Rank Revealing QR Factorization

Zane Perry, Luke Sellmayer, Maedée Trank-Greene

December 19, 2023

1 Abstract

In the field of numerical linear algebra, rank-revealing factorizations have emerged as a powerful tool across various fields, from machine learning to reduced order modeling. Traditional methods such as QR decomposition and SVD are computationally expensive methods for large matrices. The motivation behind implementing the RRQRF is due to it being a less computationally expensive numerical method.

Inspired by recent machine-learning techniques extracting components from signals, we chose to implement signal processing with RRQRFs for our independent extension. Our application is based on the paper by Per Christian Hansen and Søren Holdt Jensen that implements subspace-based noise reductions via diagonal and triangular matrix decompositions for signal enhancement, which estimates a clean signal from a noisy one [2]. The RRFs used in the paper included eigenvalue decomposition, SVD, and many triangular decompositions. We implement their algorithm with the RRQRF. Opting to use the RRQRF should be a computationally less expensive method and reduce the time complexity compared to the SVD.

First, the results of our permuted QR algorithms showed that using Householder reflection matrices provided the best balance between speed of factorization and stability when compared against both matrix size and matrix rank. However, using the permuted QR as the size of the rank of the matrix increases is less efficient and has a larger absolute error compared to the normal QR.

2 Introduction

In the field of numerical linear algebra, rank-revealing factorizations have emerged as a powerful tool across various fields, from machine learning to reduced order modeling. One of these factorizations is based on the standard QR decomposition of a matrix, which tends to be more computationally efficient than other matrix factoring methods such as the Singular Value Decomposition. The basic concept behind a rank-revealing factorization is to factor a matrix in a way that orders certain elements, often related to the singular values of that matrix, by magnitude to detect entries that can be discarded for being negligible relative to the range of the original matrix. By utilizing the efficiency of QR decompositions, the structure of the resulting R matrix can be exploited to provide this property.

In this project, we aim to dig deep into the theoretical framework of rank-revealing factorizations (RRFs), specifically the QR factorizations with column pivoting. In particular, we are interested in applying the theoretical background of the RRQRF to a signal processing problem, using the RRQRF to estimate the signal into pure and noisy components. We base our application on the paper by Per Christian Hansen and Søren Holdt Jensen that implements subspace-based noise reductions via diagonal and triangular matrix decompositions for signal enhancement, which estimates a clean signal from a noisy one [2].

The RRFs used in the paper include eigenvalue decomposition, SVD, and many triangular decompositions. We implement their algorithm with the RRQRF. From our theoretical framework developed below, opting to use the QR factorization should be computationally less expensive and reduce the time complexity compared to the SVD. Our last step will be to compare the results between the two RRFs to see if the above claim holds.

3 Introductory Material

3.1 Theoretical Background for Rank Revealing Factorizations

The rank of a matrix A is defined by the number of linearly independent rows or columns of A. In other words, the rank of a matrix is the dimension of the space spanned by its rows or columns. If the rank$(A) = r$ for $A \in \mathbb{R}^{m\times n}$, then A has r linearly independent columns and r linearly independent rows [3]. In terms of singular values, a rank r matrix will have r non-zero singular values.

In contrast, the numerical rank of a matrix A is defined by the number of singular values that are greater than some $\epsilon > 0$. A matrix has a numerical rank k if there are $k \leq n$ singular values of A that are larger than ϵ. If $k < n$ then the matrix is said to be numerically rank deficient.

Rank revealing factorizations (RRFs) are a common method to compute the numerical rank of a matrix A and can be used to create a low-rank approximation of A. The RRF of an $m \times n$ matrix A can be defined as

$$A = XDY^T,\ X \in \mathbb{R}^{m\times p},\ D \in \mathbb{R}^{p\times p},\ Y \in \mathbb{R}^{n\times p} \tag{1}$$

where $p < min\{m, n\}$, D is diagonal and nonsingular, and X and Y are well conditioned. A matrix is well-conditioned if the condition number $\kappa(A) = \frac{\sigma_1}{\sigma_r}$, which is the quotient of the largest and smallest singular values of the matrix, is small. This factorization concentrates any ill-conditioning of A into the diagonal matrix D, and all determinations of numerical rank can then be obtained by inspecting the diagonal entries of D.

Commonly used RRFs include the SVD and QR decomposition. The SVD of a nonzero real matrix A of $rank(A) = r > 0$ is given by

$$A = U\Sigma V^T, U \in \mathbb{R}^{m\times r}, \Sigma \in \mathbb{R}^{r\times r}, V \in \mathbb{R}^{n\times r} \tag{2}$$

where U and V have orthonormal columns so $U^TU = V^TV = I$ and $\Sigma = diag(\sigma_1, \cdots, \sigma_r)$ where $\sigma_1 > \sigma_2 > \cdots > \sigma_r$. The SVD is an RRF with $X = U$, $Y = V$, and $D = \Sigma$.

The QR decomposition for a matrix A is

$$A = QR, Q \in \mathbb{R}^{m\times m}, R \in \mathbb{R}^{m\times n} \tag{3}$$

where Q is a matrix with orthonormal columns and R is upper triangular. The QR factorization is an RRF with $X = Q$, $D = diag(R)$, and $Y = (D^{-1}R)^T$. However, this basic definition of the RRF with QR factorizations can be flawed. For example, consider the matrix

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4}$$

The QR factorization of this matrix is

$$A = QR = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} = IA = A. \tag{5}$$

Using the definition of rank revealing QR above, this would mean that $X = I$ and $D = diag(A)$. However, the diagonal elements of A are all zero, therefore violating the requirement that D be nonsingular, and preventing the calculation of the matrix Y. To address this issue, column pivoting is incorporated into the QR factorization.

3.2 QR Factorization with Column Pivoting (QRCP)

The QR factorization with column pivoting for a matrix A is defined by

$$AP = QR \tag{6}$$

where P is a permutation matrix, $Q \in \mathbb{R}^{m\times n}$, and $R \in \mathbb{R}^{n\times n}$. As before, Q has orthonormal columns and R is upper triangular, but the diagonal entries of R also satisfy

$$|r_{11}| \geq |r_{22}| \geq \cdots \geq |r_{nn}| \tag{7}$$

meaning the diagonal entries in R are ordered by decreasing magnitude.

Previously, determining the low-rank approximation of a matrix involved identifying and discarding the singular values less than a given ϵ. Using the fact that the diagonal elements are ordered in the permuted QR factorization, the matrix R can be partitioned into blocks according to the magnitude of its diagonal entries. For some $\epsilon > 0$, if $|r_{kk}| \geq \epsilon \geq |r_{k+1,k+1}|$, then R can be written as the following block matrix

$$\begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \tag{8}$$

where $R_{11} \in \mathbb{R}^{k \times k}$, $R_{12} \in \mathbb{R}^{k \times (n-k)}$, and $R_{22} \in \mathbb{R}^{(n-k) \times (n-k)}$.

Q can also be partitioned to agree with the dimensions of the blocks of R

$$AP = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} = Q_1 \begin{bmatrix} R_{11} & R_{12} \end{bmatrix} + \begin{bmatrix} 0 & Q_2 R_{22} \end{bmatrix} \tag{9}$$

where $Q_1 \in \mathbb{R}^{m \times k}$ and $Q_2 \in \mathbb{R}^{m \times (n-k)}$

R_{11} and R_{22} are specifically chosen so that all diagonal entries of R_{11} are greater than the given tolerance, and all diagonal entries of R_{22} are smaller than that tolerance. Because R_{11} has an order of k, and subsequently k diagonal elements, by the definition of numerical rank above the numerical rank of A is k. This can then be used to create a rank-k approximation of the matrix A.

From the definition of R_{22}, the following inequality that bounds the 2-norm and the Frobenius norm can be shown

$$||R_{22}||_2 \leq ||R_{22}||_F \leq 2^{-\frac{1}{2}}(n-k+1)\epsilon. \tag{10}$$

First, start with the 2-norm of $AP\mathbf{x} - Q_1 \begin{bmatrix} R_{11} & R_{12} \end{bmatrix} \mathbf{x}$. This equation can be manipulated as

$$||AP\mathbf{x} - Q_1 \begin{bmatrix} R_{11} & R_{12} \end{bmatrix} \mathbf{x}||_2 = || \begin{bmatrix} 0 & Q_2 R_{22} \end{bmatrix} \mathbf{x}||_2 \leq ||x||_2 ||Q_2 R_{22}||_2 \tag{11}$$

Using the inequality above, the magnitude of the second block of AP can be bounded as

$$||Q_2 R_{22}||_2 \leq ||Q_2||_2 ||R_{22}||_2 = ||R_{22}||_2 \leq 2^{-\frac{1}{2}}(n-k+1)\epsilon \tag{12}$$

By approximating AP with only the first block, the error bound is shown to be

$$||AP - Q_1 \begin{bmatrix} R_{11} & R_{12} \end{bmatrix} ||_2 \leq 2^{-\frac{1}{2}}(n-k+1)\epsilon \tag{13}$$

which shows that Q_1 gives a $\mathcal{O}(\epsilon)$ approximation to the range of A.

The full Q matrix is an orthonormal basis for the range of A, with each column j corresponding to the jth diagonal element of R. Because R_{22} has been determined to be numerically negligible, the corresponding columns in Q must also have little effect on the range of A. Therefore the first k columns of Q form a low-rank approximation to the range of A.

3.3 Developing the Permuted QR algorithm

There are 3 primary ways that the standard QR factorization algorithm can be implemented: The Gram-Schmidt Process, Householder Reflection Transformations, or Given's Rotations. The general process of each, as well as the advantages and disadvantages of each, will be explored below.

3.3.1 Gram-Schmidt

The Gram-Schmidt method is well known for taking a set of vectors and forming an orthogonal basis out of them. This process is accomplished by taking each column and subtracting the parallel component of each basis vector already computed, found using the inner product. The major drawback to this algorithm is that repeated subtractions lead to a high level of numerical instability. This process can be improved slightly through the Modified Gram-Schmidt process, which normalizes each column and then orthogonalizes all of the remaining vectors. By maintaining the orthogonality of all vectors at each step, rather than orthogonalizing them individually, the stability is greatly improved.

As the modified Gram-Schmidt algorithm returns an orthonormal set of vectors, applying this algorithm to the columns of a matrix A would result in an orthonormal basis for the range of A. This basis then becomes the matrix Q, and R is easily determined by multiplying both sides of the factorization by Q^T.

Algorithm 1 QR with Modified Gram-Schmidt

```
Require: $m \times n$ matrix $A$ with columns $\{a_1, a_2, \dots, a_n\}$
  for $j = 1, 2, \dots, n$ do
      $q_j = \frac{a_j}{||a_j||}$
      for $i = j + 1, \dots, n$ do
          $a_i = a_i - \langle q_j, a_i \rangle q_j$
      end for
  end for
  $Q = [q_1, q_2, \dots, q_n]$
  $R = Q^T A$
      return $Q, R$
```

3.3.2 Householder Transformations

Householder reflection matrices are a common technique used to "zero" out elements within a matrix. Given a single vector, a householder matrix can be built that reflects the matrix onto a specific axis such that all entries are zero except for the leading entry. In the context of QR factorization, this can be used to zero out all of the elements in a column beneath the diagonal. Householder reflections can then be repeatedly applied to each column of a matrix A until an upper diagonal matrix is formed, which becomes R in the factorization. Because

Householder matrices are involutory, they are their own inverses, and the matrix Q is found by multiplying all transformations in reverse order. The main advantage of using Householder transformations is that their unitary nature makes them much more stable than the Gram-Schmidt process above, though the fact that each householder transformation changes every entry column in the matrix past the current iteration can make them computationally expensive.

Algorithm 2 QR with Householder Reflections

```
Require: $m \times n$ matrix $A$ with columns $\{a_1, a_2, \ldots, a_n\}$
  $R_0 = A$
  $Q_0 = I$
  for $j = 1, 2, \ldots, n-1$ do
      $H_j = H(a_j)$ form the Householder matrix corresponding to column $j$
      $R = H_j A$
      $Q = QH_j$
  end for
      return $Q, R$
```

3.3.3 Givens Rotations

Similar to Householder Reflection matrices, Givens Rotations are a stable orthogonal transform that can be used to introduce zeros within a matrix. A single given rotation can zero out a single entry underneath the diagonal of a matrix. The largest benefit to using a Givens Rotation is that it only applies to the rows of the two entries being operated on (the diagonal and sub-diagonal entries), and can therefore be exploited using parallel computation for efficient implementation. However, if this feature is not exploited, forming the upper-triangular R matrix becomes quadratic in complexity because each sub-diagonal entry needs to be operated on rather than entire columns at a time.

Algorithm 3 QR with Givens Rotations

```
Require: $m \times n$ matrix $A$ with columns $\{a_1, a_2, \ldots, a_n\}$
  $R = A$
  $Q = I$
  for $j = 1, 2, \ldots, n-1$ do
      for $i = j+1, \ldots, m$ do
          $G_{ij} = G(a_{ii}, a_{ij})$ form the Givens matrix corresponding to elements $a_{ii}$ and $a_{ij}$
          $R = G_{ij} R$
          $Q = QG_{ij}$
      end for
  end for
      return $Q, R$
```

3.3.4 Introducing Column Pivoting

It is important to note that each algorithm above targets a single column at a time. Even if more than a single column is affected by each iteration, the goal of each step is to calculate the next column to be included in the Q or R matrix. In the traditional implementations of QR factorization, the order of columns is simply decided by the ordering of columns in the original matrix A. When trying to implement column pivoting, the challenge then becomes which column to select next to reach the desired structure for the factorization described above.

The diagonal elements of the matrix R roughly correspond to the singular values of A, in that their relative ordering by size is the same. To enforce the ordering $|r_{11}| \geq |r_{22}| \geq \cdots \geq |r_{nn}|$, the columns of A must be selected in the order of magnitude of the corresponding singular value of that column. In other words, the columns should be ordered by how well-conditioned they are. However, it is extremely computationally expensive and inefficient to calculate the singular values that would result from each possible column permutation. Instead, consider the intermediate factored block matrix

$$\begin{bmatrix} R_i & B \\ 0 & C \end{bmatrix} \tag{14}$$

where R_{11} is upper triangular, well conditioned, and ordered in the manner described above. Note that the entire matrix has not yet been reduced to upper-triangular form.

The challenge then is to select the next column out of the blocks B and C that correspond to the next largest singular value of A. Instead of testing the resulting diagonal element from each column, the relative magnitudes of each singular value can be estimated using the norm of the column in block C (the elements beneath the last diagonal of block R_i) [1]. The norm of these columns does not compute the actual value of the corresponding singular values, but they can accurately represent large differences in magnitude. The consequence of this is that the resulting R matrix may not have its diagonal entries in perfectly descending order, but diagonal elements smaller than the tolerance will still be separated from the well-conditioned columns. Therefore this strategy still allows for the block form of R described above in which numerical rank can be detected, the negligible columns can be discarded, and a low-rank approximation of A can be calculated, even if the elements are not specifically ordered.

All three of the standard QR factorization algorithms can be augmented to include column pivoting. At each iteration, the next column is chosen using the strategy outlined above and then the column is processed as normal.

Besides allowing for the detection of numerical rank, permuted QR factorization also allows for early termination of the factorization algorithm. Because the diagonal elements of R are not ordered in a standard QR decomposition, the entire factorization needs to be computed to make any conclusions about the numerical rank. With permuted QR algorithms, the blocks of R and Q that are determined to be negligible will be discarded when creating a low-rank approximation, and so do not need to be computed by the algorithm. Because of

Algorithm 4 QR with column pivoting

```
Require: $m \times n$ matrix $A$ with columns $\{a_1, a_2, \dots, a_n\}$, specified QR algorithm
  $P = I$
  for $j = 1, 2, \dots, n$ do
      Find $k$ such that $||a_k[i : m]||_2$ is maximized
      Switch columns $a_i$ and $a_k$, store the permutation in $P$
      Proceed with the specified QR factorization algorithm
  end for
      return $Q, R, P$
```

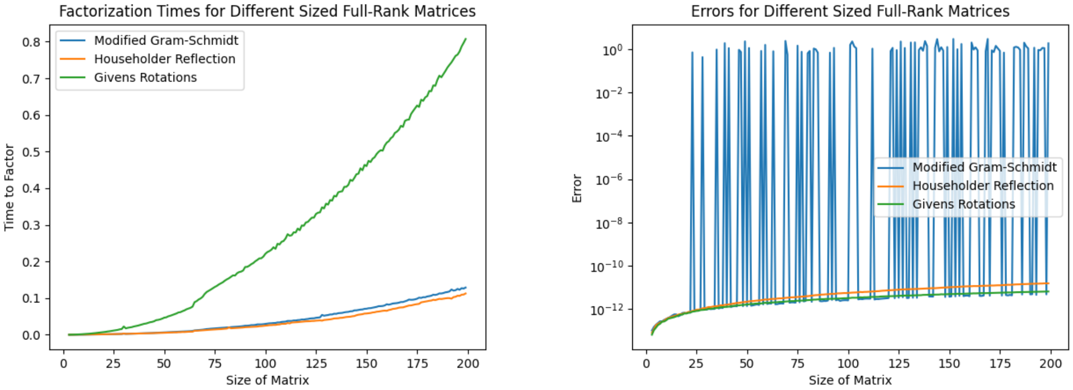

Figure 1: Factorization times and errors QR

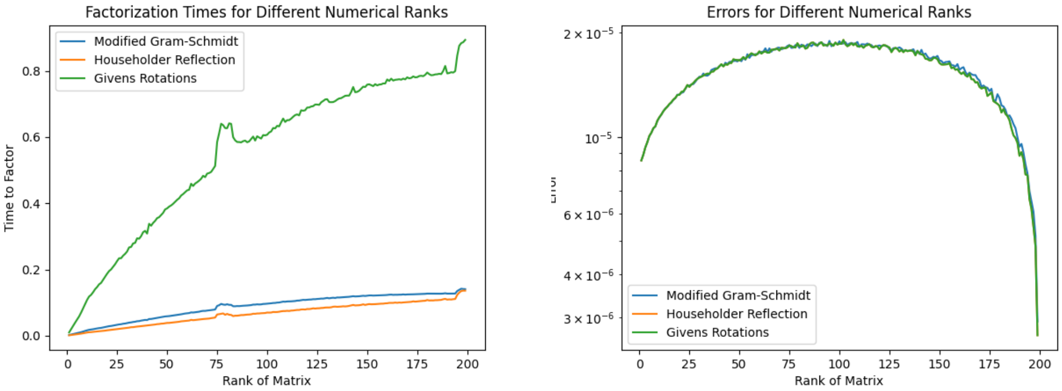

Figure 2: Factorization times and errors QR

this, a tolerance can be implemented in the algorithm such that the iteration stops if the remaining columns are all determined to be ill-conditioned and correspond to negligible singular values. This is accomplished in the same step that selects the next column for iteration.

As can be seen above, from the algorithms developed, permuted QR factorization using householder reflection transformations performed the best in time while also maintaining a high level of stability across different dimensions and ranks. For this reason, the permuted

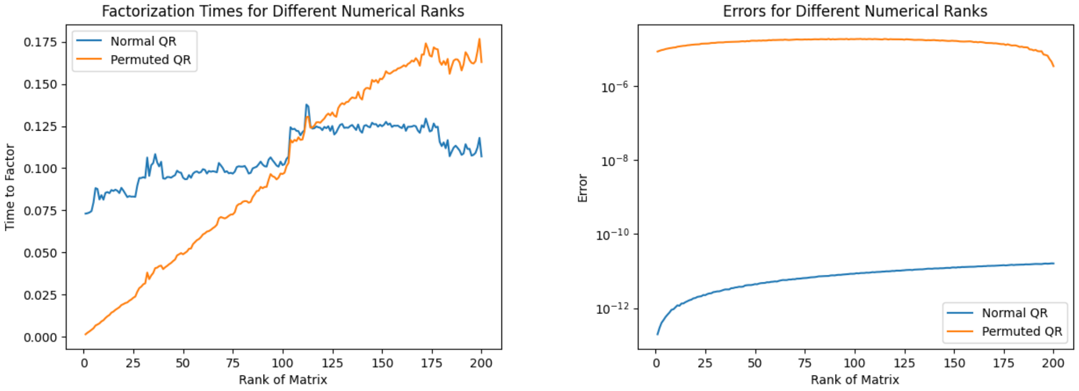

Figure 3: Factorization times and errors QR

householder QR factorization will be used for implementations in the rest of this paper.

4 Independent Extension

After developing our RRQRF, we applied the theoretical background of the RRQRF to a signal processing problem developed in a paper by Per Christian Hansen and Soren Holdt Jensen that used subspace-based noise reductions via diagonalization and triangular decompositions as the RRFs for signal enhancement, which estimates a clean signal from a noisy one [2]. As mentioned previously, we will implement their methods, but instead, we use the SVD and the QR factorization with column pivoting as our RRFs.

4.1 Mathematical Formulation

Any digital signal, s, can be represented as a column vector $s \in \mathbb{R}^n$. It can be assumed that the signal consists of a pure signal $\bar{s} \in \mathbb{R}^n$ that has been corrupted by additive noise $e \in \mathbb{R}^n$. Each of these vectors has an associated covariance matrix C, which is a square matrix where each entry represents a variance calculation, defined by

$$Cov(x_1, y_1) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \tag{15}$$

. It is also assumed that the pure signal and the noise are uncorrelated, meaning the covariance of the pure signal can be decomposed as

$$C_s = C_{\bar{s}} + C_e \tag{16}$$

Due to the random nature of noise in a signal, it can be assumed that C_e has full rank, therefore meaning that C_s has full rank as well. However, the aim of subspace-based signal processing relies on the assumption that $C_{\bar{s}}$ is not of full rank and that $\bar{s}$ lies in a proper subspace of $\mathbb{R}^n$, called the signal subspace. The main goal then is to estimate this subspace and reconstruct a good estimate of the pure signal within this subspace.

Actually computing the covariance matrix of a signal is computationally very inefficient, and can instead be approximated using a Hankel matrix. The Hankel matrix of a signal $s \in \mathbb{R}^N$ is defined as

$$\mathbb{H}(s) = \begin{bmatrix} s_1 & s_2 & s_3 & \dots & s_n \\ s_2 & s_3 & s_4 & \dots & s_{n+1} \\ s_3 & s_4 & s_5 & \dots & s_{n+2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_m & s_{m+1} & s_{m+2} & \dots & s_N \end{bmatrix} \tag{17}$$

with $N = n + m - 1$ and $m \geq n$. The associated covariance matrix can then be estimated as

$$C_s \approx \frac{1}{m} H^T H \tag{18}$$

.

4.1.1 White Noise Methods

In the case of ideal white noise, the noise covariance matrix is a scaled identity,

$$C_e = \eta^2 I \tag{19}$$

where η^2 is the variance of the noise. While noisy signals in the real world rarely follow this model, the variance of the noise can still be approximated by sampling gaps in the clean signal (such as pauses in speech when processing a vocal signal). By assuming the noise resembles the ideal case, equation 15 can be rewritten as

$$\frac{1}{m} H^T H = \frac{1}{m} \bar{H}^T \bar{H} + \eta^2 I. \tag{20}$$

By factoring this into an SVD factorization and multiplying by m, we achieve the following decomposition

$$[\bar{V}_1 \ \bar{V}_2] \begin{bmatrix} \bar{\Sigma}_1^2 + m\eta^2 I_k & 0 \\ 0 & m\eta^2 I_{n-k} \end{bmatrix} [\bar{V}_1 \ \bar{V}_2]^T. \tag{21}$$

This factorization closely resembles the RRFs discussed in the introductory section. As such, the numerical rank of the estimate of C_s is determined to be k to a tolerance $m^{1/2}\eta$. This rank can then be forced onto the original Hankel matrix to estimate the Hankel matrix of the pure signal. There are many approaches to extracting the pure signal from the estimate of its Hankel estimate, one of which is to average along its antidiagonals.

For the algorithm described to work, the signal must be a stationary sample meaning the mean and variance of the signal must be constant. For human voices this typically occurs in 30 millisecond intervals, motivating the splitting of the signal into subintervals. The above strategy can be applied to obtain estimates of the pure signal for each interval, which are then recombined to form the full pure signal estimate.

4.2 Numerical Results of Independent Extension

4.2.1 Artificial Sine Signal

Before implementing the methods from the paper with a speech signal, we implemented the algorithm with the following artificial pure sine signal

$$\bar{s}_i = sin(0.4i) + 2sin(0.9i) + 4sin(1.7i) + 3sin(2.6i) \tag{22}$$

for $i = 1, 2, \ldots, N$ where $N = n + m - 1$ and then adding Gaussian white noise, e.

After constructing the signal, $s = \bar{s} + e$, the corresponding Hankel matrix built, $\mathbb{H}(s)$. Then the corresponding covariance matrix was built, $C_s = C(\mathbb{H}(s))$, which then C_s was put through the RRQRF to find the numerical rank k.

Then impose rank k on $\mathbb{H}(s)$. This rank k Hankel matrix gives an estimate of the pure signal. Then the sum of the antidiagonal elements of the rank k Hankel matrix is the estimated pure signal.

The figure below shows frequency spectra for the pure signal, the combined noisy signal, and the corresponding estimated signal from the steps outlined above.

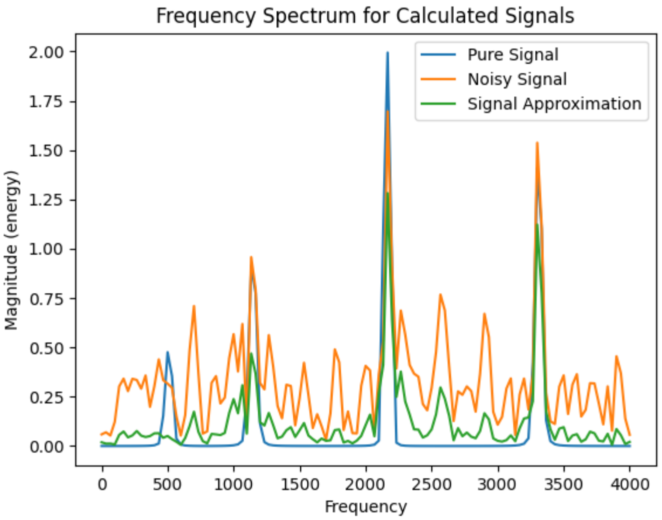

Figure 4: Extracted pure signal from sine wave

We can see that the pure signal, shown in blue, has only four base frequencies, centered at approximately 500, 1200, 2200, and 3400 Hz. Adding Gaussian white noise to the pure signal to create the noisy signal, shown in orange, adds many more "junk" frequencies to the spectrum. Our estimated signal, shown in green, retains the main four base frequencies from the pure signal, while significantly decreasing the magnitudes of the junk frequencies created by the noise.

4.2.2 Speech Signal

Now moving onto the speech signal, we have a 3-second recording of Maedée speaking and added white Gaussian noise. Before constructing the Hankel matrix and corresponding covariance matrices, we needed to estimate the variance of the Gaussian white noise added since it is used to compute the tolerance value that is used later for finding numerical rank. We then split the 3-second recording into disjoint subintervals of 30 milliseconds.

For each interval, we estimated the Hankel matrix, found the corresponding covariance matrix, then put the covariance matrix through the RRQRF to find the numerical rank k. Then we force a rank k on the Hankel matrix and extract the pure signal from the rank k approximation by summing the anti-diagonal elements.

Below are three figures showing the combined speech signal with white noise, the estimated pure signal, the residuals between the original signal and the our estimation, and the frequency spectra of the original signal and the processed signal.

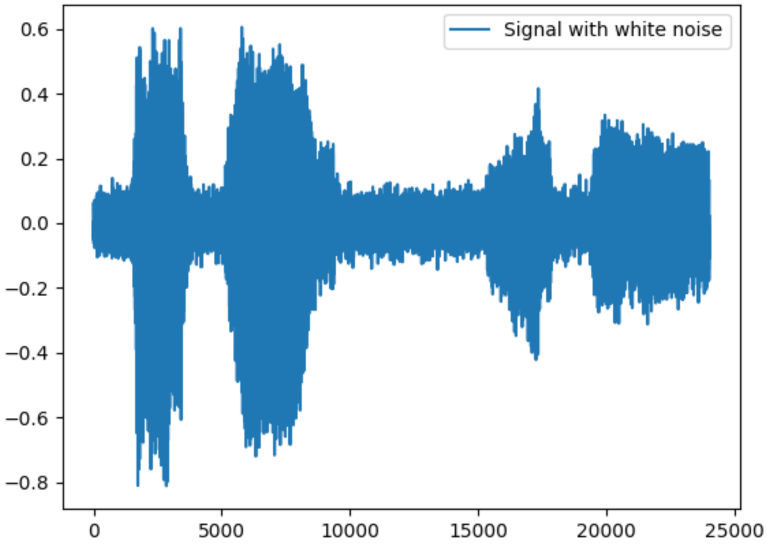

Figure 5: Combined speech signal and white noise

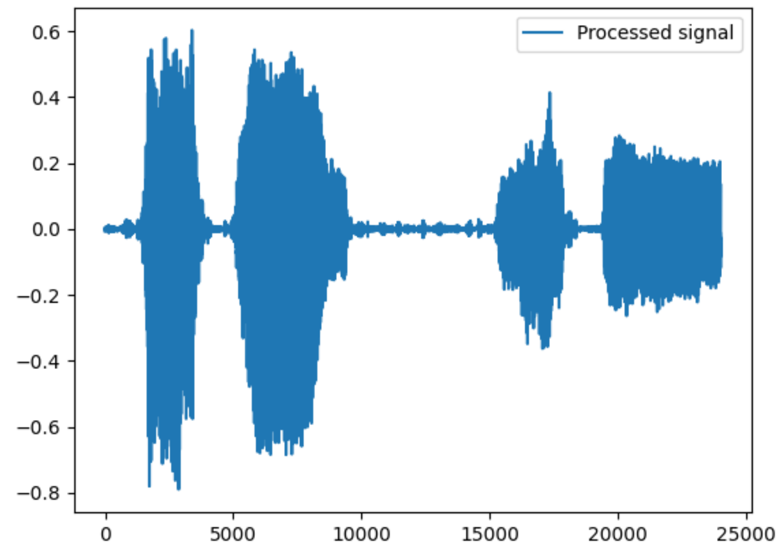

Figure 6: Processed signal

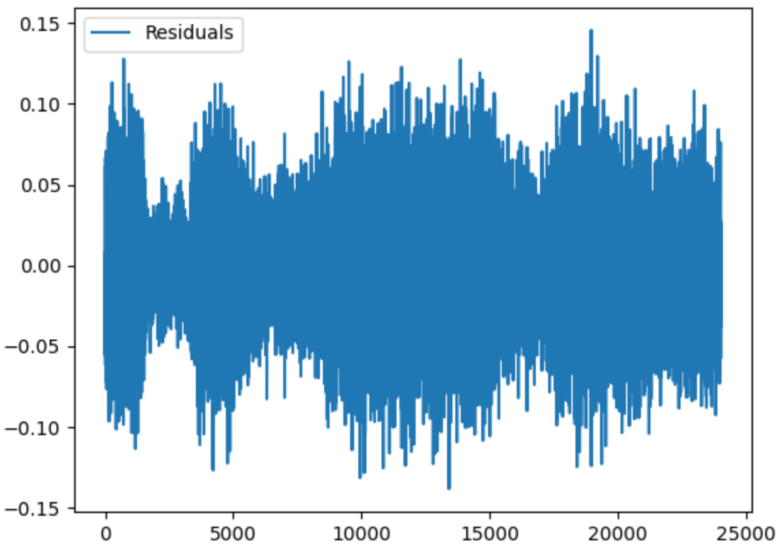

Figure 7: Error between processed signal and original clean signal

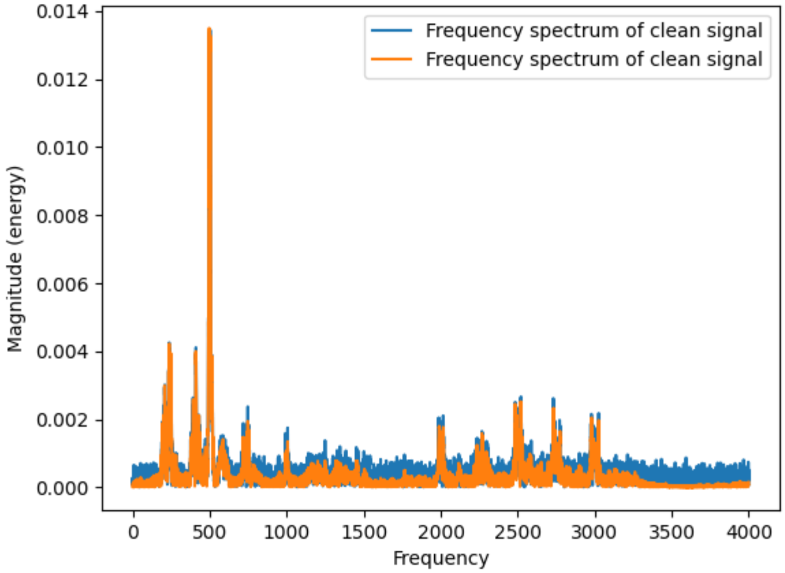

Figure 8: Frequency Spectra

5 Conclusions

The results of our permuted QR algorithms showed that using Householder reflection matrices provided the best balance between speed of factorization and stability when compared against both matrix size and matrix rank. However, using the permuted QR as the size of the rank of the matrix increases is less efficient and has a larger absolute error compared to the normal QR. Despite this inefficiency, the ordered elements of the R matrix found using permuted QR factorization allows for entire blocks of the matrix to be discarded as numerically negligible, something that would not be possible with the random ordering of traditional QR factorizations.

In addition to this, using the methods outlined in the paper by Per Christian Hansen and Søren Holdt Jensen, it was shown that rank revealing factorizations can be used to estimate the subspace that a pure signal lies in. Using this subspace, an estimation of the pure signal can be extracted by forcing a rank equal to the dimension of that subspace. While this process is not able to perfectly reconstruct the original signal, it is able to form a relatively good estimate without having any knowledge about the noise in the signal itself.

5.1 Future Work

If we were to continue this project in the future, we first continue to increase the efficiency of the QR algorithm. We would also like to explore more non-white noise methods that were mentioned in the independent extension paper, such as general noise, colored noise, or rank-deficient noise. Adding these methods should allow us to generalize the results of extracting pure signals from noisy signals using RRQRFs with many different types of noise.

References

[1] Shivkumar Chandrasekaran and Ilse C. F. Ipsen. "On Rank-Revealing Factorisations". In: *SIAM Journal on Matrix Analysis and Applications* 15.2 (1994), pp. 592–622. DOI: `10.1137/S0895479891223781`. eprint: `https://doi.org/10.1137/S0895479891223781`. URL: `https://doi.org/10.1137/S0895479891223781`.

[2] Per Christian Hansen and Søren Holdt Jensen. "Subspace-Based Noise Reduction for Speech Signals via Diagonal and Triangular Matrix Decompositions: Survey and Analysis". In: *EURASIP Journal on Advances in Signal Processing* 2007.1 (2007), p. 092953. DOI: `10.1155/2007/92953`. URL: `https://doi.org/10.1155/2007/92953`.

[3] Peter J Olver and Chehrzad Shakiban. *Applied Linear Algebra*. Undergraduate Texts in Mathematics. Springer International Publishing AG, 2018. ISBN: 978-3-319-910 41-3.

6 Appendix

For detailed code used within this project, refer to the following GitHub repository
https://github.com/zane-perry/APPM4600-Final-Project