figure 1 shows the backward error and the orthogonality error. We observe that using the combined version, both error are significantly smaller than the separate one.

From (2.2), it follows by induction that

$$\text{span}\{q_1, \ldots, q_k\} = \text{span}\{q_1, Aq_1, \ldots, Aq_{k-1}\},$$

that is, the Arnoldi vectors $\{q_i\}_{i=1}^k$ form an orthonormal basis for the Krylov subspace $\mathcal{K}_k(A, q_1)$.

The eigenvalues of the Hessenberg matrix $H_m$ produced by the Arnoldi process are called **Ritz values** and the corresponding eigenvectors are **Ritz vectors**. If $(\lambda, v)$ is an eigenpair of $H_k$, then $(\lambda, Q_k v)$ is an approximate eigenpair of $A$.

After $k$ stages, the Arnoldi process produces the factorization

$$AQ_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T, \tag{2.3}$$

**Algorithm 1.** *Arnoldi process with modified Gram-Schmidt orthogonalization.*

**Input:** Given a square matrix $A \in \mathbb{C}^{n\times n}$, a starting vector $q_1$ with unit 2 norm.

**Output:** A column-wise orthonormal matrix $Q \in \mathbb{C}^{n\times m}$ and a upper Hessenberg matrix $H \in \mathbb{C}^{m\times m}$, such that $A = QHQ^*$.

1: **for** $k = 1 : n$ **do** % For $k = n$, the only part that will be executed is the generation of the $n$th column of $H$. Afterwards, the 2-norm of "$h_{n+1,n}$" will definitely becomes zero since there are only $n$ basis for a $n$ dimensional vector space.

2: $w_k = Aq_k$

3: **for** $j = 1 : k$ **do**

4: $h_{jk} = q_j^* w_k$

5: $w_k = w_k - h_{jk} q_j$

6: **end for** % When we started from $w_k = Aq_k$, we generated the "temporary" $(k+1)$th basis. In order to make the newly generated basis orthogonal to $q_1, \ldots, q_k$, we need to delete the corresponding components, as described in (2.1) and (2.2). However, the question is *the definition of $h_{jk}$ is $q_j^* Aq_k$, but how can we keep "modify" $Aq_k$, $w_k$?* The answer can be seen from (2.1). After the first iteration, we get $w_k = \sum_{i=2}^{k+1} h_{ik} q_i$. In the second iteration, when pre-multiply $q_2^*$ to $w_k$, we will not getting any contribution from $h_{1k}q_1$, even though in the scenario we do not remove it, since $q_1$ and $q_2$ are constructed to be orthonormal. Therefore

$$q_2^* \sum_{i=1}^{k+1} h_{ik} q_i = q_2^* \sum_{i=2}^{k+1} h_{ik} q_i.$$

Therefore, removing the computed component is mathematically equivalent to the unremoved version. The next question is *why we doing in this removing approach.* Firstly, one should notice that if we do these two steps in the loop separately like:

**for** $j = 1 : k$ **do** $h_{jk} = q_j^* w_k$ **end for**

**for** $j = 1 : k$ **do** $w_k = w_k - h_{jk} q_j$ **end for**

This is mathematically equivalent to the combined one. This approach is purely for numerical consideration. Since in floating point arithmetic, the inner product of two orthonormal vectors may not be zero but a small number about $u$, the unit roundoff. Therefore, without removing the previous parts, it is possible that, at the $k$th step, all the vectors $q_1, \ldots, q_{k-1}$ will contaminate the entries of $H$, which leads to an inaccurate decomposition.

7: **if** $\|w_k\|_2 = 0$ **then**

8: **quit**

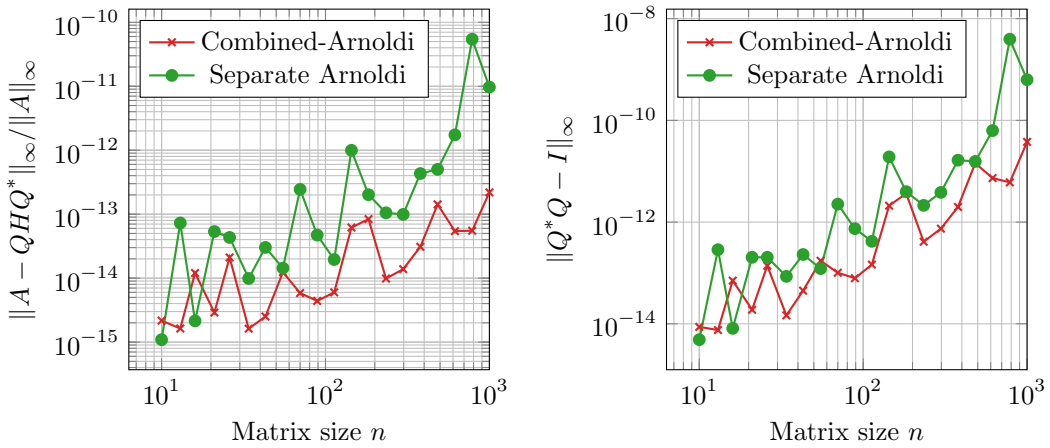
9: **else**

10: $h_{k+1,k} = \|w_k\|_2$

11: **end if**

12: $q_{k+1} = w_{k+1}/h_{k+1,k}$ % Normalize the generated vector.

13: **end for**




(a) Backward error (b) Orthogonality error

Figure 1: Behavior of the backward error and the orthogonality error for computed Hessenberg reduction using combined and separate version of Algorithm 1. The test matrix is generated with `randn(n)`.

where $Q_k = [q_1, \ldots, q_k]$ and $H_k = (h_{ij})$ is $k \times k$ upper Hessenberg. From (2.3), we have

$$AQ_k v = Q_k H_k v + h_{k+1,k} q_{k+1} e_k^T v = \lambda Q_k v + h_{k+1,k} v_k q_{k+1},$$

where $v_k$ is the $k$th entry of $v$, so that

$$\|(A - \lambda I)Q_k v\|_2 = |h_{k+1,k}||v_k|. \tag{2.4}$$

Therefore, if one wants to test the convergence of the eigensolver, it is no need to form $Q_k v$. Only monitoring the subdiagonal elements of $H$ and the last coordinate of its eigenvalue is sufficient.

If $h_{k+1,k} = 0$, then (2.4) implies $(\lambda, Q_k v)$ is an exact eigenpair of $A$. In this case, $AQ_k = Q_k H_k$ by (2.3), so the columns of $Q_k$ span an invariant subspace of $A$, and all the eigenvalues of $H$ are eigenvalues of $A$.

# 3 The Generalized Minimal Residual Method

Any vector $x$ in the shifted Krylov subspace $x_0 + \mathcal{K}_k(A, r_0)$ can be written as $x = x_0 + Q_k y$ for some $y \in \mathbb{C}^k$, where columns of $Q_k \in \mathbb{C}^{n \times k}$ are a basis for $\mathcal{K}_k(A, r_0)$. The GMRES method minimizes

$$r = b - Ax = b - Ax_0 - AQ_k y = r_0 - AQ_k y \tag{3.1}$$

over all $y$. In order to do so, it needs to build the matrix $Q_k$ using the Arnoldi process (Algorithm 1).

Suppose we run the Arnoldi process with $q_1 = r_0/\beta$, where $\beta = \|r_0\|_2$. We can rewrite (2.3) as

$$AQ_k = Q_k \widetilde{H}_k, \qquad \widetilde{H}_k = \begin{bmatrix} H_k \\ h_{k+1,k} e_k^T \end{bmatrix} \tag{3.2}$$

Then using (3.1) and (3.2), we have

$$r = r_0 - AQ_k y \tag{3.3}$$
$$= r_0 - Q_{k+1}\widetilde{H}_k y$$
$$= \beta q_1 - Q_{k+1}\widetilde{H}_k y$$
$$= Q_{k+1}(\beta e_1 - \widetilde{H}_k y) \tag{3.4}$$

and hence

$$\|r\|_2 = \|\beta e_1 - \widetilde{H}_k y\|_2,$$

since $Q_{k+1}$ has orthonormal columns. Then minimizing $r$ over all $y$ reduces to a least square problem with an $(m+1) \times m$ Hessenberg coefficient matrix, which can be solved by Givens QR factorization in $O(m^2)$ flops. **The GMRES method combines the Arnoldi process with solution of the least square problem (3.4).**

**Algorithm 2.** *GMRES - Arnoldi process with least square solution*

**Input:** $A \in \mathbb{C}^{n\times n}$, $b \in \mathbb{C}^n$ and a starting vector $x_0 \in \mathbb{C}^n$.
**Output:** A solution $x \in \mathbb{C}^n$ that solves the linear system $Ax = b$.

1: $m = n$, $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$, $q_1 = r_0/\beta$
2: **for** $k = 1 : m$ **do**
3: $\quad w_k = Aq_k$
4: $\quad$ **for** $i = 1 : k$ **do**
5: $\quad\quad h_{i,k} = q_i^* w_k$
6: $\quad\quad w_k = w_k - h_{ik} q_i$
7: $\quad$ **end for**
8: $\quad h_{k+1,k} = \|w_k\|_2$
9: $\quad$ **if** $h_{k+1,k} = 0$ **then**
10: $\quad\quad$ Set $m = k$ and quit the loop
11: $\quad$ **end if**
12: $\quad q_{k+1} = w_k / h_{k+1,k}$
13: **end for**
Until now, this is simply Algorithm 1, the Arnoldi process for constructing a Hessenberg reduction. Now, we have $AQ_k = Q_k \widetilde{H}_k$ as shown in (3.2).
14: Compute the solution $y_m$ for the least square problem $\min_{y_m} \|\beta e_1 - \widetilde{H}_m y_m\|_2$. Here, we set $\widetilde{H}_m = H_m$ if $h_{m+1,m} = 0$.
15: Recover the solution via $x = x_0 + Q_{m+1} y_m$ or $x = x_0 + Q_m y_m$ if $h_{m+1,m} = 0$.

GMRES requires $O(mn^2)$ flops for a full matrix (the dominant cost is the matrix-vector product on Line 3) and $O(mn)$ elements of storage for $Q_m$. If $m$ is large, the computation and storage costs are both prohibitive. Therefore, it is common to terminate the algorithm prematurely and restart it: the iteration stops after $p \ll n$ iterations and reruns using the current approximation $x_p$ as a new starting vector $x_0$. The method with restarts after every $k$ iterations is denoted by **GMRES(k)**.

# References


[1] Nicholas J. Higham. Numerical linear algebra and matrix analysis. Unfinished book (Version 2023 June 23), 2023. (Cited on p. 1)