mathematics

MDPI

Article

A Randomized Q-OR Krylov Subspace Method for Solving Nonsymmetric Linear Systems

Gérard Meurant

Retired Researcher, 75012 Paris, France; gerard.meurant@gmail.com

Abstract: The most popular iterative methods for solving nonsymmetric linear systems are Krylov methods. Recently, an optimal Quasi-ORthogonal (Q-OR) method was introduced, which yields the same residual norms as the Generalized Minimum Residual (GMRES) method, provided GMRES is not stagnating. In this paper, we study how to introduce matrix sketching in this algorithm. It allows us to reduce the dimension of the problem in one of the main steps of the algorithm.

Keywords: linear systems; Krylov methods; Q-OR algorithm; randomization; matrix sketching

MSC: 65F10

Academic Editors: Qing-Wen Wang and Luca Gemignani

Received: 22 April 2025
Revised: 4 June 2025
Accepted: 11 June 2025
Published: 12 June 2025

Citation: Meurant, G. A Randomized Q-OR Krylov Subspace Method for Solving Nonsymmetric Linear Systems. *Mathematics* **2025**, *13*, 1953. https://doi.org/10.3390/math13121953

1. Introduction

Let A be a real nonsingular nonsymmetric matrix of order n. So far, the most popular iterative methods for solving a nonsymmetric linear system $Ax = b$, where b is a given real vector, are Krylov methods. Many of them can be classified as Quasi-ORthogonal (Q-OR) methods or Quasi-Minimal Residual (Q-MR) methods (see, for instance, [1]). All these methods use the same framework but differ by the basis which is chosen. Different possibilities for computing the basis are described in [1] (Chapter 4). Well-known examples of Krylov methods are FOM [2,3] and GMRES [4], which use an orthonormal basis. In [5], a Q-OR optimal method that minimizes the residual norm using a non-orthogonal basis was proposed. In most cases, it must give the same residual norms as GMRES, which also minimizes the residual norm, but uses an orthonormal basis computed with the Arnoldi process (see [4]).

In recent years, randomization techniques have been proposed to reduce the dimension of some problems in numerical linear algebra (see [6–8]). In this paper, we study how to introduce randomization and matrix sketching in the Q-OR optimal algorithm. Sketching is used to solve a least squares subproblem that must be solved at each iteration.

Section 2 recalls the Q-OR optimal method [1,5]. In Section 3, we describe some known techniques for matrix sketching. Section 4 shows how to use these techniques in the Q-OR optimal method. This is illustrated by a few numerical experiments described in Section 5, showing that, even though some monotonicity properties are lost, convergence is preserved for the randomized algorithm.

2. The Q-OR Optimal Method

Let $r_0 = b - Ax_0$ be the initial residual vector. Let us assume that we have an ascending basis of the nested Krylov subspaces $\mathcal{K}_k(A, r_0)$, which are defined as

$$\mathcal{K}_k(A, r_0) = \text{span}\{r_0, Ar_0, A^2r_0, \ldots, A^{k-1}r_0\}, \quad k = 1, 2, \ldots$$

The dimension of these subspaces rises to $k_{\max} \leq n$, known as the grade of A with respect to r_0. This means that, if $v_1, \ldots, v_k$ are the basis vectors of $\mathcal{K}_k(A, r_0)$, then $v_1, \ldots, v_k, v_{k+1}$ are the basis vectors for $\mathcal{K}_{k+1}(A, r_0)$ as long as $k+1 \leq k_{\max}$.

Such basis vectors satisfy what is called an Arnoldi relation,

$$AV_k = V_k H_k + h_{k+1,k} v_{k+1} e_k^T = V_{k+1} \underline{H}_k, \tag{1}$$

where H_k is an upper Hessenberg matrix with entries $h_{i,j}$, the columns of V_k are the basis vectors $v_1, \ldots, v_k$, and e_k is the last column of the identity matrix of order k. The matrix $\underline{H}_k$ is H_k, appended at the bottom with a $k+1$st row equal to $h_{k+1,k} e_k^T$.

The iterates x_k, $k \geq 1$ in Q-OR and Q-MR methods are sought as

$$x_k = x_0 + V_k y_k, \tag{2}$$

for some unique vector $y_k \in \mathbb{R}^k$. Since we choose $v_1 = r_0 / \|r_0\|$, the residual vector r_k, defined as $r_k = b - Ax_k$, is

$$\begin{aligned} r_k &= b - Ax_k \\ &= b - Ax_0 - AV_k y_k \\ &= \|r_0\| V_k e_1 - AV_k y_k \\ &= V_k(\|r_0\| e_1 - H_k y_k) - h_{k+1,k}[y_k]_k v_{k+1}. \end{aligned} \tag{3}$$

In a Q-OR method, the kth iterate x_k^O is defined (provided that H_k is nonsingular) by computing $y_k = y_k^O$ in (2) as the solution of the linear system

$$H_k y_k = \|r_0\| e_1. \tag{4}$$

This annihilates the term within the parentheses on the right side of (3). The iterates of the Q-OR method are $x_k^O = x_0 + \|r_0\| V_k H_k^{-1} e_1$, the residual vector r_k^O is proportional to v_{k+1}, and

$$\|r_k^O\| = h_{k+1,k} \left| [y_k^O]_k \right|. \tag{5}$$

In the case where H_k is singular and x_k^O is not defined, we define the residual norm as being infinite, $\|r_k^O\| = \infty$.

The residual vector in relation (3) can also be written as

$$r_k = V_{k+1}(\|r_0\| e_1 - \underline{H}_k y_k). \tag{6}$$

Instead of removing the term within the parentheses on the right side of (3), we would like to minimize the norm of the residual itself. This is what is carried out in GMRES with an orthonormal basis [4]. Minimizing the norm of the residual may seem as costly when the columns of the matrix V_{k+1} are not orthonormal. However, we have

$$\|r_k\| \leq \|V_{k+1}\| \, \| \|r_0\| e_1 - \underline{H}_k y_k \|.$$

In a general Q-MR method, the vector y_k^M is computed as the solution of the least squares problem:

$$\min_y \| \|r_0\| e_1 - \underline{H}_k y \|. \tag{7}$$

Note that y_k^M does not minimize the norm of the residual, but the norm of what is called the quasi-residual, as follows:

$$z_k^M = \|r_0\| e_1 - \underline{H}_k y_k^M. \tag{8}$$

The Q-MR iterates are always defined as opposite to the Q-OR iterates when H_k is singular. Note that the preceding definitions do not depend on the choice of the basis. It is

a general framework that could use any basis. Q-OR and Q-MR methods, as well as their many interesting mathematical properties, are studied in detail in [1].

The Hessenberg matrices H_k are unreduced since $h_{j+1,j} \neq 0$ for $j = 1, \ldots, k-1$. Therefore, they are nonderogatory and can be factorized as $H_k = U_k C^{(k)} U_k^{-1}$, where U_k is an upper triangular matrix with $|U_k|_{1,1} = 1$, and $C^{(k)}$ is a companion matrix corresponding to the characteristic polynomial of H_k (see [1]). The matrix U_k is, in fact, a Krylov matrix:

$$U_k = \begin{pmatrix} e_1 & H_k e_1 & H_k^2 e_1 & \ldots & H_k^{k-1} e_1 \end{pmatrix}.$$

Clearly, U_k is the principal matrix of order k of U_{k+1}. Let $\vartheta_{1,j}$ be the entries of the first row of U_{k+1}^{-1}. It is proved in [1,5] that, whatever the basis of the Krylov subspace is, the Q-OR residual norms satisfy

$$\frac{\|r_k^O\|}{\|r_0\|} = \frac{1}{|\vartheta_{1,k+1}|}, \quad k = 0, 1, \ldots$$

As shown in [1,5], there exists a non-orthogonal basis such that $|\vartheta_{1,k+1}|$ is maximized. Therefore, this minimizes the Q-OR residual norm. Assuming that $\vartheta_{1,k} \neq 0$ and $v_k^T A v_k \neq 0$, it can be computed as follows:

$$\tilde{v}_k = A v_k - V_k s - \beta v_k, \quad v_{k+1} = \frac{\tilde{v}_k}{\|\tilde{v}_k\|},$$

with

$$V_k^T V_k s = V_k^T A v_k, \tag{9}$$

and

$$\beta = \frac{\alpha}{v_k^T A v_k}, \quad \alpha = \|A v_k\|^2 - (V_k^T A v_k)^T s.$$

The k first entries of the kth column of the upper Hessenberg matrix $\underline{H}_k$ are $h_{1:k,k} = s + \beta e_k$, and $h_{k+1,k} = \|\tilde{v}_k\|$. Moreover, we have $\vartheta_{1,1} = 1$, and

$$\vartheta_{1,k+1} = -\frac{1}{h_{k+1,k}} \sum_{j=1}^{k} \vartheta_{1,j} h_{j,k}, \quad k = 1, \ldots, n-1.$$

At iteration k, we have to solve the linear system (9) whose matrix is symmetric-positive-definite as long as V_k is of rank k. In [5], this linear system was solved by incrementally computing the inverses of the triangular factors of the Cholesky factorization of $V_k^T V_k$. The details of the method, as described in [5], are shown as Algorithm 1. In this algorithm, the matrix L_k contains the inverse of the Cholesky factor of $V_k^T V_k$. Preconditioning can be easily incorporated each time we have a product of the matrix A with a vector.

Note that the modulus of $\vartheta_{1,k+1}$ gives the inverse of the (relative) norm of the Q-OR residual at iteration k. Hence, we can compute the basis vectors v_k, stop the iterations using $\vartheta_{1,k+1}$, and then reduce the upper Hessenberg matrix to an upper triangular form to compute the final approximate solution.

This method is named Q-ORoptinv because it minimizes the residual norm and uses the inverses of Cholesky factors. When, for all k, $v_k^T A v_k \neq 0$, it must give the same residual norms as GMRES. The reader may wonder why we have derived an algorithm which delivers the same residual norms as GMRES but with more floating point operations. The reason is that the dot products in Q-ORoptinv are all independent and they can be computed in parallel, contrary to the dot products in the modified Gram–Schmidt (MGS) implementation of GMRES.

As in GMRES, the storage increases at every iteration, so the algorithm can be restarted every m iterations to limit the needed storage.

Algorithm 1 Q-ORoptinv.

input A, b, x_0
—Initialization
$r_0 = b - Ax_0$
$v_1 = r_0/\|r_0\|$, $v_1^A = Av_1$, $L_1 = 1$
$\omega = v_1^T v_1^A$, $\alpha = (v_1^A)^T v_1^A - \omega^2$
$h_{1,1} = \omega + \frac{\alpha}{\omega}$
$\tilde{v} = v_1^A - h_{1,1} v_1$, $h_{2,1} = \|\tilde{v}\|$
$v_2 = \frac{1}{h_{2,1}} \tilde{v}$, $v_2^A = Av_2$
$V_2 = \begin{pmatrix} v_1 & v_2 \end{pmatrix}$
$\vartheta_{1,1} = 1$, $\vartheta_{1,2} = -\frac{h_{1,1}}{h_{2,1}}$
$\vartheta = \begin{pmatrix} \vartheta_{1,1} & \vartheta_{1,2} \end{pmatrix}^T$
—End of initialization
for $k = 2, \ldots$ until convergence **do**
  $v_k^V = V_{k-1}^T v_k$, $v_k^{tA} = V_k^T v_k^A$
  $\ell_k = L_{k-1} v_k^V$, $y_k^T = \ell_k^T L_{k-1}$
  if $\ell_k^T \ell_k < 1$ **then**
    $\ell_{k,k} = \sqrt{1 - \ell_k^T \ell_k}$
  else
    $(p_k^v)^T = y_k^T V_{k-1}^T$, $\ell_{k,k} = \|v_k - p_k^v\|$
  end if
  $L_k = \begin{pmatrix} L_{k-1} & 0 \\ -\frac{1}{\ell_{k,k}} y_k^T & \frac{1}{\ell_{k,k}} \end{pmatrix}$
  $\ell_A = L_k v_k^{tA}$, $s = L_k^T \ell_A$
  $\alpha = (v_k^A)^T v_k^A - \ell_A^T \ell_A$, $\beta = \frac{\alpha}{(v_k^{tA})_k}$
  $h_{1:k,k} = \begin{pmatrix} h_{1,k} \\ \vdots \\ h_{k,k} \end{pmatrix} = s + \beta e_k$
  $\tilde{v} = v_k^A - V_k h_{1:k,k}$, $h_{k+1,k} = \|\tilde{v}\|$
  $\vartheta_{1,k+1} = -\frac{1}{h_{k+1,k}} \vartheta^T h_{1:k,k}$
  $\vartheta = \begin{pmatrix} \vartheta_{1,1} & \cdots & \vartheta_{1,k+1} \end{pmatrix}^T$
  $v_{k+1} = \frac{1}{h_{k+1,k}} \tilde{v}$, $v_{k+1}^A = Av_{k+1}$
  $V_{k+1} = \begin{pmatrix} V_k & v_{k+1} \end{pmatrix}$
  if needed, solve $H_k y^{(k)} = \|r_0\| e_1$, $x_k = x_0 + V_k y^{(k)}$
end for

The solution s of Equation (9) is also the solution of the least squares problem:

$$\min_{y \in \mathbb{R}^k} \|V_k y - Av_k\|, \tag{10}$$

since (9) is the normal equation corresponding to (10). Hence, we can use the economy size QR factorization of V_k to solve (10) with an upper triangular matrix R of order k instead of using the inverses of the Cholesky factors of $V_k^T V_k$. Since the method is often restarted with $m \ll n$, meaning that the number of columns k is small compared to the number of rows, V_k is what is called a tall-and-skinny matrix. There exist special algorithms for computing the QR factorization of such matrices that can be used on parallel computers (see [9,10]). Note that the columns of Q give an orthogonal basis of the Krylov subspace. So, if we use the QR factorization, we are more or less back to what is completed in GMRES. When the restart parameter m is large, or when there is no restart, using the QR factorization may be too expensive. However, at each iteration, we only add one more column to the matrix V_k. There exist algorithms for updating the QR factorization when we add a new column to

the matrix (see [11,12]). This can be carried out, for instance, by orthogonalizing the new column against the columns of the previous matrix Q with the modified Gram–Schmidt algorithm. This is what we used in our numerical experiments.

3. Random Sketching

Since the matrix V_k in the least squares problem (10) is tall and skinny, it may be useful to use a random sketching, a technique that was introduced during the last twenty years. This is used to reduce the dimension of the problem (see, for instance, [6]). A sketching matrix S is of order $\ell \times n$ with $\ell \ll n$. Let $\mathcal{V}$ be a subspace of $\mathbb{R}^n$. The matrix S is an ε-embedding of $\mathcal{V}$ if

$$| \, \|Sv\| - \|v\| \, | \leq \varepsilon \, \|v\|, \quad \forall v \in \mathcal{V}, \tag{11}$$

where $0 < \varepsilon < 1$. Generally, ε-embeddings are constructed with probabilistic techniques to be independent of the subspace $\mathcal{V}$ with a high probability. They are called oblivious ε-embeddings. There are several distributions for constructing such embeddings, such as Gaussian ones and the subsampled randomized Hadamard transform (SRHT) [13].

SRHT is constructed with Hadamard matrices. These matrices are defined recursively. Starting with $H = 1$, and having a Hadamard matrix H, the next matrix is

$$\begin{pmatrix} H & H \\ H & -H \end{pmatrix}.$$

Therefore, their order is always a power of 2. Let p be an integer such that 2^p is the smallest power of 2 larger than or equal to n. The $\ell \times 2^p$ SRHT matrix $\tilde{S}$ is

$$\tilde{S} = \frac{1}{\sqrt{\ell}} PHD,$$

where D is a random diagonal matrix with diagonal entries ± 1, H is a Hadamard matrix, and P is a random uniform subsampling matrix. The constant in front of PHD depends on the way the Hadamard matrix is scaled. For our purposes, the sketching matrix S is made of the first n columns of $\tilde{S}$. We apply $\tilde{S}$ to a vector with only the first n components, which are nonzero. The multiplication by H is carried out using the fast Walsh–Hadamard transform. It uses the recursive structure of H to evaluate the product in $N \, \log_2(N)$ operations with $N = 2^p$. The problem with this sketching matrix is that 2^p can be much larger than n.

Another possibility is to use the Clarkson–Woodruff transform [8,14]. The matrix S is an $\ell \times n$ sparse matrix with only one nonzero entry in each column which is ± 1 with probability $1/2$. The row number of the entry is chosen randomly. For the first ℓ columns of S, a random permutation of $[1, 2, \ldots, \ell]$ is chosen.

A delicate issue with matrix sketching is the choice of ℓ. It is known that inequality (11) is satisfied for SRHT with probability $1 - \delta$ if

$$\ell = O\left(\varepsilon^{-2}\left(k + \log \frac{N}{\delta}\right) \, \log \frac{k}{\delta}\right),$$

where k is the dimension of the subspace $\mathcal{V}$. However, this is of little help for us since we need the same sketching matrix S for all iterations and the subspace dimension is increasing by one every iteration. If the Q-OR method is restarted, k may be chosen as the restart parameter m. However, this may be too small to obtain a fast convergence. We will show experimentally in Section 5 how the choice of ℓ influences the convergence of the sketched Q-OR method.

In numerical linear algebra, matrix sketching has been mainly used with some successes for solving large least squares problems. In recent years, randomization has also been used in different Krylov methods for solving linear systems. However, methods such as randomized GMRES [15] or sketched GMRES [7] do not minimize the residual norm

as in GMRES. Hence, they are misnamed. In fact, some of them are Q-MR methods with non-orthogonal bases.

4. The Randomized Q-OR Method

A randomized variant of the Q-OR optimal method can be carried out by simply replacing (10) with

$$\min_{y \in \mathbb{R}^k} \|SV_k y - SAv_k\|, \tag{12}$$

where S is an $\ell \times n$ sketching matrix that is computed before running the algorithm. The matrix SV_k can be computed incrementally since $SV_k = \begin{pmatrix} SV_{k-1} & Sv_k \end{pmatrix}$. Thus, there are only two matrix–vector products with S per iteration. Now, it makes more sense to use a QR factorization to solve the least squares problem (12) because it is of a smaller dimension than (10). Of course, the basis that is obtained is no longer optimal, and the method does not minimize the residual norm. However, since $\|S(V_k y - Av_k)\| \approx \|V_k y - Av_k\|$, the convergence of the method must not be too different, even though the decrease in the residual norm may not be monotone, as we will see with the numerical experiments detailed the next section.

The sketched algorithm is described as Algorithm 2. In statement 12, the QR factorization is simply a normalization of the vector v_1^S, and the initial matrix R is a scalar, i.e., the norm of v_1^S. Statement 17 is an update of the QR factorization when we append a new vector v_k^S to the previous matrix. This can be completed in different ways. In numerical experiments, we use a modified Gram–Schmidt implementation of the update. Note that the first dimension ℓ of the sketching matrix must be larger than the iteration number k.

Algorithm 2 Q-ORsketch.

1: **input** A, b, x_0, S
2: —Initialization
3: $r_0 = b - Ax_0$
4: $v_1 = r_0/\|r_0\|, \; v_1^A = Av_1, \;\; v_1^S = Sv_1$
5: $\omega = v_1^T v_1^A, \;\; \alpha = (v_1^A)^T v_1^A - \omega^2$
6: $h_{1,1} = \omega + \frac{\alpha}{\omega}$
7: $\tilde{v} = v_1^A - h_{1,1} v_1, \;\; h_{2,1} = \|\tilde{v}\|$
8: $v_2 = \frac{1}{h_{2,1}} \tilde{v}, \;\; v_2^A = Av_2$
9: $V_2 = \begin{pmatrix} v_1 & v_2 \end{pmatrix}$
10: $\vartheta_{1,1} = 1, \;\; \vartheta_{1,2} = -\frac{h_{1,1}}{h_{2,1}}$
11: $\vartheta = \begin{pmatrix} \vartheta_{1,1} & \vartheta_{1,2} \end{pmatrix}^T$
12: $[Q, R] = QR(v_1^S)$
13: —End of initialization
14: **for** $k = 2, \ldots$ until convergence **do**
15: $\quad v_k^V = V_{k-1}^T v_k, \;\; v_k^{tA} = V_k^T v_k^A$
16: $\quad v_k^S = Sv_k, \;\; v_k^{SA} = Sv_k^A$
17: $\quad [Q, R] = \text{update\_}QR(Q, R, v_k^S)$
18: $\quad s = R^{-1}(Q^T v_k^{SA})$
19: $\quad \alpha = (v_k^A)^T v_k^A - (v_k^A)^T (V_k s), \;\; \beta = \frac{\alpha}{(v_k^{tA})_k}$
20: $\quad h_{1:k,k} = \begin{pmatrix} h_{1,k} \\ \vdots \\ h_{k,k} \end{pmatrix} = s + \beta e_k$
21: $\quad \tilde{v} = v_k^A - V_k h_{1:k,k}, \;\; h_{k+1,k} = \|\tilde{v}\|$
22: $\quad \vartheta_{1,k+1} = -\frac{1}{h_{k+1,k}} \vartheta^T h_{1:k,k}$
23: $\quad \vartheta = \begin{pmatrix} \vartheta_{1,1} & \cdots & \vartheta_{1,k+1} \end{pmatrix}^T$
24: $\quad v_{k+1} = \frac{1}{h_{k+1,k}} \tilde{v}, \; v_{k+1}^A = Av_{k+1}$
25: $\quad V_{k+1} = \begin{pmatrix} V_k & v_{k+1} \end{pmatrix}$
26: $\quad$ if needed, solve $H_k y^{(k)} = \|r_0\| e_1, \;\; x_k = x_0 + V_k y^{(k)}$
27: **end for**

5. Numerical Experiments

For the first experiment, we consider the matrix `fs_680_1` (https://sparse.tamu.edu, URL accessed on 1 January 2025). We scale this matrix to have a unit diagonal and name it `fs_680_1c`. This sparse matrix of order 680 has 21,184 nonzero entries and a condition number equal to 8.6944×10^3. Figure 1 shows the true residual norms $\|b - Ax_k\|$ for the standard Q-ORoptinv method using the inverses of Cholesky factors and the randomized method using SRHT sketching without preconditioning and without restarting. The initial iterate is the zero vector. Note that for SRHT, $N = 1204$ when $n = 680$. The value of ℓ is $n/4 = 170$. Using Clarkson–Woodruff sketching provides almost the same results. The residual norms of the two algorithms are almost similar, but since the method with sketching does not minimize the residual norm, it is slightly larger and with small oscillations.

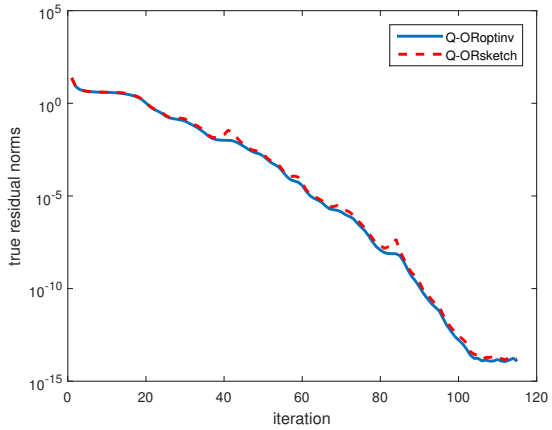

Figure 1. `fs_680_1c`, true residual norms, Q-ORoptinv (solid), Q-OR with SRHT sketching, $\ell = n/4$ (dashed).

Figure 2 displays the true residual norms for the method with SRHT sketching for $\ell = n/2, n/4, n/8$, and $n/16$. Note that $680/8 = 85$ and $\lceil 680/16 \rceil = 43$. This limits the number of iterations that we can perform with these small values of ℓ. In fact, one can see that, after 43 iterations, the algorithm with $\ell = n/16$ does not converge. The results with $n/2$, $n/4$, and $n/8$ are more or less the same, showing that the algorithm is only weakly dependent on the choice of ℓ. However, with $\ell = n/8$, we cannot perform much more than 85 iterations.

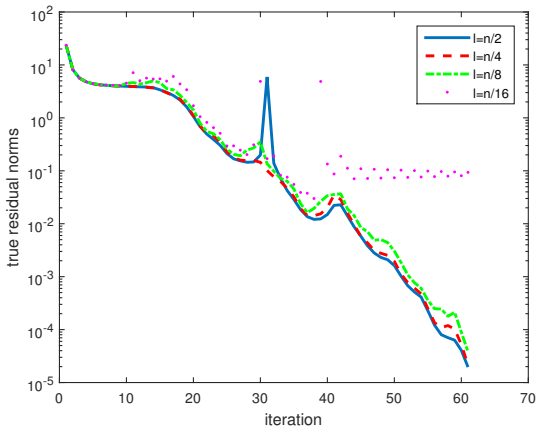

Figure 2. `fs_680_1c`, true residual norms, Q-OR with SRHT sketching, $\ell = n/2$ (solid), $n/4$ (dashed), $n/8$ (dash-dotted), $n/16$ (dotted).

For the second example, we consider the matrix `rajat27` (https://sparse.tamu.edu) of order 20,640. Since this matrix has some zero entries on the diagonal and this can be a problem for some preconditioners, we add $2I$ to the matrix, and we name it `rajat27b`. This matrix has 101,681 nonzero entries and an estimated condition number equal to 4.8588×10^7. We use a diagonal preconditioner.

Figure 3 shows the computed residual norms (using relation (5)) for the standard Q-ORoptinv method and the randomized method using SRHT sketching. Once again, the method with sketching converges similarly to the standard method.

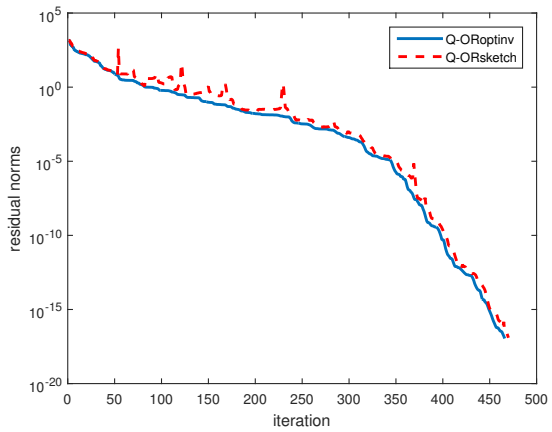

Figure 3. `rajat27b`, diagonal preconditioner, computed residual norms, Q-ORoptinv (solid), Q-OR with SRHT sketching, $\ell = n/4$ (dashed).

Figure 4 displays the computed residual norms for the method with SRHT sketching for $\ell = n/4, n/8, n/16$, and $n/32$. Note that all these values of ℓ are larger than the number of iterations we have to perform. The results with these values of ℓ are more or less the same, showing, once again, that the randomized algorithm is only weakly dependent on the choice of ℓ.

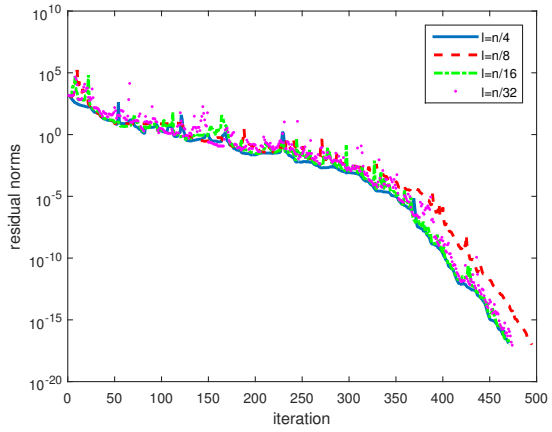

Figure 4. `rajat27b`, diagonal preconditioner, computed residual norms, Q-OR with SRHT sketching, $\ell = n/4$ (solid), $n/8$ (dashed), $n/16$ (dash-dotted), $n/32$ (dotted).

Figure 5 compares SRHT and Clarkson–Woodruff sketching. The two algorithms converge similarly, but more oscillations occur with Clarkson–Woodruff sketching. However, it is cheaper than SRHT.

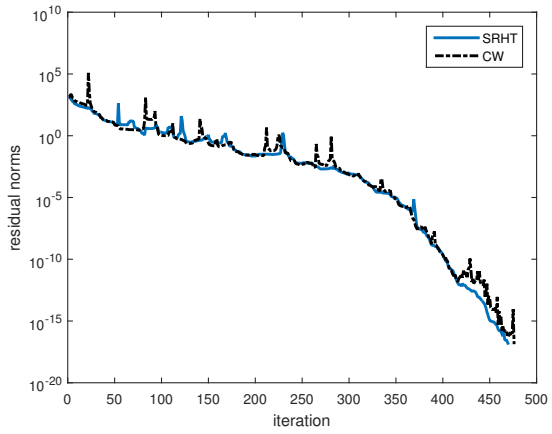

Figure 5. `rajat27b`, diagonal preconditioner, computed residual norms, Q-OR with sketching, SRHT (solid), Clarkson–Woodruff (dot-dashed).

The third example corresponds to the finite difference discretization of a convection–diffusion equation,

$$-\frac{\partial}{\partial x}\left(\lambda(x,y)\frac{\partial u}{\partial x}\right)-\frac{\partial}{\partial y}\left(\lambda(x,y)\frac{\partial u}{\partial y}\right)+\frac{\partial u}{\partial x}+\frac{\partial u}{\partial y}=f \quad \text{in } [0,1]^2,$$

with homogeneous Dirichlet boundary conditions. The diffusion coefficient $\lambda(x,y)$ is piecewise constant, being equal to 100 in $[1/4,3/4]^2$ and 1 elsewhere. The mesh size is $h = 1/151$, providing a matrix of order 22,500. Its estimated condition number is 9.3909×10^5. The right-hand side is a random vector.

We use an incomplete LU preconditioner without fill-in (ILU(0)) and we restart the methods every 100 iterations. Figure 6 shows that, even though there are some oscillations with the method using sketching, the convergence is very similar to that of Q-ORoptinv.

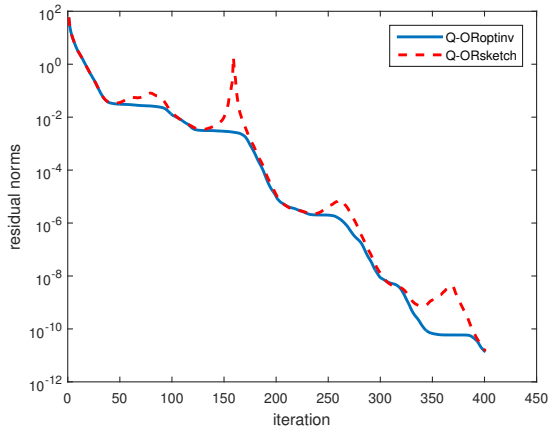

Figure 6. `convection-diffusion`, ILU(0) preconditioner, computed residual norms, Q-ORoptinv (solid), Q-OR with SRHT sketching, $\ell = n/4$ (dashed), $m = 100$.

6. Conclusions

In this paper, we have shown how to use the technique of matrix sketching in the Krylov method Q-ORoptinv for solving nonsymmetric linear systems. This was accomplished to reduce the complexity of an important part of the algorithm. Even though the sketched method does not minimize the norm of the residual, it converges almost as fast as the genuine method, as demonstrated by the numerical experiments. This new variant of the method can be interesting when solving large nonsymmetric linear systems.

Funding: This research received no external funding.

Data Availability Statement: No available data.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Meurant, G.; Duintjer Tebbens, J. *Krylov Methods for Nonsymmetric Linear Systems*; Springer Series in Computational Mathematics; Springer International Publishing: Cham, Switzerland, 2020; Volume 57.
2. Saad, Y. Krylov subspace methods for solving large nonsymmetric linear systems. *Math. Comput.* **1981**, *37*, 105–126. [CrossRef]
3. Saad, Y. Practical use of some Krylov subspace methods for solving indefinite and nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* **1984**, *5*, 203–228. [CrossRef]
4. Saad, Y.; Schultz, M.H. GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* **1986**, *7*, 856–869. [CrossRef]
5. Meurant, G. An optimal Q-OR Krylov subspace method for solving linear systems. *Electron. Trans. Numer. Anal.* **2017**, *47*, 127–152. [CrossRef]
6. Halko, N.; Martinsson, P.G.; Tropp, J.A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.* **2011**, *53*, 217–288. [CrossRef]
7. Nakatsukasa, Y.; Tropp, J.A. Fast and accurate randomized algorithms for linear systems and eigenvalue problems. *SIAM J. Matrix Anal. Appl.* **2024**, *45*, 1183–1214. [CrossRef]
8. Woodruff, D.P. Sketching as a tool for numerical linear algebra. *Found. Trends Theor. Comput. Sci.* **2014**, *10*, 1–157. [CrossRef]
9. Buttari, A.; Langou, J.; Kurzak, J.; Dongarra, J. Parallel tiled QR factorization for multicore architectures. *Concurr. Comput.-Pract. Exp.* **2008**, *20*, 1573–1590. [CrossRef]
10. Demmel, J.W.; Grigori, L.; Hoemmen, M.; Langou, J. Communication-optimal parallel and sequential QR and LU factorizations. *SIAM J. Sci. Comput.* **2012**, *34*, A206–A239. [CrossRef]
11. Hammarling, S.; Higham, N.J.; Lucas, C. LAPACK-style codes for pivoted Cholesky and QR updating. In Proceedings of the International Workshop on Applied Parallel Computing, Umea, Sweden, 18–21 June 2006; Springer: Berlin/Heidelberg, 2006; pp. 137–146.
12. Hammarling, S.; Lucas, G. *Updating the QR Factorization and the Least Squares Problem*; Technical Report MIMS Eprint 2008-111; Manchester Institute for Mathematical Sciences, University of Manchester: Manchester, UK, 2008.
13. Tropp, J. Improved analysis of the subsampled randomized Hadamard transform. *Adv. Adapt. Data Anal.* **2011**, *3*, 115–126. [CrossRef]
14. Clarkson, K.L.; Woodruff, D.P. Low-rank approximation and regression in input sparsity time. *J. ACM* **2017**, *63*, 1–45. [CrossRef]
15. Balabanov, O.; Grigori, L. Randomized Gram-Schmidt process with application to GMRES. *SIAM J. Sci. Comput.* **2022**, *44*, A1450–A1474. [CrossRef]