11-21-2008

Factorization of Quasiseparable Matrices

Paul D. Johnson

FACTORIZATION OF QUASISEPARABLE MATRICES

by

Paul D. Johnson

Under the Direction of Michael Stewart

ABSTRACT

This paper investigates some of the ideas and algorithms developed for exploiting the structure of quasiseparable matrices. The case of purely scalar generators is considered initially. The process by which a quasiseparable matrix is represented as the product of matrices comprised of its generators is explained. This is done clearly in the scalar case, but may be extended to block generators. The complete factoring approach is then considered. This consists of two stages: inner-outer factorization followed by inner-coprime factorization. Finally, the stability of the algorithm is investigated. The algorithm is used to factor various quasiseparable matrices R created first using minimal generators, and subsequently using non-minimal generators. The result is that stability of the algorithm is compromised when non-minimal generators are present.

INDEX WORDS: Structured matrices, Quasiseparable matrices, QR factorization, Fast algorithms.

FACTORIZATION OF QUASISEPARABLE MATRICES

by

PAUL D. JOHNSON

A Thesis Submitted in Partial Fulfillment of Requirements for the Degree of

Master of Science

in the College of Arts and Sciences

Georgia State University

2008

FACTORIZATION OF QUASISEPARABLE MATRICES

by

PAUL D. JOHNSON

Committee Chair: Michael Stewart

Committee: Frank Hall
George Davis

Electronic Version Approved:

Office of Graduate Studies
College of Arts and Sciences
Georgia State University
December 2008

Contents

List of Tables

Chapter 1

Introduction and Definitions

This paper primarily investigates claims and algorithms presented in [2]. Although the main point of this process is to gain efficiency in multiplying, factoring, and solving systems involving quasiseparable matrices, this paper is focused on stability of such processes. The Octave programs developed herein are designed with functionality and stability in mind, with relatively little thought given to efficiency. As in the case of [2], it is assumed that the generators of the quasiseparable matrix are known.

A quasiseparable matrix R is a structured matrix in the form

$$\begin{bmatrix} d_1 & g_1h_2 & g_1b_2h_3 & g_1b_2b_3h_4 & \cdots & g_1b_2\cdots b_{N-1}h_N \\ p_2q_1 & d_2 & g_2h_3 & g_2b_3h_4 & \cdots & g_2b_3\cdots b_{N-1}h_N \\ p_3a_2q_1 & p_3q_2 & d_3 & g_3h_4 & \cdots & g_3b_4\cdots b_{N-1}h_N \\ p_4a_3a_2q_1 & p_4a_3q_2 & p_4q_3 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & d_{N-1} & g_{N-1}h_N \\ p_Na_{N-1}\cdots a_2q_1 & p_Na_{N-1}\cdots a_3q_2 & p_Na_{N-1}\cdots a_4q_3 & \cdots & p_Nq_{N-1} & d_N \end{bmatrix} \tag{1.1}$$

or, expressed in a more compact form,

$$R_{ij} = \begin{cases} p_i a_{i-1} \cdots a_{j+1} q_j, & 1 \le j < i \le N, \\ d_i, & 1 \le i = j \le N, \\ g_i b_{i+1} \cdots b_{j-1} h_j, & 1 \le i < j \le N \end{cases} . \tag{1.2}$$

(Note that on the subdiagonal, $j = i - 1$, allowing for no a_k, and similarly on the superdiagonal, $j = i + 1$, allowing for no b_k.) The generators, d_k, p_i, q_j, a_k, g_i, h_j, and b_k may be scalar or matrix quantities. Matrix generators are commonly referred to as block generators. For block generators, the dimensions are defined in the following table. (This is taken directly from [2].)

Generator	Dimensions
d_k	$m_k \times n_k$
p_i	$m_i \times r'_{i-1}$
q_j	$r'_j \times n_j$
a_k	$r'_k \times r'_{k-1}$
g_i	$m_i \times r''_i$
h_j	$r''_{j-1} \times n_j$
b_k	$r''_{k-1} \times r''_k$

1. Rank of Submatrices

   Any submatrix entirely in the strict lower triangle, or equivalently, in the strict upper triangle, is of rank equal to the largest size ($\max(r'_k)$ in the lower triangle, and $\max(r''_k)$ in the upper triangle) of any generator in that submatrix. The diagram below illustrates submatrices of each type. As a side note, the rank of such submatrices is referred to as the "Hankel rank" in [1].

$$
\begin{bmatrix}
d_1 & & * & \cdots & * & * & * & * \\
& d_2 & * & \cdots & * & * & * & * \\
& & d_3 & & & & & \\
& & & \ddots & & & & \\
& & & & d_{N-3} & & & \\
* & * & * & \cdots & * & d_{N-2} & & \\
* & * & * & \cdots & * & & d_{N-1} & \\
* & * & * & \cdots & * & & & d_N
\end{bmatrix}
$$

2. Upper and Lower Rank

   The "upper rank" of a quasiseparable matrix R is the largest rank of any submatrix in the strict upper triangle of R, i.e. upper rank = $\max(r''_k, k = 2, \ldots, N-1)$, and the "lower rank" of R is the largest rank of any submatrix in the strict lower triangle of R, i.e. lower rank = $\max(r'_k, k = 2, \ldots, N-1)$.

3. The Scalar Case

   In this paper, "the scalar case" refers to a quasiseparable matrix in which all generators $p_i, a_k, q_j, d_k, g_i, b_k, h_j$ are scalar, i.e. in which $m_k = n_k = 1, k = 1, ..., N$ and $r'_k = r''_k = 1, k = 1, ..., N-1$.

4. Subclasses

   The class of quasiseparable matrices includes several other well-known matrices, such as band matrices, diagonal plus semiseparable matrices, tridiagonal matrices, and unitary Hessenberg matrices [2].

5. Benefits

The structure of a quasiseparable matrix may be exploited in order to reduce the number of calculations required in performing certain operations. Specifically:

(a) Multiplying a quasiseparable matrix R by a vector $\mathbf{x}$ may be performed in $O(N)$ operations, as opposed to $O(N^2)$ operations for the multiplication of a non-structured matrix M by a vector $\mathbf{x}$. (The details are provided in [1] and reproduced in the included programs quasifactor.m, lowermult.m, dmult.m, and uppermult.m. The idea is to separate the quasiseparable matrix into three parts: the strict lower triangle, the diagonal, and the strict upper triangle; multiply each by the given vector; and then add the three products to find the product $R\mathbf{x}$.)

(b) Solving a system $R\mathbf{x} = \mathbf{y}$ may be accomplished with fewer computations (solution is $(O(N))$ by first efficiently factoring R into unitary factors to the greatest extent possible. This is the main focus of this paper.

6. Non-Minimal Generators

The idea of minimal generators is that the information represented in the generators, and more specifically in their product, cannot be represented by generators of smaller size. To illustrate, as simply as possible, the idea of non-minimal generators, consider the following example.

$$p = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad a = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}, \qquad q = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

such that

$$paq = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \end{bmatrix},$$

which could easily have been represented by scalar generators. Using generators of larger size represented more information than was available in the final product. (This example was transcribed directly from notes provided by Dr. Michael Stewart in his explanation of the concept of minimality in [5].)

The idea of near non-minimality describes generators that are numerically very close to a non-minimal state (typically where one or a few elements have been very slightly perturbed from a non-minimal state.)

This thesis is developed in the following steps. Chapter 2 initially considers a representation of the lower triangle of a quasiseparable matrix, which can be thought of as a portion of a lower Hessenberg matrix of larger dimension. This representation, developed in Section 2.2, demonstrates a simple way to multiply the generators of (1.2) to create the lower triangle of (1.1). This is a particularly explicit way of seeing how the factoring/decomposition of the quasiseparable matrix R may be performed. The initial development is intended to be as transparent as possible, so it is performed with the idea that all generators are scalar. Before things become too complicated, factoring the scalar case of R is considered in Section 2.4. Many of the important ideas about the construction and factoring of a quasiseparable matrix should be clear, allowing easier transition to the general (block) case of R.

The complete factoring approach is then considered in Chapter 3. This consists of two steps. First is inner coprime factorization, in which R is decomposed into $R = VT$, where V is a block lower triangular unitary matrix and T is a block upper triangular matrix, each quasiseparable. Next is inner-outer factorization, in which T is decomposed into $T = US$, where U is a block upper triangular unitary matrix and S is a block upper triangular matrix with square invertible blocks on the main diagonal [2, p. 429]. Both steps rely heavily on QR factorization applied to two block rows of R at a time. The unitary factor Q from each step is separated into blocks that act as generators of one of the new quasiseparable factors: V in the inner coprime factorization and U in the inner-outer factorization. This chapter is simply an explicit description of the algorithm described in [2].

Chapter 4 considers the results of the algorithm as applied to various quasiseparable matrices R created first using minimal generators, and subsequently using non-minimal and nearly non-minimal generators. The result is that stability of the algorithm is compromised when non-minimal or nearly non-minimal generators are present.

Chapter 2

Factoring: The Scalar Case

2.1 Applying Givens Rotations

Current ideas for exploiting the structure of quasiseparable matrices tend to use related factoring approaches. The principles are based primarily on QR factorization applied consecutively to pairs of block rows of R in a way that produces unitary factors to the greatest extent possible.

To begin, consider the scalar case: a quasiseparable matrix with lower order one and upper order one; that is to say, one in which all generators are scalar. A powerful way to factor this matrix is by applying plane rotations row by row, from the bottom to the top of the matrix. Because each submatrix below the diagonal is of rank one, every element in one row $(r_{ij}, 1 \leq j < i-1)$, is a constant multiple of the row directly above it $(r_{i-1,j}, 1 \leq j < i-1)$. This allows a single sweep of a plane rotation to zero out all elements $[R_{i,j}], j < i-1$ in a given row, $i = N, \ldots, 3$. Thus, all entries below the subdiagonal may be eliminated by such rotations. Whereas a true plane rotation is expressed as a unitary matrix of the form

$$\begin{bmatrix} f_i & -\bar{e}_i \\ e_i & f_i \end{bmatrix},$$

this paper generalizes and modifies such rotations to allow work in $\mathbb{C}^2$ and according to a convention common in the context of unitary Hessenberg matrices which pre-multiplies the plane rotation by the permutation matrix

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

which is equivalent to following the rotation by a reflection about $y = x$ in the Euclidean plane, $\mathbb{R}^2$. These modified plane rotations take the form

$$U_i^* = I_{i-2} \oplus \begin{bmatrix} e_i & f_i \\ f_i & -\bar{e}_i \end{bmatrix} \oplus I_{N-i}, \qquad \sqrt{|e_i|^2 + |f_i|^2} = 1, \qquad 1 < i \leq N$$

where U_i^* acts on the $(i-1)$ and i rows of R, and U_i^* is unitary. (The selection of e_i and f_i will be derived so as to zero out elements of R.) Note that f_i and $\bar{f}_i$ would be used in a more general representation of a plane rotation. Here, it is arbitrarily decided to let $f_i = \bar{f}_i$ (this goes back at least as far as [4]), resulting in a pure real value for f_i.

The selection of e_i and f_i is based on the goal of zeroing out elements in the i row of R. Because

$$\begin{bmatrix} e_i & f_i \\ f_i & -\bar{e}_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} m \\ 0 \end{bmatrix} \tag{2.1}$$

implies

$$\begin{bmatrix} \bar{f}_i \\ -e_i \end{bmatrix} \perp \begin{bmatrix} x \\ y \end{bmatrix}$$

(because $\begin{bmatrix} f_i & -\bar{e}_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$) and

$$\begin{bmatrix} \bar{f}_i \\ -e_i \end{bmatrix} \perp \begin{bmatrix} \bar{e}_i \\ \bar{f}_i \end{bmatrix}$$

(by the orthogonality of the columns of the modified rotation matrix), then

$$\begin{bmatrix} \bar{e}_i \\ \bar{f}_i \end{bmatrix} = \alpha \begin{bmatrix} x \\ y \end{bmatrix}$$

which implies

$$e_i = \frac{\bar{x}y}{|y|\sqrt{|x|^2 + |y|^2}} \qquad \text{and} \qquad f_i = \frac{|y|}{\sqrt{|x|^2 + |y|^2}}. \tag{2.2}$$

The nature of the x and y used to determine e and f will be investigated following the discussion of the product of modified plane rotations, and defined precisely in Algorithm 1. Assuming that e_i and f_i may be determined in U_i^* for $i = N, N-1, \ldots, 4, 3$, each modified rotation may be applied from the bottom (U_N^*) to the top (U_3^*) to zero out all elements in R below the subdiagonal. The result of this iterative process is $U_3^* U_4^* \cdots U_{N-1}^* U_N^* R = H$ where H is upper Hessenberg. Hence, $R = QH$, where $Q = U_N U_{N-1} \cdots U_3$ and is thus unitary. In fact, the product of these modified plane rotations is lower Hessenberg, so Q is unitary lower Hessenberg.

Before investigating how to take a quasiseparable matrix apart, it is instructive to consider one way to put one together. The process of factoring will resume in Section 2.3.

2.2 Constructing a Hessenberg Matrix

This section demonstrates a method for constucting a lower Hessenberg matrix of dimensions $(N+1) \times (N+1)$ that can be modified to create the lower part of a quasiseparable matrix, as will be shown in Section 2.3. The quasiseparable structure created in Section 2.3 is part of the Hessenberg matrix created in this section. These two sections demonstrate a way of understanding the explicit parameterization of R in (1.1) as a product of block 2×2 matrices. This generalizes a well-known representation of unitary Hessenberg matrices.

Without regard to the (unitary) structure of the modified plane rotations described above, consider the product of matrices $L = L_N \cdots L_2 L_1$ where

$$L_k = I_{k-1} \oplus \begin{bmatrix} p_k & d_k \\ a_k & q_k \end{bmatrix} \oplus I_{N-k}, \quad 1 \leq k \leq N. \tag{2.3}$$

Multiplying from right to left, partition each factor such that its diagonal blocks are square, i.e. such that the column partition is identical to the row partition. When multiplying L_k by the previously computed product $L_{k-1} \cdots L_1$, partition each of the factors such that its diagonal blocks are of size $(k-1) \times (k-1)$, 1×1, and $(N-k+1) \times (N-k+1)$. So

$$L_k = \left[\begin{array}{c|c|cc} I_{k-1} & 0 & 0 & 0 \\ \hline 0 & p_k & d_k & 0 \\ \hline 0 & a_k & q_k & 0 \\ 0 & 0 & 0 & I_{N-k} \end{array}\right].$$

Now it is easy to see that the effect of the latest factor (L_k) on the previous product

$$
L_{k-1}\cdots L_1 \\
= \left[\begin{array}{ccccccc}
p_1 & d_1 & & & & & \\
p_2a_1 & p_2q_1 & d_2 & & & 0 & \\
p_3a_2a_1 & p_3a_2q_1 & p_3q_2 \quad d_3 & & & & \\
\vdots & \vdots & \ddots & \ddots & & & \\
p_{k-2}a_{k-3}\cdots a_1 & p_{k-2}a_{k-3}\cdots a_2q_1 & \cdots & p_{k-2}q_{k-3} & d_{k-2} & & \\
p_{k-1}a_{k-2}\cdots a_1 & p_{k-1}a_{k-2}\cdots a_2q_1 & \cdots & p_{k-1}a_{k-2}q_{k-2} & p_{k-1}q_{k-2} & d_{k-1} & \\
\hline
a_{k-1}\cdots a_1 & a_{k-1}\cdots a_2q_1 & \cdots & a_{k-1}a_{k-2}q_{k-3} & a_{k-1}q_{k-2} & q_{k-1} & 0 \\
\hline
0 & & \cdots & & & 0 & I_{N-k+1}
\end{array}\right]
$$

is the following:

1. Retain the first $(k-1)$ rows,
2. Create a new k row by multiplying p_k by each element in the k row and appending d_k in the $(k, k+1)$ position.
3. Create a new $(k+1)$ row by multiplying a_k by each element in the k row and appending q_k in the $(k+1, k+2)$ position.
4. Retain the last $(N-k)$ rows.

So $L_k \cdots L_1 =$

$$\left[\begin{array}{ccccccc}
p_1 & d_1 & & & & & \\
p_2 a_1 & p_2 q_1 & d_2 & & & 0 & \\
p_3 a_2 a_1 & p_3 a_2 q_1 & p_3 q_2 \quad d_3 & & & & \\
\vdots & \vdots & \ddots & \ddots & & & \\
p_{k-1} a_{k-2} \cdots a_1 & p_{k-1} a_{k-2} \cdots a_2 q_1 & \cdots & p_{k-1} q_{k-2} & d_{k-1} & & \\
\hline
p_k a_{k-1} \cdots a_1 & p_k a_{k-1} \cdots a_2 q_1 & \cdots & p_k a_{k-1} q_{k-2} & p_k q_{k-1} & d_k & \\
\hline
a_k \cdots a_1 & a_k \cdots a_2 q_1 & \cdots & a_k a_{k-1} q_{k-2} & a_k q_{k-1} & q_k & 0 \\
0 & & \cdots & & & 0 & I_{N-k}
\end{array}\right]$$

for $k = 2, ..., N$. Hence

$$\begin{aligned}
L &= L_N \cdots L_1 \\
&= \begin{bmatrix}
p_1 & d_1 & & & & \\
p_2 a_1 & p_2 q_1 & d_2 & & & \\
p_3 a_2 a_1 & p_3 a_2 q_1 & p_3 q_2 & d_3 & & \\
\vdots & \vdots & \ddots & \ddots & \ddots & \\
p_N a_{N-1} \cdots a_1 & p_N a_{N-1} \cdots a_2 q_1 & \cdots & p_N a_{N-1} q_{N-2} & p_N q_{N-1} & d_N \\
a_N \cdots a_1 & a_N \cdots a_2 q_1 & \cdots & a_N a_{N-1} q_{N-2} & a_N q_{N-1} & q_N
\end{bmatrix}
\end{aligned}$$

and is clearly lower Hessenberg. (If the generators, p, a, q, d are matrices, L is block lower Hessenberg.)

2.3 Converting from Hessenberg to Lower Quasiseparable Structure

To create the lower part of a quasiseparable matrix, it is only necessary to delete the first column and last row of the lower Hessenberg matrix obtained in the last section, producing an $N \times N$ matrix. This can be accomplished through the use of an L_0 and L_{N+1} to create a modified L_1 and L_N. Simply let

$$\begin{aligned} L &= L_{N+1} L_N L_{N-1} \cdots L_2 L_1 L_0 \\ &= \tilde{L}_N L_{N-1} \cdots L_2 \tilde{L}_1 \end{aligned} \tag{2.4}$$

where $L_0 = \left[\begin{array}{c} 0 \\ \hline I_N \end{array}\right]$ and $L_{N+1} = \left[\begin{array}{c|c} I_N & 0 \end{array}\right]$ such that

$$\tilde{L}_1 = L_1 L_0 = \left[\begin{array}{c|cc} p_1 & d_1 & 0 \\ a_1 & q_1 & 0 \\ 0 & 0 & I_{N-1} \end{array}\right] \left[\begin{array}{c} 0 \\ \hline I_N \end{array}\right] = \left[\begin{array}{c} d_1 \\ q_1 \end{array}\right] \oplus I_{N-1}$$

and

$$\tilde{L}_N = L_{N+1} L_N = \left[\begin{array}{c|c} I_N & 0 \end{array}\right] \left[\begin{array}{ccc} I_{N-1} & 0 & 0 \\ 0 & p_N & d_N \\ \hline 0 & a_N & q_N \end{array}\right] = I_{N-1} \oplus \left[\begin{array}{cc} p_N & d_N \end{array}\right].$$

This shaves the first column and last row off of the lower Hessenberg structure, leaving behind precisely the quasiseparable structure of interest:

$$L = R = \begin{bmatrix} d_1 & & & & \\ p_2 q_1 & d_2 & & & \\ p_3 a_2 q_1 & p_3 q_2 & d_3 & & \\ \vdots & \ddots & \ddots & \ddots & \\ p_N a_{N-1} \cdots a_2 q_1 & \cdots & p_N a_{N-1} q_{N-2} & p_N q_{N-1} & d_N \end{bmatrix}. \tag{2.5}$$

In this instance, all upper generators are effectively zero.

One drawback to this technique is the lack of invertibility of $\tilde{L}_1$ and $\tilde{L}_N$ if d_k, p_k, and q_k are scalar. However, in the case of block generators, there are cases in which $\tilde{L}_1$ and $\tilde{L}_N$ may be invertible. Particularly, they may each be unitary, by design, as will be shown in Section 3.1. Their selection will be based on the Q (the unitary factor) from QR factorization of specific generators of R.

It should be clear that the same technique may be used to construct the upper part of a quasiseparable matrix. For this, let

$$W = \tilde{W}_1 W_2 \cdots W_{N-1} \tilde{W}_N$$

where

$$\tilde{W}_1 = \left[\begin{array}{c|c} 0 & I_N \end{array}\right] \left[\begin{array}{ccc} h_1 & b_1 & 0 \\ \hline d_1 & g_1 & 0 \\ 0 & 0 & I_{N-1} \end{array}\right] = \begin{bmatrix} d_1 & g_1 \end{bmatrix} \oplus I_{N-1}$$

$$W_k = I_{k-1} \oplus \begin{bmatrix} h_k & b_k \\ d_k & g_k \end{bmatrix} \oplus I_{N-k}, \quad 2 \le k \le N-1, \text{ and}$$

$$\tilde{W}_N = \left[\begin{array}{cc|c} I_{N-1} & 0 & 0 \\ 0 & h_N & b_N \\ 0 & d_N & g_N \end{array}\right] \left[\begin{array}{c} I_N \\ \hline 0 \end{array}\right] = I_{N-1} \oplus \begin{bmatrix} h_N \\ d_N \end{bmatrix}.$$

2.4 Factoring

Returning to the idea of using modified plane rotations to factor the quasiseparable matrix R, whose lower triangle is defined in (1.1) and created equivalently as L by the generators leading to (2.5), R may be factored into $R = QH$, where Q is unitary and H is upper Hessenberg. Consider a submatrix taken from two rows in the strict lower triangle of R:

$$\begin{bmatrix} r_{i-1,j} \\ r_{i,j} \end{bmatrix} \text{ for } 3 \le i \le N, 1 \le j \le i-2.$$

$$\begin{aligned} \begin{bmatrix} r_{i-1,j} \\ r_{i,j} \end{bmatrix} &= \begin{bmatrix} p_{i-1}a_{i-2}\cdots a_2q_1 & p_{i-1}a_{i-2}\cdots a_3q_2 & \cdots & p_{i-1}a_{i-2}q_{i-3} & p_{i-1}q_{i-2} \\ p_ia_{i-1}a_{i-2}\cdots a_2q_1 & p_ia_{i-1}a_{i-2}\cdots a_3q_2 & \cdots & p_ia_{i-1}a_{i-2}q_{i-3} & p_ia_{i-1}q_{i-2} \end{bmatrix} \\ &= \begin{bmatrix} p_{i-1} \\ p_ia_{i-1} \end{bmatrix} \begin{bmatrix} a_{i-2}\cdots a_2q_1 & a_{i-2}\cdots a_3q_2 & \cdots & a_{i-2}q_{i-3} & q_{i-2} \end{bmatrix}. \end{aligned}$$

Now it is clear that the whole lower row in this submatrix may be zeroed out if a modified rotation that turns $p_i a_{i-1}$ into 0 is applied [3]. This can easily be accomplished through careful selection of e_i and f_i discussed in Section 2.1.

Beginning with rows $N-1$ and N, select e_N and f_N based on (2.1) and (2.2) by setting

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} p_{N-1} \\ p_N a_{N-1} \end{bmatrix}, \tag{2.6}$$

so that

$$\begin{bmatrix} e_N & f_N \\ f_N & -\bar{e}_N \end{bmatrix} \begin{bmatrix} p_{N-1} \\ p_N a_{N-1} \end{bmatrix} = \begin{bmatrix} m_{N-1} \\ 0 \end{bmatrix}.$$

Then

$$U_N^* R =$$

$$= \left[\begin{array}{c|cc} I_{N-2} & 0 & 0 \\ \hline 0 & e_N & f_N \\ 0 & f_N & -\bar{e}_N \end{array}\right] \left[\begin{array}{ccccc} d_1 & & & & \\ p_2 q_1 & d_2 & & * & \\ p_3 a_2 q_1 & p_3 q_2 & d_3 & & \\ \vdots & & \ddots & \ddots & \\ \hline p_{N-1} a_{N-2} \cdots a_2 q_1 & \cdots & p_{N-1} q_{N-2} & d_{N-1} & * \\ p_N a_{N-1} a_{N-2} \cdots a_2 q_1 & \cdots & p_N a_{N-1} q_{N-2} & p_N q_{N-1} & d_N \end{array}\right]$$

$$= \left[\begin{array}{cccccc} d_1 & & & & & \\ p_2 q_1 & d_2 & & & * & \\ p_3 a_2 q_1 & p_3 q_2 & d_3 & & & \\ \vdots & & \ddots & \ddots & & \\ \hline m_{N-1} a_{N-2} \cdots a_2 q_1 & \cdots & & m_{N-1} q_{N-2} & * & * \\ 0 & \cdots & 0 & 0 & f_N d_{N-1} - \bar{e}_N p_N q_{N-1} & * \end{array}\right]$$

where

$$m_{N-1} = e_N p_{N-1} + f_N p_N a_{N-1} = z = \sqrt{|p_{N-1}|^2 + |p_N a_{N-1}|^2}.$$

Note that in addition to zeroing out elements in the lower triangle, this modified rotation affects a diagonal element and introduces a nonzero element in the upper triangle. (The nature of the new elements in the upper triangle is not extremely important here, but will be investigated in detail in the general factoring case in Section 3.1.)

This process can be repeated iteratively. In the next step, select e_{N-1} and f_{N-1} by setting

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} p_{N-2} \\ m_{N-1} a_{N-2} \end{bmatrix}.$$

Then

$$
\begin{aligned}
U_{N-1}^*(U_N^*R) &= \\
&= \left[\begin{array}{c|cc|c}
I_{N-3} & 0 & 0 & 0 \\
\hline
0 & e_{N-1} & f_{N-1} & 0 \\
0 & f_{N-1} & -\bar{e}_{N-1} & 0 \\
\hline
0 & 0 & 0 & I_1
\end{array}\right] \cdot \\
&\quad \left[\begin{array}{cccccc}
d_1 & & & & & \\
p_2q_1 & & d_2 & & * & \\
\vdots & & & \ddots & & \\
\hline
p_{N-2}a_{N-3}\cdots a_2q_1 & \cdots & p_{N-2}q_{N-3} & d_{N-2} & * & * \\
m_{N-1}a_{N-2}a_{N-3}\cdots a_2q_1 & \cdots & m_{N-1}a_{N-2}q_{N-3} & m_{N-1}q_{N-2} & * & * \\
\hline
0 & \cdots & 0 & 0 & * & *
\end{array}\right] \\
&= \left[\begin{array}{cccccc}
d_1 & & & & & \\
p_2q_1 & & d_2 & & * & \\
\vdots & & & \ddots & & \\
\hline
m_{N-2}a_{N-3}\cdots a_2q_1 & \cdots & m_{N-2}q_{N-3} & * & * & * \\
0 & \cdots & 0 & * & * & * \\
\hline
0 & \cdots & 0 & 0 & * & *
\end{array}\right]
\end{aligned}
$$

where

$$m_{N-2} = e_{N-1}p_{N-2} + f_{N-1}m_{N-1}a_{N-2}.$$

This process may be continued iteratively. In each successive step, for $i = N-1, \ldots, 3$, select e_i and f_i by setting

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} p_{i-1} \\ m_i a_{i-1} \end{bmatrix} \tag{2.7}$$

with the details provided in the next algorithm.

Algorithm 1: Let $R = [R_{ij}]_{i,j=1}^N = \begin{cases} p_i a_{i-1} \cdots a_{j+1} q_j, & \text{for } 1 \leq j < i \leq N, \\ d_i, & 1 \leq i = j \leq N, \\ g_i b_{i+1} \cdots b_{j-1} h_j, & 1 \leq j < i \leq N \end{cases}$
with all scalar generators: $p_k, a_k, q_k, d_k, g_k, b_k, h_k \in \mathbb{C}$.

Then R admits factoring $R = QH$ where Q is unitary lower Hessenberg and H is upper Hessenberg as follows.

1. Let $m_N = p_N$.

2. Compute recursively for $i = N, \dots, 3$:

$$\begin{aligned}
e_i &= \tfrac{\bar{p}_{i-1} m_i a_{i-1}}{|m_i a_{i-1}| \sqrt{|p_{i-1}|^2 + |m_i a_{i-1}|^2}}, \\
f_i &= \tfrac{|m_i a_{i-1}|}{\sqrt{|x|^2 + |y|^2}}, \\
U_i^* &= I_{i-2} \oplus \begin{bmatrix} e_i & f_i \\ f_i & -\bar{e}_i \end{bmatrix} \oplus I_{N-i}, \\
U_i &= I_{i-2} \oplus \begin{bmatrix} \bar{e}_i & \bar{f}_i \\ \bar{f}_i & -e_i \end{bmatrix} \oplus I_{N-i}, \text{ and} \\
m_{i-1} &= e_i p_{i-1} + f_i m_i a_{i-1}.
\end{aligned}$$

3. Then compute the products

$$Q = U_N U_{N-1} \cdots U_3 \qquad \text{and} \qquad H = U_3^* \cdots U_{N-1}^* U_N^* R.$$

Essentially, this approach allows the upper or lower triangle of a quasiseparable matrix to be treated using the techniques developed for factoring a unitary Hessenberg matrix. We now have defined recurrences for Q as a product of modified rotations but have not described the structure of H. The details of the effect of a sweep of modified rotations on the upper quasiseparable structure have been ignored thus far, but will be quantified precisely in Chapter 3. To broaden the benefits of this more detailed investigation, it is worth expanding the definition of generators from purely scalar to block generators of rank greater than one.

Chapter 3

Factoring: The General Case

In moving from the case of a quasiseparable matrix created from scalar generators to one created from matrix generators, similar ideas may be applied, but with some modification. The goal is to describe the algorithm of [2] for the QR factorization of a general quasiseparable matrix.

First, the quasiseparable matrix is factored into $R = VT$, where V is a block lower triangular unitary matrix, and T is a block upper triangular matrix [2, p. 429]. This is achieved through QR factorization, introducing zeros from the bottom to the top of R, exploiting the quasiseparable structure in a way analogous to the application of modified plane rotations applied in the case of scalar generators. From the nomenclature of [1], this stage is referred to as inner coprime factorization. One detail that is not immediately obvious from this factorization is that the diagonal blocks of T are not necessarily square. This creates problems for solving systems via back substitution, and is thus considered undesirable.

Hence, the matrix T is factored into $T = US$, where U is a block upper triangular unitary matrix and S is a block upper triangular matrix with square invertible blocks on the main diagonal [2, p. 429]. From [1], this stage is referred to as inner-outer factorization. The main objective of this stage is the creation of new generators that are easily inverted. Specifically, this factoring step causes the diagonal blocks of S to be square, gaining significant advantages over the non-square diagonal blocks of T.

The result of these two stages is the block QR factorization $R = VUS$.

3.1 Inner Coprime Factorization

Similar to the application of a modified rotation to two rows of a quasiseparable matrix formed from scalar generators, such that each e_i and f_i is selected to create U_i^* by applying (2.1) and (2.2) to the more general case of (2.7), QR factorization may be applied to

$$\begin{bmatrix} p_{i-1} \\ X_i a_{i-1} \end{bmatrix}$$

in such a way as to zero out much, or all, of a block row in R. As in the scalar case, this is performed from the bottom to the top of R.

The objective is to factor $R = VT$, where V is a block lower triangular unitary matrix, and T is a block upper triangular matrix. This is analogous to the factoring in the scalar case: $R = QH$. Here, V is the product of unitary matrices $\tilde{V}_N V_{N-1} \cdots V_2 \tilde{V}_1$, much like $Q = L = \tilde{L}_N L_{N-1} \cdots L_2 \tilde{L}_1$ in the scalar case. Now there is an opportunity to make $\tilde{V}_N$ and $\tilde{V}_1$ unitary, unlike in the scalar case, where

$$\begin{bmatrix} p_N & d_N \end{bmatrix} \text{ and } \begin{bmatrix} d_1 \\ q_1 \end{bmatrix}$$

are not invertible, much less unitary, allowing V to be block lower triangular and unitary, rather than lower Hessenberg and unitary.

This process will be performed from the bottom to the top of R. In the first step, let $\rho_{N-1} = \min(m_N, r'_{N-1})$, $\nu_N = m_N - \rho_{N-1}$, and then perform QR factorization on p_N, which will be used to zero out parts of the last block row if p_N is rank deficient. Let

$$p_N = Q_N \begin{bmatrix} X_N \\ 0 \end{bmatrix} = \begin{bmatrix} (p_V)_N & (d_V)_N \end{bmatrix} \begin{bmatrix} X_N \\ 0 \end{bmatrix},$$

where

$$Q_N \qquad \text{and} \qquad \begin{bmatrix} X_N \\ 0 \end{bmatrix}$$

are the unitary and upper triangular factors, respectively, with dimensions:

$$\begin{aligned} (p_V)_N: &\quad m_N \times \rho_{N-1} \\ (d_V)_N: &\quad m_N \times \nu_N \\ X_N: &\quad \rho_{N-1} \times r'_{N-1}. \end{aligned}$$

Then let $\tilde{V}_N = I_{\eta_N} \oplus Q_N$ where

$$\eta_N = \sum_{k=1}^{N-1} m_k.$$

Multiply $\tilde{V}_N^*$ by R to see its effect on the last block row of R:

$$
\begin{aligned}
\tilde{V}_N^* R &= \left[\begin{array}{c|c} I_{\eta_N} & 0 \\ 0 & (p_V)_N^* \\ 0 & (d_V)_N^* \end{array}\right] \left[\begin{array}{ccccc} \multicolumn{5}{c}{R(1:N-1,:)} \\ \hline p_N a_{N-1}\cdots a_2 q_1 & \cdots & p_N a_{N-1} q_{N-2} & p_N q_{N-1} & d_N \end{array}\right] \\
&= \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-1,:)} \\ \hline Q_N^* p_N \begin{pmatrix} a_{N-1}\cdots a_2 q_1 & \cdots & a_{N-1} q_{N-2} & q_{N-1} \end{pmatrix} & Q_N^* d_N \end{array}\right] \\
&= \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-1,:)} \\ \hline \begin{bmatrix} X_N \\ 0 \end{bmatrix} \begin{pmatrix} a_{N-1}\cdots a_2 q_1 & \cdots & a_{N-1} q_{N-2} & q_{N-1} \end{pmatrix} & d_N \end{array}\right] \\
&= \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-1,:)} \\ \hline X_N \begin{pmatrix} a_{N-1}\cdots a_2 q_1 & \cdots & a_{N-1} q_{N-2} & q_{N-1} \end{pmatrix} & (p_V)_N^* d_N \\ 0 \quad \cdots \quad 0 & (d_V)_N^* d_N \end{array}\right]. \qquad (3.1)
\end{aligned}
$$

For convenience, we define

$$
\begin{aligned}
h'_N &= (p_V)_N^* d_N \qquad \text{and} \\
(d_T)_N &= (d_V)_N^* d_N,
\end{aligned}
$$

so that (3.1) may be written more simply:

$$
\tilde{V}_N^* R = \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-1,:)} \\ \hline X_N \begin{pmatrix} a_{N-1}\cdots a_2 q_1 & \cdots & a_{N-1} q_{N-2} & q_{N-1} \end{pmatrix} & h'_N \\ 0 \quad \cdots \quad 0 & (d_T)_N \end{array}\right]. \qquad (3.2)
$$

For the next step, observe that

$$R(N-1:N, 1:N-2) = \begin{bmatrix} p_{N-1} \\ X_N a_{N-1} \\ 0 \end{bmatrix} \begin{bmatrix} a_{N-1}\cdots a_2 q_1 & \cdots & a_{N-1} q_{N-2} \end{bmatrix},$$

so QR factorization applied to

$$\begin{bmatrix} p_{N-1} \\ X_N a_{N-1} \end{bmatrix}$$

produces factors

$$\begin{bmatrix} p_{N-1} \\ X_N a_{N-1} \end{bmatrix} = Q_{N-1} \begin{bmatrix} X_{N-1} \\ 0 \end{bmatrix}$$

in a way analogous to the application of a modified rotation defined in (2.1) and applied to the specific case of (2.7).

As with Q_N, Q_{N-1} may be separated into blocks that will ultimately be used as generators to characterize the quasiseparable structure of the upper triangular factor T. Here,

$$Q_{N-1} = \begin{bmatrix} (p_V)_{N-1} & (d_V)_{N-1} \\ (a_V)_{N-1} & (q_V)_{N-1} \end{bmatrix}$$

with dimensions:

$$\begin{aligned} (p_V)_{N-1}: &\quad m_{N-1} \times \rho_{N-2} \\ (d_V)_{N-1}: &\quad m_{N-1} \times \nu_{N-1} \\ (a_V)_{N-1}: &\quad \rho_{N-1} \times \rho_{N-2} \\ (q_V)_{N-1}: &\quad \rho_{N-1} \times \nu_{N-1}, \end{aligned}$$

where $\rho_{N-2} = \min(m_{N-1} + \rho_{N-1}, r'_{N-2})$ and $\nu_{N-1} = m_{N-1} + \rho_{N-1} - \rho_{N-2}$.

Let $V_{N-1} = I_{\eta_{N-1}} \oplus Q_{N-1} \oplus I_{\phi_{N-1}}$, where

$$\eta_{N-1} = \sum_{k=1}^{N-2} m_k \quad \text{and} \quad \phi_{N-1} = \sum_{k=N}^{N} m_k = m_N.$$

Then multiply V_{N-1}^* by the previous product to see its effects on the $(N-1)$ and N block rows of R.

$$
\begin{aligned}
&V_{N-1}^*\tilde{V}_N^*R \\
&= \left[\begin{array}{c|c|c} I_{\eta_{N-1}} & & \\ \hline & Q_{N-1}^* & \\ \hline & & I_{\phi_{N-1}} \end{array}\right]
\left[\begin{array}{ccccc} \multicolumn{5}{c}{R(1:N-2,:)} \\ \hline p_{N-1}a_{N-2}\cdots a_2q_1 & \cdots & p_{N-1}q_{N-2} & d_{N-1} & g_{N-1}h_N \\ X_Na_{N-1}a_{N-2}\cdots a_2q_1 & \cdots & X_Na_{N-1}q_{N-2} & X_Nq_{N-1} & h'_N \\ \hline 0 & \cdots & 0 & 0 & (d_T)_N \end{array}\right] \\
&= \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-2,:)} \\ \hline Q_{N-1}^*\begin{bmatrix} p_{N-1} \\ X_Na_{N-1}\end{bmatrix}\begin{bmatrix} a_{N-2}\cdots a_2q_1 & \cdots & q_{N-2}\end{bmatrix} & Q_{N-1}^*\begin{bmatrix} d_{N-1} & g_{N-1}h_N \\ X_Nq_{N-1} & h'_N\end{bmatrix} \\ \hline 0 & (d_T)_N \end{array}\right] \\
&= \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-2,:)} \\ \hline \begin{bmatrix} X_{N-1} \\ 0\end{bmatrix}\begin{bmatrix} a_{N-2}\cdots a_2q_1 & \cdots & q_{N-2}\end{bmatrix} & Q_{N-1}^*\begin{bmatrix} d_{N-1} & g_{N-1}h_N \\ X_Nq_{N-1} & h'_N\end{bmatrix} \\ \hline 0 & (d_T)_N \end{array}\right] \\
&= \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-2,:)} \\ \hline \begin{bmatrix} X_{N-1} \\ 0\end{bmatrix}\begin{bmatrix} a_{N-2}\cdots a_2q_1 & \cdots & q_{N-2}\end{bmatrix} & \begin{bmatrix} (p_V)_{N-1}^* & (a_V)_{N-1}^* \\ (d_V)_{N-1}^* & (q_V)_{N-1}^*\end{bmatrix}\begin{bmatrix} d_{N-1} & g_{N-1}h_N \\ X_Nq_{N-1} & h'_N\end{bmatrix} \\ \hline 0 & (d_T)_N \end{array}\right] \\
&= \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-2,:)} \\ \hline \begin{matrix} X_{N-1}a_{N-2}\cdots a_2q_1 & \cdots & X_{N-1}q_{N-2} \\ 0 & \cdots & 0 \end{matrix} & \begin{bmatrix} (p_V)_{N-1}^* & (a_V)_{N-1}^* \\ (d_V)_{N-1}^* & (q_V)_{N-1}^*\end{bmatrix}\begin{bmatrix} d_{N-1} & g_{N-1}h_N \\ X_Nq_{N-1} & h'_N\end{bmatrix} \\ \hline 0 & (d_T)_N \end{array}\right]. \quad (3.3)
\end{aligned}
$$

The complicated part is taking place in the product

$$\begin{bmatrix} (p_V)^*_{N-1} & (a_V)^*_{N-1} \\ (d_V)^*_{N-1} & (q_V)^*_{N-1} \end{bmatrix} \begin{bmatrix} d_{N-1} & g_{N-1}h_N \\ X_N q_{N-1} & h'_N \end{bmatrix},$$

which warrants some simplifying and renaming:

$$\begin{aligned} &\begin{bmatrix} (p_V)^*_{N-1} & (a_V)^*_{N-1} \\ (d_V)^*_{N-1} & (q_V)^*_{N-1} \end{bmatrix} \begin{bmatrix} d_{N-1} & g_{N-1}h_N \\ X_N q_{N-1} & h'_N \end{bmatrix} \\ &= \begin{bmatrix} (p_V)^*_{N-1} d_{N-1} + (a_V)^*_{N-1} X_N q_{N-1} & \begin{bmatrix} (p_V)^*_{N-1} g_{N-1} & (a_V)^*_{N-1} \end{bmatrix} \begin{bmatrix} h_N \\ h'_N \end{bmatrix} \\ (d_V)^*_{N-1} d_{N-1} + (q_V)^*_{N-1} X_N q_{N-1} & \begin{bmatrix} (d_V)^*_{N-1} g_{N-1} & (q_V)^*_{N-1} \end{bmatrix} \begin{bmatrix} h_N \\ h'_N \end{bmatrix} \end{bmatrix}. \end{aligned}$$

Let

$$\begin{aligned} h'_{N-1} &= (p_V)^*_{N-1} d_{N-1} + (a_V)^*_{N-1} X_N q_{N-1}, \\ b'_{N-1} &= \begin{bmatrix} (p_V)^*_{N-1} g_{N-1} & (a_V)^*_{N-1} \end{bmatrix}, \\ (d_T)_{N-1} &= (d_V)^*_{N-1} d_{N-1} + (q_V)^*_{N-1} X_N q_{N-1}, \\ (g_T)_{N-1} &= \begin{bmatrix} (d_V)^*_{N-1} g_{N-1} & (q_V)^*_{N-1} \end{bmatrix}, \text{ and} \\ (h_T)_N &= \begin{bmatrix} h_N \\ h'_N \end{bmatrix}. \end{aligned}$$

Then

$$\begin{bmatrix} (p_V)^*_{N-1} & (a_V)^*_{N-1} \\ (d_V)^*_{N-1} & (q_V)^*_{N-1} \end{bmatrix} \begin{bmatrix} d_{N-1} & g_{N-1}h_N \\ X_N q_{N-1} & h'_N \end{bmatrix} = \begin{bmatrix} h'_{N-1} & b'_{N-1}(h_T)_N \\ (d_T)_{N-1} & (g_T)_{N-1}(h_T)_N \end{bmatrix}. \tag{3.4}$$

By substituting (3.4) into (3.3), we have

$$
\begin{aligned}
&V_{N-1}^* \tilde{V}_N^* R \\
&= \left[\begin{array}{ccccc}
\multicolumn{5}{c}{R(1:N-2,:)} \\ \hline
X_{N-1}a_{N-2}\cdots a_2 q_1 & \cdots & X_{N-1}q_{N-2} & h'_{N-1} & b'_{N-1}\begin{bmatrix} h_N \\ h'_N \end{bmatrix} \\
0 & \cdots & 0 & (d_T)_{N-1} & (g_T)_{N-1}(h_T)_N \\ \hline
0 & & \cdots & 0 & (d_T)_N
\end{array}\right] \\
&= \left[\begin{array}{ccccc}
\multicolumn{5}{c}{R(1:N-3,:)} \\ \hline
\multicolumn{5}{c}{R(N-2,:)} \\
X_{N-1}a_{N-2}\cdots a_2 q_1 & \cdots & X_{N-1}q_{N-2} & h'_{N-1} & b'_{N-1}\begin{bmatrix} h_N \\ h'_N \end{bmatrix} \\ \hline
0 & \cdots & 0 & (d_T)_{N-1} & (g_T)_{N-1}(h_T)_N \\
0 & & \cdots & 0 & (d_T)_N
\end{array}\right] \\
&= \left[\begin{array}{cccccc}
\multicolumn{6}{c}{R(1:N-3,:)} \\ \hline
p_{N-2}a_{N-3}\cdots a_2 q_1 & \cdots & p_{N-2}q_{N-3} & d_{N-2} & g_{N-2}h_{N-1} & g_{N-2}b_{N-1}h_N \\
X_{N-1}a_{N-2}\cdots a_2 q_1 & \cdots & X_{N-1}a_{N-2}q_{N-3} & X_{N-1}q_{N-2} & h'_{N-1} & b'_{N-1}\begin{bmatrix} h_N \\ h'_N \end{bmatrix} \\ \hline
\multicolumn{6}{c}{T(N-1:N,:)}
\end{array}\right]. \quad (3.5)
\end{aligned}
$$

In the next step, apply QR factorization to $\begin{bmatrix} p_{N-2} \\ X_{N-1}a_{N-2} \end{bmatrix}$:

$$
\begin{bmatrix} p_{N-2} \\ X_{N-1}a_{N-2} \end{bmatrix} = Q_{N-2}\begin{bmatrix} X_{N-2} \\ 0 \end{bmatrix} = \begin{bmatrix} (p_V)_{N-2} & (d_V)_{N-2} \\ (a_V)_{N-2} & (q_V)_{N-2} \end{bmatrix}\begin{bmatrix} X_{N-2} \\ 0 \end{bmatrix}.
$$

Let $V_{N-2} = I_{\eta_{N-2}} \oplus Q_{N-2} \oplus I_{\phi_{N-2}}$. Then V_{N-2}^* acts on two block rows of the previous product, given in (3.5). So

$$V_{N-2}^* V_{N-1}^* \tilde{V}_N^* R = \left[\begin{array}{c|c|c} I_{\eta_{N-2}} & & \\ \hline & Q_{N-2}^* & \\ \hline & & I_{\phi_{N-2}} \end{array}\right] V_{N-1}^* \tilde{V}_N^* R$$

$$= \left[\begin{array}{cc} \multicolumn{2}{c}{R(1:N-3,:)} \\ \hline \begin{bmatrix} X_{N-2} \\ 0 \end{bmatrix} \begin{bmatrix} a_{N-3} \cdots a_2 q_1 & \cdots & q_{N-3} \end{bmatrix} \quad Q_{N-2}^* \begin{bmatrix} d_{N-2} \\ X_{N-1} q_{N-2} \end{bmatrix} & T'_{N-2} \\ \hline \multicolumn{2}{c}{T(N-1:N,:)} \end{array}\right] \tag{3.6}$$

where

$$\begin{bmatrix} X_{N-2} \\ 0 \end{bmatrix} \begin{bmatrix} a_{N-3} \cdots a_2 q_1 & \cdots & q_{N-3} \end{bmatrix} = \begin{bmatrix} X_{N-2} a_{N-3} \cdots a_2 q_1 & \cdots & X_{N-2} q_{N-3} \\ 0 & \cdots & 0 \end{bmatrix} \tag{3.7}$$

and

$$\begin{aligned} Q_{N-2}^* \begin{bmatrix} d_{N-2} \\ X_{N-1} q_{N-2} \end{bmatrix} &= \begin{bmatrix} (p_V)_{N-2}^* & (a_V)_{N-2}^* \\ (d_V)_{N-2}^* & (q_V)_{N-2}^* \end{bmatrix} \begin{bmatrix} d_{N-2} \\ X_{N-1} q_{N-2} \end{bmatrix} \\ &= \begin{bmatrix} (p_V)_{N-2}^* d_{N-2} + (a_V)_{N-2}^* X_{N-1} q_{N-2} \\ (d_V)_{N-2}^* d_{N-2} + (q_V)_{N-2}^* X_{N-1} q_{N-2} \end{bmatrix} \\ &= \begin{bmatrix} h'_{N-2} \\ (d_T)_{N-2} \end{bmatrix} \end{aligned} \tag{3.8}$$

by setting

$$\begin{aligned} h'_{N-2} &= (p_V)_{N-2}^* d_{N-2} + (a_V)_{N-2}^* X_{N-1} q_{N-2} \text{ and} \\ (d_T)_{N-2} &= (d_V)_{N-2}^* d_{N-2} + (q_V)_{N-2}^* X_{N-1} q_{N-2}, \end{aligned}$$

and

$$T'_{N-2} = Q^*_{N-2}\begin{bmatrix} g_{N-2}h_{N-1} & g_{N-2}b_{N-1}h_N \\ h'_{N-1} & \begin{bmatrix}(p_V)^*_{N-1}g_{N-1} & (a_V)^*_{N-1}\end{bmatrix}\begin{bmatrix}h_N \\ h'_N\end{bmatrix}\end{bmatrix}$$

$$= \begin{bmatrix}(p_V)^*_{N-2} & (a_V)^*_{N-2} \\ (d_V)^*_{N-2} & (q_V)^*_{N-2}\end{bmatrix}\begin{bmatrix} g_{N-2}h_{N-1} & g_{N-2}b_{N-1}h_N \\ h'_{N-1} & \begin{bmatrix}(p_V)^*_{N-1}g_{N-1} & (a_V)^*_{N-1}\end{bmatrix}\begin{bmatrix}h_N \\ h'_N\end{bmatrix}\end{bmatrix}$$

$$= \begin{bmatrix}(p_V)^*_{N-2} & (a_V)^*_{N-2} \\ (d_V)^*_{N-2} & (q_V)^*_{N-2}\end{bmatrix}\begin{bmatrix} g_{N-2} & 0 \\ 0 & 1\end{bmatrix}\left(\begin{matrix} h_{N-1} & \begin{bmatrix}b_{N-1} & 0\end{bmatrix}\begin{bmatrix}h_N \\ h'_N\end{bmatrix} \\ h'_{N-1} & \begin{bmatrix}(p_V)^*_{N-1}g_{N-1} & (a_V)^*_{N-1}\end{bmatrix}\begin{bmatrix}h_N \\ h'_N\end{bmatrix}\end{matrix}\right)$$

$$= \begin{bmatrix}(p_V)^*_{N-2}g_{N-2} & (a_V)^*_{N-2} \\ (d_V)^*_{N-2}g_{N-2} & (q_V)^*_{N-2}\end{bmatrix}\left(\begin{bmatrix}h_{N-1} \\ h'_{N-1}\end{bmatrix} \quad \begin{bmatrix} b_{N-1} & 0 \\ (p_V)^*_{N-1}g_{N-1} & (a_V)^*_{N-1}\end{bmatrix}\begin{bmatrix}h_N \\ h'_N\end{bmatrix}\right). \quad (3.9)$$

By setting

$$\begin{aligned}
b'_{N-2} &= \begin{bmatrix}(p_V)^*_{N-2}g_{N-2} & (a_V)^*_{N-2}\end{bmatrix}, \\
(g_T)_{N-2} &= \begin{bmatrix}(d_V)^*_{N-2}g_{N-2} & (q_V)^*_{N-2}\end{bmatrix}, \\
(h_T)_{N-1} &= \begin{bmatrix}h_{N-1} \\ h'_{N-1}\end{bmatrix}, \text{ and} \\
(b_T)_{N-1} &= \begin{bmatrix} b_{N-1} & 0 \\ (p_V)^*_{N-1}g_{N-1} & (a_V)^*_{N-1}\end{bmatrix},
\end{aligned}$$

(3.9) can be written more simply as:

$$
\begin{aligned}
&Q^*_{N-2}\begin{bmatrix} g_{N-2}h_{N-1} & g_{N-2}b_{N-1}h_N \\ h'_{N-1} & \begin{bmatrix} (p_V)^*_{N-1}g_{N-1} & (a_V)^*_{N-1} \end{bmatrix}\begin{bmatrix} h_N \\ h'_N \end{bmatrix} \end{bmatrix} \\
&= \begin{bmatrix} (p_V)^*_{N-2}g_{N-2} & (a_V)^*_{N-2} \\ \multicolumn{2}{c}{(g_T)_{N-2}} \end{bmatrix}\left((h_T)_{N-1} \quad (b_T)_{N-1}(h_T)_N\right) \\
&= \left(\begin{array}{cc} \multicolumn{2}{c}{b'_{N-2}\begin{bmatrix} (h_T)_{N-1} & (b_T)_{N-1}(h_T)_N \end{bmatrix}} \\ \hline (g_T)_{N-2}(h_T)_{N-1} & (g_T)_{N-2}(b_T)_{N-1}(h_T)_N \end{array}\right). \qquad (3.10)
\end{aligned}
$$

Now that each detail has been worked out, (3.6) may be simplified. By substituting (3.7), (3.8),and (3.10) into (3.6), we have

$$
\begin{aligned}
&V_{N-2}^* V_{N-1}^* \tilde{V}_N^* R \\
&= \left[\begin{array}{cccc}
\multicolumn{4}{c}{R(1:N-3,:)} \\ \hline
\begin{bmatrix} X_{N-2} \\ 0 \end{bmatrix} \begin{bmatrix} a_{N-3}\cdots a_2 q_1 & \cdots & q_{N-3} \end{bmatrix} & Q_{N-2}^* & \begin{bmatrix} d_{N-2} \\ X_{N-1} q_{N-2} \end{bmatrix} & T'_{N-2} \\ \hline
\multicolumn{4}{c}{T(N-1:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{cccccc}
\multicolumn{6}{c}{R(1:N-3,:)} \\ \hline
X_{N-2}\begin{pmatrix} a_{N-3}\cdots a_2 q_1 & \cdots & q_{N-3} \end{pmatrix} & & & h'_{N-2} & \multicolumn{2}{c}{b'_{N-2}\begin{bmatrix} (h_T)_{N-1} & (b_T)_{N-1}(h_T)_N \end{bmatrix}} \\
0 & \cdots & 0 & (d_T)_{N-2} & (g_T)_{N-2}(h_T)_{N-1} & (g_T)_{N-2}(b_T)_{N-1}(h_T)_N \\ \hline
\multicolumn{6}{c}{T(N-1:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{cccccc}
\multicolumn{6}{c}{R(1:N-4,:)} \\ \hline
\multicolumn{6}{c}{R(N-3,:)} \\
X_{N-2}\begin{pmatrix} a_{N-3}\cdots a_2 q_1 & \cdots & q_{N-3} \end{pmatrix} & & & h'_{N-2} & \multicolumn{2}{c}{b'_{N-2}\begin{bmatrix} (h_T)_{N-1} & (b_T)_{N-1}(h_T)_N \end{bmatrix}} \\ \hline
0 & \cdots & 0 & (d_T)_{N-2} & (g_T)_{N-2}(h_T)_{N-1} & (g_T)_{N-2}(b_T)_{N-1}(h_T)_N \\
0 & \cdots & & 0 & (d_T)_{N-1} & (g_T)_{N-1}(h_T)_N \\
0 & & \cdots & & 0 & (d_T)_N
\end{array}\right] \\
&= \left[\begin{array}{ccc}
\multicolumn{3}{c}{R(1:N-4,:)} \\ \hline
\multicolumn{3}{c}{R(N-3,:)} \\
X_{N-2}\begin{pmatrix} a_{N-3}\cdots a_2 q_1 & \cdots & q_{N-3} \end{pmatrix} & h'_{N-2} & b'_{N-2}\begin{bmatrix} (h_T)_{N-1} & (b_T)_{N-1}(h_T)_N \end{bmatrix} \\ \hline
\multicolumn{3}{c}{T(N-2:N,:)}
\end{array}\right]. \qquad (3.11)
\end{aligned}
$$

Everything in the latest step of factoring may be applied iteratively for block rows R_i, $i = N-1, \ldots, 2$. At each step, perform QR factorization:

$$\begin{bmatrix} p_i \\ X_{i+1} a_i \end{bmatrix} = Q_i \begin{bmatrix} X_i \\ 0 \end{bmatrix}.$$

Compute $\rho_{i-1} = \min(m_i + \rho_i, r'_{i-1})$ and $\nu_i = m_i + \rho_i - \rho_{i-1}$. Then partition

$$Q_i = \begin{bmatrix} (p_V)_i & (d_V)_i \\ (a_V)_i & (q_V)_i \end{bmatrix}$$

according to the dimensions:

$$\begin{aligned} (p_V)_i :& \quad m_i \times \rho_{i-1} \\ (d_V)_i :& \quad m_i \times \nu_i \\ (a_V)_i :& \quad \rho_i \times \rho_{i-1} \\ (q_V)_i :& \quad \rho_i \times \nu_i, \end{aligned}$$

so

$$\begin{bmatrix} p_i \\ X_{i+1} a_i \end{bmatrix} = Q_i \begin{bmatrix} X_i \\ 0 \end{bmatrix} = \begin{bmatrix} (p_V)_i & (d_V)_i \\ (a_V)_i & (q_V)_i \end{bmatrix} \begin{bmatrix} X_i \\ 0 \end{bmatrix}.$$

Let

$$\eta_i = \sum_{k=1}^{i-1} m_k, \qquad \phi_i = \sum_{k=i+1}^{N} \nu_k, \qquad i = 1, \ldots, N$$

and let

$$V_i = I_{\eta_i} \oplus Q_i \oplus I_{\phi_i} = \left[\begin{array}{c|c|c} I_{\eta_i} & & \\ \hline & Q_i & \\ \hline & & I_{\phi_i} \end{array}\right].$$

For notational convenience, compute

$$\begin{aligned}
h_i' &= (p_V)_i^* d_i + (a_V)_i^* X_{i+1} q_i, \\
(d_T)_i &= (d_V)_i^* d_i + (q_V)_i^* X_{i+1} q_i, \\
b_i' &= \begin{bmatrix} (p_V)_i^* g_i & (a_V)_i^* \end{bmatrix}, \\
(g_T)_i &= \begin{bmatrix} (d_V)_i^* g_i & (q_V)_i^* \end{bmatrix}, \\
(h_T)_{i+1} &= \begin{bmatrix} h_{i+1} \\ h_{i+1}' \end{bmatrix}, \text{ and} \\
(b_T)_i &= \begin{bmatrix} b_i & 0 \\ (p_V)_i^* g_i & (a_V)_i^* \end{bmatrix}.
\end{aligned}$$

Then V_i^* acts on the i^{th} and $(i+1)^{th}$ block rows of the previous product, $V_{i+1}^* \cdots V_{N-1}^* \tilde{V}_N^* R$:

$$V_i^*(V_{i+1}^* \cdots V_{N-1}^* \tilde{V}_N^* R) = \left[\begin{array}{cccc} \multicolumn{4}{c}{R(1:i-1,:)} \\ \hline \begin{bmatrix} X_i \\ 0 \end{bmatrix}\begin{bmatrix} a_{i-1}\cdots a_2 q_1 & \cdots & q_{i-1} \end{bmatrix} & Q_i^* & \begin{bmatrix} d_i \\ X_{i+1} q_i \end{bmatrix} & T_i' \\ \hline \multicolumn{4}{c}{T(i+1:N,:)} \end{array}\right] \tag{3.12}$$

where

$$\begin{aligned}
&\begin{bmatrix} X_i \\ 0 \end{bmatrix}\begin{bmatrix} a_{i-1}\cdots a_2 q_1 & a_{i-1}\cdots a_3 q_2 & \cdots & a_{i-1} q_{i-2} & q_{i-1} \end{bmatrix} \\
&= \begin{pmatrix} X_i a_{i-1} \begin{bmatrix} a_{i-2}\cdots a_2 q_1 & a_{i-2}\cdots a_3 q_2 & \cdots & q_{i-2} \end{bmatrix} & X_i q_{i-1} \\ 0 \quad\quad \cdots \quad\quad 0 & 0 \end{pmatrix}
\end{aligned} \tag{3.13}$$

and

$$Q_i^* \begin{bmatrix} d_i \\ X_{i+1}q_i \end{bmatrix} = \begin{bmatrix} (p_V)_i^* & (a_V)_i^* \\ (d_V)_i^* & (q_V)_i^* \end{bmatrix} \begin{bmatrix} d_i \\ X_{i+1}q_i \end{bmatrix}$$
$$= \begin{bmatrix} (p_V)_i^* d_i + (a_V)_i^* X_{i+1} q_i \\ (d_V)_i^* d_i + (q_V)_i^* X_{i+1} q_i \end{bmatrix}$$
$$= \begin{bmatrix} h_i' \\ (d_T)_i \end{bmatrix} \tag{3.14}$$

and

$$T_i' = Q_i^* \left[\begin{pmatrix} g_i h_{i+1} \\ h_{i+1}' \end{pmatrix} \quad \begin{pmatrix} g_i b_{i+1} \begin{pmatrix} h_{i+2} & b_{i+2}h_{i+3} & \cdots & b_{i+2}\cdots b_{N-1}h_N \end{pmatrix} \\ b_{i+1}' \begin{bmatrix} (h_T)_{i+2} & (b_T)_{i+2}(h_T)_{i+3} & \cdots & (b_T)_{i+2}\cdots(b_T)_{N-1}(h_T)_N \end{bmatrix} \end{pmatrix} \right]$$
$$= \begin{bmatrix} (p_V)_i^* & (a_V)_i^* \\ (d_V)_i^* & (q_V)_i^* \end{bmatrix} \begin{bmatrix} g_i & 0 \\ 0 & 1 \end{bmatrix} \left[\begin{pmatrix} h_{i+1} \\ h_{i+1}' \end{pmatrix} \quad \begin{pmatrix} b_{i+1} \begin{pmatrix} h_{i+2} & \cdots & b_{i+2}\cdots b_{N-1}h_N \end{pmatrix} \\ b_{i+1}' \begin{bmatrix} (h_T)_{i+2} & \cdots & (b_T)_{i+2}\cdots(b_T)_{N-1}(h_T)_N \end{bmatrix} \end{pmatrix} \right]$$
$$= \begin{bmatrix} (p_V)_i^* g_i & (a_V)_i^* \\ (d_V)_i^* g_i & (q_V)_i^* \end{bmatrix} \left[\begin{pmatrix} h_{i+1} \\ h_{i+1}' \end{pmatrix} \quad \begin{pmatrix} b_{i+1} \begin{pmatrix} h_{i+2} & \cdots & b_{i+2}\cdots b_{N-1}h_N \end{pmatrix} \\ b_{i+1}' \begin{bmatrix} (h_T)_{i+2} & \cdots & (b_T)_{i+2}\cdots(b_T)_{N-1}(h_T)_N \end{bmatrix} \end{pmatrix} \right]$$
$$= \begin{bmatrix} b_i' \\ (g_T)_i \end{bmatrix} \begin{bmatrix} h_{i+1} & b_{i+1}h_{i+2} & \cdots & b_{i+1}b_{i+2}\cdots b_{N-1}h_N \\ h_{i+1}' & b_{i+1}'(h_T)_{i+2} & \cdots & b_{i+1}'(b_T)_{i+2}\cdots(b_T)_{N-1}(h_T)_N \end{bmatrix}$$
$$= \begin{bmatrix} b_i' \\ (g_T)_i \end{bmatrix} \begin{bmatrix} (h_T)_{i+1} & (b_T)_{i+1}(h_T)_{i+2} & \cdots & (b_T)_{i+1}(b_T)_{i+2}\cdots(b_T)_{N-1}(h_T)_N \end{bmatrix} \tag{3.15}$$
$$= \begin{pmatrix} b_i' \begin{bmatrix} (h_T)_{i+1} & (b_T)_{i+1}(h_T)_{i+2} & \cdots & (b_T)_{i+1}(b_T)_{i+2}\cdots(b_T)_{N-1}(h_T)_N \end{bmatrix} \\ (g_T)_i(h_T)_{i+1} \quad (g_T)_i(b_T)_{i+1}(h_T)_{i+2} \quad \cdots \quad (g_T)_i(b_T)_{i+1}(b_T)_{i+2}\cdots(b_T)_{N-1}(h_T)_N \end{pmatrix} \tag{3.16}$$

Note that (3.15) can be clearly seen as follows:

$$\begin{aligned}
&\begin{pmatrix} b_{i+1}b_{i+2}\cdots b_{k-1}h_k \\ b'_{i+1}(b_T)_{i+2}\cdots(b_T)_{k-1}(h_T)_k \end{pmatrix} \\
&= \begin{pmatrix} \begin{bmatrix} b_{i+1} & 0 \end{bmatrix} \begin{bmatrix} b_{i+2} & 0 \\ * & * \end{bmatrix} \cdots \begin{bmatrix} b_{k-2} & 0 \\ * & * \end{bmatrix} \begin{bmatrix} b_{k-1} & 0 \\ * & * \end{bmatrix} \begin{bmatrix} h_k \\ h'_k \end{bmatrix} \\ \begin{bmatrix} (p_V)^*_{i+1}g_{i+1} & (a_V)^*_{i+1} \end{bmatrix} \begin{bmatrix} b_{i+2} & 0 \\ (p_V)^*_{i+2}g_{i+2} & (a_V)^*_{i+2} \end{bmatrix} \cdots \begin{bmatrix} b_{k-1} & 0 \\ (p_V)^*_{k-1}g_{k-1} & (a_V)^*_{k-1} \end{bmatrix} \begin{bmatrix} h_k \\ h'_k \end{bmatrix} \end{pmatrix} \\
&= \begin{bmatrix} b_{i+1} & 0 \\ (p_V)^*_{i+1}g_{i+1} & (a_V)^*_{i+1} \end{bmatrix} \begin{bmatrix} b_{i+2} & 0 \\ (p_V)^*_{i+2}g_{i+2} & (a_V)^*_{i+2} \end{bmatrix} \cdots \begin{bmatrix} b_{k-1} & 0 \\ (p_V)^*_{k-1}g_{k-1} & (a_V)^*_{k-1} \end{bmatrix} \begin{bmatrix} h_k \\ h'_k \end{bmatrix} \\
&= (b_T)_{i+1}(b_T)_{i+2}\cdots(b_T)_{k-1}(h_T)_k, \qquad k = i+2, \ldots, N. \qquad (3.17)
\end{aligned}$$

It is also worth noting that all of the complexity from the lower and upper triangles of R is accumulated in the upper-triangular blocks of T. The order of these terms is ρ', which is generally the sum $r' + r''$. Briefly consider one simple example that illustrates the significant amount of information accumulated in one block of T:

$$\begin{aligned}
&(g_T)_i(b_T)_{i+1}(h_T)_{i+2} \\
&\quad = \begin{bmatrix} (d_V)^*_i g_i & (q_V)^*_i \end{bmatrix} \begin{bmatrix} b_{i+1} & 0 \\ (p_V)^*_{i+1}g_{i+1} & (a_V)^*_{i+1} \end{bmatrix} \begin{bmatrix} h_{i+2} \\ h'_{i+2} \end{bmatrix} \\
&\quad = \begin{bmatrix} (d_V)^*_i g_i & (q_V)^*_i \end{bmatrix} \begin{bmatrix} b_{i+1} & 0 \\ (p_V)^*_{i+1}g_{i+1} & (a_V)^*_{i+1} \end{bmatrix} \begin{bmatrix} h_{i+2} \\ (p_V)_{i+2}d_{i+2} + (a_V)_{i+2}X_{i+3}q_{i+2} \end{bmatrix} \\
&\quad = \begin{bmatrix} (d_V)^*_i g_i & (q_V)^*_i \end{bmatrix} \begin{bmatrix} b_{i+1}h_{i+2} \\ (p_V)^*_{i+1}g_{i+1}h_{i+2} + (a_V)^*_{i+1}(p_V)_{i+2}d_{i+2} + (a_V)^*_{i+1}(a_V)_{i+2}X_{i+3}q_{i+2} \end{bmatrix}
\end{aligned}$$

$$= \begin{bmatrix} (d_V)^*_i g_i b_{i+1}h_{i+2} + (q_V)^*_i(p_V)^*_{i+1}g_{i+1}h_{i+2} + (q_V)^*_i(a_V)^*_{i+1}(p_V)_{i+2}d_{i+2} + (q_V)^*_i(a_V)^*_{i+1}(a_V)_{i+2}X_{i+3}q_{i+2} \end{bmatrix}.$$

This simple block demonstrates approximately the least complexity in a block of T, in this case, one formed by the product of only three generators, i.e. R_{ij} where $j = i + 2$.

Substituting (3.13), (3.14), and (3.16) into (3.12) gives

$$
\begin{aligned}
&V_i^* V_{i+1}^* \cdots V_{N-1}^* \tilde{V}_N^* R \\
&= \left[\begin{array}{cccccc}
\multicolumn{6}{c}{R(1:i-1,:)} \\ \hline
X_i a_{i-1} & \begin{bmatrix} a_{i-2} \cdots a_2 q_1 & \cdots & q_{i-2} \end{bmatrix} & X_i q_{i-1} & h_i' & & \\
0 & \cdots \qquad\qquad 0 & 0 & (d_T)_i & & T_i' \\ \hline
\multicolumn{6}{c}{T(i+1:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{cccccc}
\multicolumn{6}{c}{R(1:i-2,:)} \\ \hline
X_i a_{i-1} & \begin{bmatrix} a_{i-2} \cdots a_2 q_1 & \cdots & q_{i-2} \end{bmatrix} & X_i q_{i-1} & h_i' & b_i' & \begin{bmatrix} (h_T)_{i+1} & \cdots & (b_T)_{i+1} \cdots (h_T)_N \end{bmatrix} \\
0 & \cdots \qquad\qquad 0 & & (d_T)_i & \multicolumn{2}{c}{(g_T)_i (h_T)_{i+1} \quad \cdots \quad (g_T)_i (b_T)_{i+1} \cdots (h_T)_N} \\ \hline
\multicolumn{6}{c}{T(i+1:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{cccccc}
\multicolumn{6}{c}{R(1:i-2,:)} \\ \hline
\multicolumn{6}{c}{R(i-1,:)} \\
X_i a_{i-1} & \begin{bmatrix} a_{i-2} \cdots a_2 q_1 & \cdots & q_{i-2} \end{bmatrix} & X_i q_{i-1} & h_i' & b_i' & \begin{bmatrix} (h_T)_{i+1} & \cdots & (b_T)_{i+1} \cdots (h_T)_N \end{bmatrix} \\ \hline
0 & \cdots \qquad\qquad 0 & & (d_T)_i & \multicolumn{2}{c}{(g_T)_i (h_T)_{i+1} \quad \cdots \quad (g_T)_i (b_T)_{i+1} \cdots (h_T)_N} \\
\multicolumn{6}{c}{T(i+1:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{cccccc}
\multicolumn{6}{c}{R(1:i-2,:)} \\ \hline
\multicolumn{6}{c}{R(i-1,:)} \\
X_i a_{i-1} & \begin{bmatrix} a_{i-2} \cdots a_2 q_1 & \cdots & q_{i-2} \end{bmatrix} & X_i q_{i-1} & h_i' & b_i' & \begin{bmatrix} (h_T)_{i+1} & \cdots & (b_T)_{i+1} \cdots (h_T)_N \end{bmatrix} \\ \hline
\multicolumn{6}{c}{T(i:N,:)}
\end{array}\right].
\end{aligned}
\tag{3.18}
$$

All that remains is to select a $\tilde{V}_1^*$ to multiply by the previously computed product $V_2^* \cdots V_{N-1}^* \tilde{V}_N^* R$ to cause $V = \tilde{V}_1^* V_2^* \cdots V_{N-1}^* \tilde{V}_N^*$ to be a unitary block lower triangular matrix. (This is the block unitary version of what was demonstrated in (2.4) and (2.5). All that is required is to select a unitary matrix Q_1 of dimensions $\nu_1 \times \nu_1$ where $\nu_1 = m_1 + \rho_1$. Partition

$$\tilde{V}_1 = \begin{bmatrix} (d_V)_1 \\ (q_V)_1 \end{bmatrix}$$

according to the dimensions:

$$\begin{aligned} (d_V)_1 : &\quad m_1 \times \nu_1 \\ (q_V)_1 : &\quad \rho_1 \times \nu_1. \end{aligned}$$

As in previous iterations, let $\tilde{V}_1 = Q_1 \oplus I_{\phi_1}$ and compute

$$\begin{aligned} (d_T)_1 &= (d_V)_1^* d_1 + (q_V)_1^* X_2 q_1, \\ (g_T)_1 &= \begin{bmatrix} (d_V)_1^* g_1 & (q_V)_1^* \end{bmatrix}, \text{ and} \\ (h_T)_2 &= \begin{bmatrix} h_2 \\ h_2' \end{bmatrix}. \end{aligned}$$

Then $\tilde{V}_1^*$ acts on the 1^{st} and 2^{nd} block rows of the product $V_2 \cdots V_{N-1}^* \tilde{V}_N^* R$. Details are very similar to those in the case of $i = N-1, \cdots, 2$.

$$
\begin{aligned}
T &= \tilde{V}_1^*(V_2^* \cdots V_{N-1}^* \tilde{V}_N^* R) \\
&= \tilde{V}_1^* \left[\begin{array}{c} R(1,:) \\ \begin{array}{cccc} X_2 q_1 & h_2' & b_2' \begin{bmatrix} (h_T)_3 & (b_T)_3 (h_T)_4 & \cdots & (b_T)_3 \cdots (h_T)_N \end{bmatrix} \end{array} \\ \hline T(2:N,:) \end{array} \right] \\
&= \left[\begin{array}{cc|c} (d_V)_1^* & (q_V)_1^* & \\ \hline & & I_{\phi_1} \end{array} \right] \left[\begin{array}{cccc} d_1 & g_1 h_2 & g_1 b_2 \begin{bmatrix} h_3 & b_3 h_4 & \cdots & b_3 \cdots h_N \end{bmatrix} \\ X_2 q_1 & h_2' & b_2' \begin{bmatrix} (h_T)_3 & (b_T)_3 (h_T)_4 & \cdots & (b_T)_3 \cdots (h_T)_N \end{bmatrix} \\ \hline \multicolumn{3}{c}{T(2:N,:)} \end{array} \right] \\
&= \left[\begin{array}{cc} (d_T)_1 & T_1' \\ \hline \multicolumn{2}{c}{T(2:N,:)} \end{array} \right]
\end{aligned} \tag{3.19}
$$

where

$$
\begin{aligned}
T_1' &= \begin{bmatrix} (d_V)_1^* & (q_V)_1^* \end{bmatrix} \begin{bmatrix} g_1 h_2 & g_1 b_2 \begin{bmatrix} h_3 & b_3 h_4 & \cdots & b_3 \cdots h_N \end{bmatrix} \\ h_2' & b_2' \begin{bmatrix} (h_T)_3 & (b_T)_3 (h_T)_4 & \cdots & (b_T)_3 \cdots (h_T)_N \end{bmatrix} \end{bmatrix} \\
&= \begin{bmatrix} (d_V)_1^* g_1 & (q_V)_1^* \end{bmatrix} \begin{bmatrix} \begin{pmatrix} h_2 \\ h_2' \end{pmatrix} & \begin{pmatrix} b_2 \begin{bmatrix} h_3 & \cdots & b_3 \cdots h_N \end{bmatrix} \\ b_2' \begin{bmatrix} (h_T)_3 & (b_T)_3 (h_T)_4 & \cdots & (b_T)_3 \cdots (h_T)_N \end{bmatrix} \end{pmatrix} \end{bmatrix} \\
&= (g_T)_1 \begin{bmatrix} h_2 & b_2 h_3 & b_2 b_3 h_4 & \cdots & b_2 b_3 \cdots b_{N-1} h_N \\ h_2' & b_2'(h_T)_3 & b_2'(b_T)_3 (h_T)_4 & \cdots & b_2'(b_T)_3 \cdots (b_T)_{N-1} h_N \end{bmatrix} \\
&= (g_T)_1 \begin{bmatrix} (h_T)_2 & (b_T)_2 (h_T)_3 & (b_T)_2 (b_T)_3 (h_T)_4 & \cdots & (b_T)_2 (b_T)_3 \cdots (b_T)_{N-1} h_N \end{bmatrix} \\
&= \begin{pmatrix} (g_T)_1 (h_T)_2 & (g_T)_1 (b_T)_2 (h_T)_3 & \cdots & (g_T)_1 (b_T)_2 \cdots (b_T)_{N-1} (h_T)_N \end{pmatrix},
\end{aligned} \tag{3.20}
$$

so, substituting (3.20) into (3.19), we have

$$
\begin{aligned}
T &= \left[\begin{array}{ccccc} (d_T)_1 & (g_T)_1(h_T)_2 & (g_T)_1(b_T)_2(h_T)_3 & \cdots & (g_T)_1(b_T)_2\cdots(b_T)_{N-1}(h_T)_N \\ \hline \multicolumn{5}{c}{T(2:N,:)} \end{array}\right] \\
&= \begin{bmatrix} (d_T)_1 & (g_T)_1(h_T)_2 & (g_T)_1(b_T)_2(h_T)_3 & \cdots & (g_T)_1(b_T)_2\cdots(b_T)_{N-1}(h_T)_N \\ 0 & (d_T)_2 & (g_T)_2(h_T)_3 & \cdots & (g_T)_2(b_T)_3\cdots(b_T)_{N-1}(h_T)_N \\ \vdots & & \ddots & \ddots & \vdots \\ & & \ddots & (d_T)_{N-1} & (g_T)_{N-1}(h_T)_N \\ 0 & & \cdots & 0 & (d_T)_N \end{bmatrix}
\end{aligned}
\tag{3.21}
$$

This is the exact result sought:

$$
\tilde{V}_1^* V_2^* \cdots V_{N-1}^* \tilde{V}_N^* R = T,
$$

so

$$
\begin{aligned}
R &= \tilde{V}_N V_{N-1} \cdots V_2 \tilde{V}_1 T \\
&= VT
\end{aligned}
$$

where V is unitary (as the product of unitary matrices) block lower triangular, and T is block upper triangular.

The process thus described is written more concisely as Algorithm 2.

Algorithm 2: Let R be a quasiseparable block matrix with generators as defined in (1.2). Then R admits factorization $R = VT$ where V is a block lower triangular unitary matrix, and T is a block upper triangular matrix according to the following steps.

1. Calculate generator dimensions.

   (a) First stage, performed on p_N:

$$\rho_{N-1} = \min(m_N, r'_{N-1})$$

$$\nu_N = m_N - \rho_{N-1}$$

$$\rho'_{N-1} = r''_{N-1} + \rho_{N-1}$$

(b) Middle stages, performed on p_k, $\quad k = N-1, \ldots, 2$:

$$\begin{aligned}
&\textbf{for} \quad k = N-1 : 2 \\
&\qquad \rho_{k-1} = \min(m_k + \rho_k, r'_{k-1}) \\
&\qquad \nu_k = m_k + \rho_k - \rho_{k-1} \\
&\qquad \rho'_{k-1} = r''_{k-1} + \rho_{k-1} \\
&\textbf{end}
\end{aligned}$$

(c) Final dimension used in inner-coprime factoring:

$$\nu_1 = m_1 + \rho_1$$

2. Use QR factorization row by row to zero out block rows of R.

(a) Perform QR factorization on last row (p_N). Determine generators of V and T.

$$[Q, r] = \mathrm{qr}(p_N)$$

$$(p_V)_N = Q(:, 1 : \rho_{N-1})$$

$$(d_V)_N = Q(:, \rho_{N-1} + 1 : m_N)$$

$$X_N = r(1 : \rho_{N-1}, 1 : r'_{N-1})$$

$$(d_T)_N = (d_V)^*_N d_N$$

$$h'_N = (p_V)^*_N d_N$$

$$(h_T)_N = \begin{bmatrix} h_N \\ h'_N \end{bmatrix}$$

(b) Middle stages, performed on $p_k, k = N-1, ..., 2$

for $k = N-1, \ldots, 2$

$$[Q, r] = qr\left(\begin{bmatrix} p_k \\ X_{k+1}a_k \end{bmatrix}\right)$$

$$(p_V)_k = Q(1 : m_k, 1 : \rho_{k-1})$$

$$(d_V)_k = Q(1 : m_k, \rho_{k-1}+1 : m_k + \rho_k)$$

$$(a_V)_k = Q(m_k + 1 : m_k + \rho_k, 1 : \rho_{k-1})$$

$$(q_V)_k = Q(m_k + 1 : m_k + \rho_k, \rho_{k-1} + 1 : m_k + \rho_k)$$

$$X_k = r(1 : \rho_{k-1} : 1 : r'_{k-1})$$

$$h' = (p_V)_k^* d_k + (a_V)_k^* X_{k+1} q_k$$

$$(h_T)_k = \begin{bmatrix} h_k \\ h'_k \end{bmatrix}$$

$$(b_T)_k = \begin{bmatrix} b_k & 0 \\ (p_V)_k^* g_k & (a_V)_k^* \end{bmatrix}$$

$$(g_T)_k = \begin{bmatrix} (d_V)_k^* g_k & (q_V)_k^* \end{bmatrix}$$

$$(d_T)_k = (d_V)_k^* d_k + (q_V)_k^* X_{k+1} q_k$$

end

(c) Final stage of inner-coprime factoring

$$Q = I_{\nu_1}$$

$$(d_V)_1 = Q(1 : m_1, 1 : \nu_1)$$

$$(q_V)_1 = Q(m_1 + 1 : \nu_1, 1 : \nu_1)$$

$$(d_T)_1 = (d_V)_1^* d_1 + (q_V)_1^* X_2 q_1$$

$$(g_T)_1 = \begin{bmatrix} (d_V)_1^* g_1 & (q_V)_1^* \end{bmatrix}$$

3.2 Inner-outer Factorization

In a way similar to the factoring of R into $R = VT$, T may be factored by applying QR factorization to two consecutive block rows, in this case working from the top to the bottom. The objective is to factor T into $T = US$, where U is a block upper triangular matrix and S is a block upper triangular *invertible* matrix with block entries of size $n_i \times n_j$. Note the important feature that diagonal blocks of S are square.

For the first two block rows, compute $s_1 = \nu_1 - n_1$. Perform QR factorization on $\begin{bmatrix} (d_T)_1 & (g_T)_1 \end{bmatrix}$ such that

$$\begin{bmatrix} (d_T)_1 & (g_T)_1 \end{bmatrix} = P_1 \begin{bmatrix} (d_S)_1 & (g_S)_1 \\ 0 & Y_1 \end{bmatrix} = \begin{bmatrix} (d_U)_1 & (g_U)_1 \end{bmatrix} \begin{bmatrix} (d_S)_1 & (g_S)_1 \\ 0 & Y_1 \end{bmatrix},$$

with dimensions:

$$\begin{aligned} (d_S)_1 :&\quad n_1 \times n_1 \\ (g_S)_1 :&\quad n_1 \times \rho'_1 \\ Y_1 :&\quad s_1 \times \rho'_1 \\ (d_U)_1 :&\quad \nu_1 \times n_1 \\ (g_U)_1 :&\quad \nu_1 \times s_1. \end{aligned}$$

Let $\tilde{U}_1 = P_1 \oplus I_{\phi_1}$. Then

$$
\begin{aligned}
\tilde{U}_1^* T &= \left[\begin{array}{c|c} (d_U)_1^* & \\ (g_U)_1^* & \\ \hline & I_{\phi_1} \end{array}\right] \left[\begin{array}{cccc} (d_T)_1 & (g_T)_1 (h_T)_2 & \cdots & (g_T)_1 (b_T)_2 \cdots (b_T)_{N-1} (h_T)_N \\ \hline \multicolumn{4}{c}{T(2:N,:)} \end{array}\right] \\
&= \left[\begin{array}{c|c} (d_U)_1^* & \\ (g_U)_1^* & \\ \hline & I_{\phi_1} \end{array}\right] \left[\begin{array}{ccc} (d_T)_1 & (g_T)_1 \left(\begin{array}{ccc} (h_T)_2 & \cdots & (b_T)_2 \cdots (b_T)_{N-1} (h_T)_N \end{array} \right) \\ \hline \multicolumn{2}{c}{T(2:N,:)} \end{array}\right] \\
&= \left[\begin{array}{c|c} (d_U)_1^* & \\ (g_U)_1^* & \\ \hline & I_{\phi_1} \end{array}\right] \left[\begin{array}{c} \begin{bmatrix} (d_T)_1 & (g_T)_1 \end{bmatrix} \begin{bmatrix} I_{n_1} & 0 & \cdots & 0 \\ 0 & (h_T)_2 & \cdots & (b_T)_2 \cdots (b_T)_{N-1} (h_T)_N \end{bmatrix} \\ \hline T(2:N,:) \end{array}\right] \\
&= \left[\begin{array}{c|c} P_1^* & \\ \hline & I_{\phi_1} \end{array}\right] \left[\begin{array}{c} P_1 \begin{bmatrix} (d_S)_1 & (g_S)_1 \\ 0 & Y_1 \end{bmatrix} \begin{bmatrix} I_{n_1} & 0 & \cdots & 0 \\ 0 & (h_T)_2 & \cdots & (b_T)_2 \cdots (b_T)_{N-1} (h_T)_N \end{bmatrix} \\ \hline T(2:N,:) \end{array}\right] \\
&= \left[\begin{array}{c} \begin{bmatrix} (d_S)_1 & (g_S)_1 \\ 0 & Y_1 \end{bmatrix} \begin{bmatrix} I_{n_1} & 0 & \cdots & 0 \\ 0 & (h_T)_2 & \cdots & (b_T)_2 \cdots (b_T)_{N-1} (h_T)_N \end{bmatrix} \\ \hline T(2:N,:) \end{array}\right] \\
&= \left[\begin{array}{cc} (d_S)_1 & (g_S)_1 \left(\begin{array}{ccc} (h_T)_2 & \cdots & (b_T)_2 \cdots (b_T)_{N-1} (h_T)_N \end{array} \right) \\ 0 & Y_1 \left(\begin{array}{ccc} (h_T)_2 & \cdots & (b_T)_2 \cdots (b_T)_{N-1} (h_T)_N \end{array} \right) \\ \hline \multicolumn{2}{c}{T(2:N,:)} \end{array}\right]. \qquad (3.22)
\end{aligned}
$$

Because no modification of the generators $(h_T)_k$, $(b_T)_k$ for $k = 2, \ldots, N$ is required, this step is complete. For consistency in the naming of generators, set $(h_S)_k = (h_T)_k$ and $(b_S)_k = (b_T)_k$ for $k = 2, \ldots, N$. Then (3.22) becomes:

$$
\begin{aligned}
\tilde{U}_1^* T &= \left[\begin{array}{cc}
(d_S)_1 & (g_S)_1 \begin{pmatrix} (h_S)_2 & \cdots & (b_S)_2 \cdots (b_S)_{N-1}(h_S)_N \end{pmatrix} \\
0 & Y_1 \begin{pmatrix} (h_S)_2 & \cdots & (b_S)_2 \cdots (b_S)_{N-1}(h_S)_N \end{pmatrix} \\
\hline
\multicolumn{2}{c}{T(2:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{ccccc}
(d_S)_1 & (g_S)_1(h_S)_2 & (g_S)_1(b_S)_2(h_S)_3 & \cdots & (g_S)_1(b_S)_2 \cdots (b_S)_{N-1}(h_S)_N \\
\hline
0 & Y_1(h_S)_2 & Y_1(b_S)_2(h_S)_3 & \cdots & Y_1(b_S)_2 \cdots (b_S)_{N-1}(h_S)_N \\
0 & (d_S)_2 & (g_S)_2(h_S)_3 & \cdots & (g_S)_2(b_S)_3 \cdots (b_S)_{N-1}(h_S)_N \\
\hline
\multicolumn{5}{c}{T(3:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{ccc}
\multicolumn{3}{c}{S(1,:)} \\
\hline
\begin{matrix} 0 & Y_1(h_S)_2 \\ 0 & (d_S)_2 \end{matrix} & \begin{bmatrix} Y_1(b_S)_2 \\ (g_S)_2 \end{bmatrix} & \begin{bmatrix} (h_S)_3 & (b_S)_3(h_S)_4 & \cdots & (b_S)_3 \cdots (b_S)_{N-1}(h_S)_N \end{bmatrix} \\
\hline
\multicolumn{3}{c}{T(3:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{ccc}
\multicolumn{3}{c}{S(1,:)} \\
\hline
\begin{matrix} 0 \\ 0 \end{matrix} & \begin{bmatrix} Y_1(h_S)_2 & Y_1(b_S)_2 \\ (d_S)_2 & (g_S)_2 \end{bmatrix} & \begin{bmatrix} I_{n_2} & 0 & \cdots & 0 \\ 0 & (h_S)_3 & \cdots & (b_S)_3 \cdots (b_S)_{N-1}(h_S)_N \end{bmatrix} \\
\hline
\multicolumn{3}{c}{T(3:N,:)}
\end{array}\right]. \qquad (3.23)
\end{aligned}
$$

Next, for $i = 2, \ldots, N-1$, compute $s_i = s_{i-1} + \nu_i - n_i$. Then perform QR factorization on

$$
\begin{bmatrix} Y_{i-1}(h_S)_i & Y_{i-1}(b_S)_i \\ (d_S)_i & (g_S)_i \end{bmatrix}
$$

to produce

$$\begin{bmatrix} Y_{i-1}(h_S)_i & Y_{i-1}(b_S)_i \\ (d_S)_i & (g_S)_i \end{bmatrix} = P_i \begin{bmatrix} (d_S)_i & (g_S)_i \\ 0 & Y_i \end{bmatrix}$$
$$= \begin{bmatrix} (h_U)_i & (b_U)_i \\ (d_U)_i & (g_U)_i \end{bmatrix} \begin{bmatrix} (d_S)_i & (g_S)_i \\ 0 & Y_i \end{bmatrix}$$

with dimensions:

$$\begin{aligned} (d_S)_k : &\quad n_k \times n_k \\ (g_S)_i : &\quad n_i \times \rho'_i \\ Y_i : &\quad s_i \times \rho'_i \\ (h_U)_j : &\quad s_{j-1} \times n_j \\ (b_U)_k : &\quad s_{k-1} \times s_k \\ (d_U)_k : &\quad \nu_k \times n_k \\ (g_U)_i : &\quad \nu_i \times s_i. \end{aligned}$$

Let

$$\chi_i = \sum_{k=1}^{i-1} n_k,$$

with ϕ_i defined as before:

$$\phi_i = \sum_{k=i+1}^{N} \nu_k,$$

and let $U_i = I_{\chi_i} \oplus P_i \oplus I_{\phi_i}$. Then

$$
\begin{aligned}
&U_i^* \cdots U_2^* \tilde{U}_1^* T \\
&= \left[\begin{array}{c|c|c} I_{\chi_i} & & \\ \hline & P_i^* & \\ \hline & & I_{\phi_i} \end{array}\right]
\left[\begin{array}{c}
S(1:i-1,:) \\ \hline
\begin{array}{ccc} 0 & \cdots & 0 \\ 0 & \cdots & 0 \end{array}
\begin{bmatrix} Y_{i-1}(h_S)_i & Y_{i-1}(b_S)_i \\ (d_S)_i & (g_S)_i \end{bmatrix}
\begin{bmatrix} I_{n_i} & 0 & \cdots & 0 \\ 0 & (h_S)_{i+1} & \cdots & (b_S)_{i+1}\cdots(h_S)_N \end{bmatrix} \\ \hline
T(i+1:N,:)
\end{array}\right] \\
&= \left[\begin{array}{c|c|c} I_{\chi_i} & & \\ \hline & P_i^* & \\ \hline & & I_{\phi_i} \end{array}\right]
\left[\begin{array}{c}
S(1:i-1,:) \\ \hline
\begin{array}{ccc} 0 & \cdots & 0 \\ 0 & \cdots & 0 \end{array}
\; P_i
\begin{bmatrix} (d_S)_i & (g_S)_i \\ 0 & Y_i \end{bmatrix}
\begin{bmatrix} I_{n_i} & 0 & \cdots & 0 \\ 0 & (h_S)_{i+1} & \cdots & (b_S)_{i+1}\cdots(b_S)_{N-1}(h_S)_N \end{bmatrix} \\ \hline
T(i+1:N,:)
\end{array}\right] \\
&= \left[\begin{array}{c}
S(1:i-1,:) \\ \hline
\begin{array}{ccc} 0 & \cdots & 0 \\ 0 & \cdots & 0 \end{array}
\begin{bmatrix} (d_S)_i & (g_S)_i \\ 0 & Y_i \end{bmatrix}
\begin{bmatrix} I_{n_i} & 0 & & \cdots & 0 \\ 0 & (h_S)_{i+1} & (b_S)_{i+1}(h_S)_{i+2} & \cdots & (b_S)_{i+1}\cdots(b_S)_{N-1}(h_S)_N \end{bmatrix} \\ \hline
T(i+1:N,:)
\end{array}\right] \\
&= \left[\begin{array}{c}
S(1:i-1,:) \\ \hline
\begin{array}{cccccl} 0 & \cdots & 0 & (d_S)_i & (g_S)_i & \begin{pmatrix} (h_S)_{i+1} & (b_S)_{i+1}(h_S)_{i+2} & \cdots & (b_S)_{i+1}\cdots(b_S)_{N-1}(h_S)_N \end{pmatrix} \\ 0 & \cdots & 0 & 0 & Y_i & \begin{pmatrix} (h_S)_{i+1} & (b_S)_{i+1}(h_S)_{i+2} & \cdots & (b_S)_{i+1}\cdots(b_S)_{N-1}(h_S)_N \end{pmatrix} \end{array} \\ \hline
T(i+1:N,:)
\end{array}\right] \\
&= \left[\begin{array}{cccccccc}
\multicolumn{8}{c}{S(1:i-1,:)} \\
0 & \cdots & 0 & (d_S)_i & (g_S)_i(h_S)_{i+1} & (g_S)_i(b_S)_{i+1}(h_S)_{i+2} & \cdots & (g_S)_i(b_S)_{i+1}\cdots(b_S)_{N-1}(h_S)_N \\ \hline
0 & \cdots & 0 & 0 & Y_i(h_S)_{i+1} & Y_i(b_S)_{i+1}(h_S)_{i+2} & \cdots & Y_i(b_S)_{i+1}\cdots(b_S)_{N-1}(h_S)_N \\
0 & \cdots & 0 & 0 & (d_S)_{i+1} & (g_S)_{i+1}(h_S)_{i+2} & \cdots & (g_S)_{i+1}(b_S)_{i+2}\cdots(b_S)_{N-1}(h_S)_N \\ \hline
\multicolumn{8}{c}{T(i+2:N,:)}
\end{array}\right] \\
&= \left[\begin{array}{c}
S(1:i,:) \\ \hline
0
\begin{bmatrix} Y_i(h_S)_{i+1} & Y_i(b_S)_{i+1} \\ (d_S)_{i+1} & (g_S)_{i+1} \end{bmatrix}
\begin{bmatrix} I_{n_i} & 0 & \cdots & 0 \\ 0 & (h_S)_{i+2} & \cdots & (b_S)_{i+2}\cdots(b_S)_{N-1}(h_S)_N \end{bmatrix} \\ \hline
T(i+2:N,:)
\end{array}\right].
\end{aligned}
\tag{3.24}
$$

Note that in the case of $i = N-2$:

$$U_{N-2}^* \cdots U_2^* \tilde{U}_1^* T \quad = \quad \left[\begin{array}{ccc} \multicolumn{3}{c}{S(1:N-2,:)} \\ \hline 0 & \begin{bmatrix} Y_{N-2}(h_S)_{N-1} & Y_{N-2}(b_S)_{N-1} \\ (d_S)_{N-1} & (g_S)_{N-1} \end{bmatrix} & \begin{bmatrix} I_{n_{N-2}} & 0 \\ 0 & (h_S)_N \end{bmatrix} \\ \hline \multicolumn{3}{c}{T(N,:)} \end{array}\right],$$

and in the case of $i = N-1$:

$$\begin{aligned} U_{N-1}^* \cdots U_2^* \tilde{U}_1^* T \quad &= \quad \left[\begin{array}{ccc} \multicolumn{3}{c}{S(1:N-1,:)} \\ \hline 0 & \begin{bmatrix} Y_{N-1}(h_S)_N & Y_{N-1}(b_S)_N \\ (d_S)_N & (g_S)_N \end{bmatrix} & \begin{bmatrix} I_{n_{N-2}} \\ 0 \end{bmatrix} \end{array}\right] \\ &= \quad \left[\begin{array}{cccc} \multicolumn{4}{c}{S(1:N-1,:)} \\ \hline 0 & \cdots & 0 & \begin{bmatrix} Y_{N-1}(h_S)_N \\ (d_S)_N \end{bmatrix} \\ 0 & \cdots & 0 & \end{array}\right]. \end{aligned}$$

In the final step of inner-outer factorization, perform QR factorization on

$$\begin{bmatrix} Y_{N-1}(h_S)_N \\ (d_S)_N \end{bmatrix}$$

to produce

$$\begin{bmatrix} Y_{N-1}(h_S)_N \\ (d_S)_N \end{bmatrix} = P_N(d_S)_N = \begin{bmatrix} (h_U)_N \\ (d_U)_N \end{bmatrix} (d_S)_N$$

with dimensions:

$$\begin{aligned} (d_S)_N : &\quad n_N \times n_N \\ (h_U)_N : &\quad s_{N-1} \times n_N \\ (d_U)_N : &\quad \nu_N \times n_N. \end{aligned}$$

Let

$$\chi_N = \sum_{k=1}^{N-1} n_k,$$

and let $\tilde{U}_N = I_{\chi_N} \oplus P_N$. Then

$$\begin{aligned}
S &= \tilde{U}_N^* U_{N-1}^* \cdots U_2^* \tilde{U}_1^* T \\
&= \left[\begin{array}{c|c} I_{\chi_N} & \\ \hline & P_N^* \end{array}\right] \left[\begin{array}{cccc} \multicolumn{4}{c}{S(1:N-1,:)} \\ \hline 0 & \cdots & 0 & \begin{bmatrix} Y_{N-1}(h_S)_N \\ (d_S)_N \end{bmatrix} \end{array}\right] \\
&= \left[\begin{array}{c|c} I_{\chi_N} & \\ \hline & P_N^* \end{array}\right] \left[\begin{array}{cccc} \multicolumn{4}{c}{S(1:N-1,:)} \\ \hline 0 & \cdots & 0 & P_N(d_S)_N \end{array}\right] \\
&= \left[\begin{array}{cccc} \multicolumn{4}{c}{S(1:N-1,:)} \\ \hline 0 & \cdots & 0 & (d_S)_N \end{array}\right] \\
&= \begin{bmatrix} (d_S)_1 & (g_S)_1(h_S)_2 & (g_S)_1(b_S)_2(h_S)_3 & \cdots & (g_S)_1(b_S)_2\cdots(b_S)_{N-1}(h_S)_N \\ 0 & (d_S)_2 & (g_S)_2(h_S)_3 & \cdots & (g_S)_2(b_S)_3\cdots(b_S)_{N-1}(h_S)_N \\ \vdots & & \ddots & \ddots & \vdots \\ & & \ddots & (d_S)_{N-1} & (g_S)_{N-1}(h_S)_N \\ 0 & & \cdots & 0 & (d_S)_N \end{bmatrix} \qquad (3.25)
\end{aligned}$$

This is precisely what was sought:

$$\tilde{U}_N^* U_{N-1}^* \cdots U_2^* \tilde{U}_1^* T = S$$

so

$$\begin{aligned}
T &= \tilde{U}_1 U_2 \cdots U_{N-1} U_N S \\
&= US
\end{aligned}$$

where U is a block upper triangular unitary matrix and S is a block upper triangular invertible matrix with invertible square blocks on the diagonal.

Chapter 4

Findings

One important claim made by Eidelman and Gohberg is that the need for minimality stated in [1] is no longer relevant in their algorithm: "It allows us to avoid the requirement of the minimality of generators..." [2, p. 421]. But tests of their algorithm, in the solution of linear systems $R\mathbf{x} = \mathbf{y}$, were performed on generators created from random numbers, resulting, essentially as a given, in minimal generators every time. However, non-minimal generators or nearly non-minimal generators can lead to significant residuals, as will be demonstrated in Section 4.2.

To verify the efficacy of the program `quasifactor.m` and the functions it calls, several tests were performed using generators of various sizes with randomly generated elements. The objective is to solve the system $R\mathbf{x} = \mathbf{y}$ and to observe residuals in various cases of generators with particular characteristics. Initially, quasiseparable matrices were constructed in a way intended to mimic that described in [2]. Then, in order to investigate stability of the algorithm in the case of non-minimal and nearly non-minimal generators, it was decided to reduce complexity and to perform all tests after establishing matrix R as block-lower Hessenberg (this simply requires making all generators b_k zero). Also in the interest of

N	$\max(r')$	$\text{cond}(R)$	max. (relative) residual
20	2	10^4	10^{-17}
20	3	10^6	10^{-16}
40	2	10^4	10^{-16}
40	3	10^{10}	10^{-16}
80	2	10^5	10^{-16}
80	3	10^{17}	10^{-16}
500	2	10^8	10^{-16}

Table 4.1: Results with minimal generators, all elements in $[0, 1)$

reducing complexity, it was decided that $\mathbf{y}$ would be created by multiplying R by a column vector, the nominal $\mathbf{x}$, of the appropriate length and consisting of all 1's.

4.1 Generic case: minimal generators

First, the algorithm is performed on quasiseparable matrices of various sizes, comprised entirely of generators d_k, p_i, q_j, a_k, g_i, h_j, and b_k of size 2×2 with all elements randomly selected from $[0, 1)$.

It was found that the relative residual had a value on the order of 10^{-16} or smaller, for experiments on quasiseparable matrices R up to dimension 1000×1000 ($N = 500$ and $r' = 2$). Some typical results are listed in Table 4.1. Clearly, the algorithm performs with minimal error for cases involving minimal (randomly generated) generators.

This result confirms the findings of [2], specifically that the algorithm is stable, but in a setting that involves generators that are very unlikely to be non-minimal, or even near non-minimal.

Next, to confirm that the results hold for generators containing elements outside of the right half of the unit circle, i.e. for z that is an element in any generator d_k, p_i, q_j, a_k, g_i, h_j, and b_k such that $|z| > 1$ or $Re(z) < 0$, the previous experiment is repeated, except that the elements are randomly selected from the interval $[-10, 10)$.

N	$\max(r')$	$\mathrm{cond}(R)$	max. (relative) residual
20	2	10^{17}	10^{-17}
20	3	10^{20}	10^{-19}
40	2	10^{34}	10^{-19}
40	3	10^{40}	10^{-18}

Table 4.2: Results with minimal generators, all elements in $[-10, 10)$

Relative residuals similar to those in Table 4.1 were found, demonstrating that the algorithm works well for a variety of matrices formed from minimal generators, regardless of the size of the elements of the generators. The only notable difference was that the matrix R was generally less well-conditioned, which is not surprising considering the larger values of its elements, compared to the previous experiment. In fact, the matrix became so ill-conditioned that beyond size 120×120, the invertible factor, S, contained many generators $(d_S)_k$, that were singular to machine precision, rendering any results meaningless. The results through $N = 40$, $r'_k = 3, k = 2, \ldots, N-1$ are shown in Table 4.2.

4.2 Non-minimal generators

To shed more light on the effect of minimality, some nearly non-minimal generators are now selected based on classical linear systems theory. The idea is that a generator associated with an uncontrollable mode is not minimal [5], and that errors may be introduced in each multiplication by such a generator. To confound the numerical processes of the algorithm, a non-minimal system is established (with an uncontrollable mode) and then transformed by a similarity matrix to remove the computational benefit of multiplying by zero (which is an exact operation in floating point arithmetic). The non-minimal system is then modified to a nearly non-minimal system. Residuals in the non-minimal and nearly non-minimal cases are recorded and tabulated in Table 4.3.

The first example to confound the algorithm is constructed of 2×2 blocks, with all generators of size 2×2, and N (the number of block rows and block columns) $= 20$. The lower generators are selected such that they fail to meet the minimality condition. The generators

$$p' = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad a' = \begin{bmatrix} 3.3 & 0 \\ 0 & 0.9 \end{bmatrix}, \qquad \text{and} \qquad q' = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$$

are modified by a similarity transformation, via an arbitrary 2×2 matrix:

$$S = \begin{bmatrix} .6 & .88 \\ -.4 & .7 \end{bmatrix}$$

to produce

$$\begin{aligned} p_i &= p'S && \text{for} \quad i = 2, \ldots, n, \\ a_k &= S^{-1}a'S && \text{for} \quad k = 2, \ldots, n-1, \qquad \text{and} \\ q_j &= S^{-1}q && \text{for} \quad j = 1, \ldots, n-1. \end{aligned}$$

To minimize the number of variables in play, let

$$\begin{aligned} d_k &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, && k = 1, \ldots, N \\ g_i &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, && i = 1, \ldots, N-1 \\ h_j &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, && j = 2, \ldots, N \\ b_k &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, && k = 2, \ldots, N-1. \end{aligned}$$

The results of factoring and solving $R\mathbf{x} = \mathbf{y}$ are summarized in the first line of Table 4.3. The next four lines document the results of making small changes to the values in a'. Because

$$a' = \begin{bmatrix} 4 & 0 \\ 0 & .92 \end{bmatrix}$$

produced the largest relative residuals, it was used in every subsequent investigation.

To broaden the investigation, it is worth considering two modifications. First, allow the matrix to grow, i.e. consider larger values of N. Also, consider the somewhat more realistic possiblity, from imperfectly measured data that are based on a non-minimal system (which is not uncommon in applications of linear systems). Simply modify the generator q' by a small perturbation. (For simplicity, just perturb one zero element that was causing the system to be minimal.) Let

$$q' = \begin{bmatrix} 0 & \delta \\ 1 & 1 \end{bmatrix}$$

for various values of δ. These changes are implemented, and their results are summarized in the lower portion of Table 4.3. It should be noted that in the case of $N = 40$ and $\delta = 10^{-4}$, the factorization produced generators in S that were singular to machine precision, invalidating results. This seems to be related to a very high condition number of R, in this case 2×10^{21}. The same was true for $N = 80$ and every value of δ attempted. These trials produced condition number of R greater than 10^{32} in each case.

Some interesting results were found. Clearly, non-minimal and nearly non-minimal generators produce a matrix R whose factorization can result in relative residuals (in the solution of $R\mathbf{x} = \mathbf{y}$) significantly larger than the machine precision. This contradicts a key claim of [2]. It appears that minimality may be a necessary condition to guarantee the stability of the algorithm. The most surprising result was the alarming discrepancy between the relative residuals in the case of $N = 40$ after a' was perturbed from the non-minimal case ($\delta = 0$) to

N	a'_{11}	a'_{22}	δ	cond(R)	max. residual
20	3.3	0.9	0	3×10^8	2×10^{-9}
20	3.84	0.92	0	1×10^8	4×10^{-8}
20	4	0.9	0	5×10^6	4×10^{-8}
20	4	0.92	0	5×10^7	8×10^{-8}
20	4	0.95	0	2×10^7	8×10^{-8}
20	4	0.92	10^{-16}	8×10^6	3×10^{-8}
20	4	0.92	10^{-12}	9×10^2	8×10^{-8}
20	4	0.92	10^{-8}	1×10^4	2×10^{-9}
20	4	0.92	10^{-4}	1×10^8	2×10^{-13}
40	4	0.92	0	8×10^7	6×10^{-16}
40	4	0.92	10^{-16}	3×10^8	5×10^{-2}
40	4	0.92	10^{-12}	3×10^{12}	7×10^{-6}
40	4	0.92	10^{-8}	3×10^{16}	1×10^{-9}

Table 4.3: Results with non-minimal generators

a very near nonminmal case (by changing δ to 10^{-16}). The change in the relative residual from less than 10^{-15} to more than 10^{-2} was dramatic to say the least. This single instance may be particularly informative in gaining a deeper understanding of how computational errors arise and propagate in the factoring process.

How the algorithm may be modified to provide stability even in the case of non-minimal generators is a pressing question that warrants further investigation. The first step in this task is to quantify precisely how computational errors are propagated in the factoring process. It may be necessary to convert non-minimal generators into minimal ones before performing any factoring, as suggested in [1], or it may be possible to quantify and minimize error propagation by modifying the algorithm of [2] without having to alter the generators. Significant research is still needed.

Bibliography

[1] P. M. Dewilde and A. J. van der Veen, *Time-Varying Systems and Computations*, Kluwer Academic Publishers, New York, 1998.

[2] Y. Eidelman and I. Gohberg, *A modification of the Dewilde- van der Veen method for inversion of finite structured matrices*, Linear Algebra and its Applications, 343-344 (2002), pp. 419-450.

[3] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, second ed., 1989.

[4] W. B. Gragg, *The QR algorithm for unitary Hessenberg matrices*, Journal of Computational and Applied Mathematics, 16 (1986), pp. 1–8. Cited in [3].

[5] T. Kailath, *Linear Systems*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1980.

Appendix A

Programs and Functions

1. quasifactor.m

   This program:

   (a) defines the generators of the quasiseparable matrix, R.

   (b) creates the matrix R from its generators.

   (c) performs inner coprime factorization of R such that $R = VT$ where V is unitary block lower triangular and T is block upper triangular.

   (d) performs inner-outer factorization of T such that $T = US$ where U is unitary block upper triangular and S is block upper triangular invertible (and such that $R = VT = VUS$).

   (e) efficiently solves the system $R\mathbf{x} = \mathbf{y}$ by solving $S\mathbf{x} = V^*U^*\mathbf{y}$.

   If the variable "verify" is set to 1, then the program calculates V, T, U, and S explicitly This is not necessary, as their generators are sufficient for all relevant computation.

2. lower.m

   This function efficiently ($O(n^2)$) multiplies to create the lower part of a quasiseparable matrix from its generators: p, a, q, with dimensions $m(i), n(j), r'(k)$ (rprime).

3. upper.m

   This function efficiently ($O(n^2)$) multiplies to create the upper part of a quasiseparable matrix from its generators: g, b, h, with dimensions $m(i), n(j), r''(k)$ (rdprime).

4. diagonal.m

   This function creates a block-diagonal matrix from given generators, d with dimensions $m(i), n(j)$, essentially just placing the generators in the appropriate positions.

5. innercoprime.m

   This function performs inner coprime factorization of R such that $R = VT$ where V is unitary block lower triangular and T is block upper triangular. The primary mechanism of this algorithm relies on QR factorization.

6. innerouter.m

   This function performs inner-outer factorization of T such that $T = US$ where U is block upper triangular unitary, and S is block upper triangular invertible. This process also relies primarily on QR factorization.

7. lowermult.m

   This function efficiently ($O(n)$) multiplies a strict lower block quasiseparable matrix times a column vector.

8. uppermult.m

   This function efficiently ($O(n)$) multiplies a strict upper block quasiseparable matrix times a column vector.

9. dmult.m

   This function efficiently ($O(n)$) multiplies a block diagonal matrix times a column vector.

(Note that any quasiseparable matrix may be multiplied by a column vector by decomposing it into a strict lower, a strict upper and a diagonal part, multiplying each by the given vector, and then summing the products.)

10. backsolve.m

    This function efficiently solves the system $R\mathbf{x} = \mathbf{y}$ using the generators of V, U, S found by performing inner coprime factorization and inner-outer factorization on R.

Appendix B

Generator Dimension Quick-Reference

1. Dimensions of generators of R, with block entries of size $m_i \times n_j$:

Generator	Dimensions
d_k	$m_k \times n_k$
p_i	$m_i \times r'_{i-1}$
q_j	$r'_j \times n_j$
a_k	$r'_k \times r'_{k-1}$
g_i	$m_i \times r''_i$
h_j	$r''_{j-1} \times n_j$
b_k	$r''_{k-1} \times r''_k$

2. Dimensions of generators of V, with block entries of size $m_i \times \nu_j$:

Generator	Dimensions
$(d_V)_k$	$m_k \times \nu_k$
$(p_V)_i$	$m_i \times \rho_{i-1}$
$(q_V)_j$	$\rho_j \times \nu_j$
$(a_V)_k$	$\rho_k \times \rho_{k-1}$
X_i	$\rho_{i-1} \times r'_{i-1}$

where

$$\begin{aligned} \rho_{N-1} &= \min(m_N, r'_{N-1}) \\ \rho_{k-1} &= \min(m_k + \rho_k, r'_{k-1}), \qquad k = N-1, \ldots, 2 \\ \nu_N &= m_N - \rho_{N-1} \\ \nu_k &= m_k + \rho_k - \rho_{k-1}, \qquad k = N-1, \ldots, 2. \end{aligned}$$

3. Dimensions of generators of T, with block entries of size $\nu_i \times n_j$:

Generator	Dimensions
$(d_T)_k$	$\nu_k \times n_k$
$(g_T)_i$	$\nu_i \times \rho'_i$
$(h_T)_j$	$\rho'_{j-1} \times n_j$
$(b_T)_k$	$\rho'_{k-1} \times \rho'_k$

with ρ_{k-1} and ν_k defined as above for $k = N, \ldots, 2$, and $\rho'_k = \rho_k + r''_k$ for $k = N-1, \ldots, 2$.

4. Dimensions of generators of U, with block entries of size $\nu_i \times n_j$:

Generator	Dimensions
$(d_U)_k$	$\nu_k \times n_k$
$(g_U)_i$	$\nu_i \times s_i$
$(h_U)_j$	$s_{j-1} \times n_j$
$(b_U)_k$	$s_{k-1} \times s_k$

where

$$\begin{aligned} s_1 &= \nu_1 - n_1 \\ s_k &= s_{k-1} + \nu_k - n_k, \qquad k = 2, \ldots, N-1. \end{aligned}$$

5. Dimensions of generators of S, with block entries of size $n_i \times n_j$:

Generator	Dimensions
$(d_S)_k$	$n_k \times n_k$
$(g_S)_i$	$n_i \times \rho'_i$
$(h_S)_j$	$\rho'_{j-1} \times n_j$
$(b_S)_k$	$\rho'_{k-1} \times \rho'_k$.