Computing with quasiseparable matrices
Clément Pernet

To cite this version:

HAL Id: hal-01264131
https://hal.science/hal-01264131v2
Submitted on 19 Sep 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Computing with Quasiseparable Matrices

Clément Pernet
Univ. Grenoble Alpes Laboratoire LIP Inria, Université de Lyon
46, Allée d’Italie, F69364 Lyon Cedex 07, France
Clement.Pernet@imag.fr

ABSTRACT
The class of quasiseparable matrices is defined by a pair of bounds, called the quasiseparable orders, on the ranks of the sub-matrices entirely located in their strictly lower and upper triangular parts. These arise naturally in applications, as e.g. the inverse of band matrices, and are widely used for they admit structured representations allowing to compute with them in time linear in the dimension. We show, in this paper, the connection between the notion of quasiseparability and the rank profile matrix invariant, presented in [Dumas & al. ISSAC’15]. This allows us to propose an algorithm computing the quasiseparable orders \((r_L, r_U)\) in time \(O(n^2s^{-2})\) where \(s = \max(r_L, r_U)\) and \(\omega\) the exponent of matrix multiplication. We then present two new structured representations, a binary tree of PLUQ decompositions, and the Bruhat generator, using respectively new structured representations, a binary tree of PLUQ decompositions, and the classical generator and \(O(ns\log n)\) field elements instead of \(O(ns^2\omega)\) for the classical generator and \(O(ns\log n)\) for the hierarchically semiseparable representations. We present algorithms computing these representations in time \(O(n^2s^{-2})\). These representations allow a matrix-vector product in time linear in the size of their representation. Lastly we show how to multiply two such structured matrices in time \(O(n^2s^{\omega-2})\).

1. INTRODUCTION
The inverse of a tridiagonal matrix, when it exists, is a dense matrix with the property that all sub-matrices entirely below or above its diagonal have rank at most one. This property and many generalizations of it, defining the semiseparable and quasiseparable matrices, have been extensively studied over the past 80 years. We refer to [16] and [17] for a broad bibliographic overview on the topic. In this paper, we will focus on the class of quasiseparable matrices, introduced in [8]:

Definition 1. An \(n \times n\) matrix \(M\) is \((r_L, r_U)\)-quasiseparable if its strictly lower and upper triangular parts satisfy the following low rank structure: for all \(1 \leq k \leq n - 1\),

\[
\text{rank}(M_{k+1\ldots n\ldots k}) \leq r_L,
\]

\[
\text{rank}(M_{1\ldots k\ldots k+1\ldots n}) \leq r_U.
\]

The values \(r_L\) and \(r_U\) define the quasiseparable orders of \(M\).

Quasiseparable matrices can be represented with fewer than \(n^2\) coefficients, using a structured representation, called a generator. The most commonly used generator [8, 16, 17, 9, 4] for a matrix \(M\), consists of \((n-1)\) pairs of vectors \(p(i), q(i)\) of size \(r_L\), \((n-1)\) pairs of vectors \(g(i), h(i)\) of size \(r_U\), \(n-1\) matrices \(a(i)\) of dimension \(r_L \times r_U\), and \(n-1\) matrices \(b(i)\) of dimension \(r_U \times r_U\) such that

\[
M_{i,j} = \begin{cases} 
    p(i)^T a_{ij} q(j), & 1 \leq j < i \leq n \\
    d(i), & 1 \leq i = j \leq n \\
    g(i)^T b_{ij} h(j), & 1 \leq i < j \leq n
\end{cases}
\]

where \(a_{ij} = a(i - 1)\ldots a(j + 1)\) for \(j > i + 1\), \(a_{j+1,j} = 1\), and \(b_{ij} = b(i + 1)\ldots b(i - 1)\) for \(i > j + 1\), \(b_{i,i+1} = 1\). This representation, of size \(O(n(r_L^2 + r_U^2))\) makes it possible to apply a vector in \(O(n(r_L^2 + r_U^2))\) field operations, multiply two quasiseparable matrices in time \(O(n\max(r_L, r_U)^3)\) and also compute the inverse in time \(O(n\max(r_L, r_U)^3)\) [8].

The contribution of this paper, is to make the connection between the notion of quasiseparability and a matrix invariant, the rank profile matrix, that we introduced in [6]. More precisely, we show that the PLUQ decompositions of the lower and upper triangular parts of a quasiseparable matrix, using a certain class of pivoting strategies, also have a structure ensuring that their memory footprint and the time complexity to compute them does not depend on the rank of the matrix but on the quasiseparable order (which can be arbitrarily lower). Note that we will assume throughout the paper that the PLUQ decomposition algorithms mentioned have the ability to reveal ranks. This is the case when computing with exact arithmetic (e.g., finite fields or multprecision rationals), but not always with finite precision floating point arithmetic. In the latter context, a special care need to be taken for the pivoting of LU decompositions [10, 14], and QR or SVD decompositions are often more commonly used [2, 3]. This study is motivated by the design of new algorithms on polynomial matrices where quasiseparable matrices naturally occur, and more generally by the framework of the LinBox library [15] for black-box exact linear algebra.

After defining and recalling the properties of the rank profile matrix in Section 2, we propose in Section 3 an algorithm computing the quasiseparable orders \((r_L, r_U)\) in time \(O(n^2s^{\omega-2})\) where \(s = \max(r_L, r_U)\) and \(\omega\) the exponent of

Publication rights licensed to ACM. ACM acknowledges that this contribution was authored or co-authored by an employee, contractor or affiliate of a national government. As such, the Government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for Government purposes only.

ISSAC ’16, July 19 - 22, 2016, Waterloo, ON, Canada
ACM ISBN 978-1-4503-4380-0/16/07 . . . $15.00
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
DOI: http://dx.doi.org/10.1145/2930889.2930915
2. PRELIMINARIES

2.1 Left triangular matrices

We will make intensive use of matrices with non-zero elements only located above the main anti-diagonal. We will refer to these matrices as left triangular, to avoid any confusion with upper triangular matrices.

Definition 2. A left triangular matrix is any \( m \times n \) matrix \( A \) such that \( A_{i,j} = 0 \) for all \( i > n - j \).

The left triangular part of a matrix \( A \), denoted by \( \text{Left}(A) \), will refer to the left triangular matrix extracted from it. We will need the following property on the left triangular part of the product of a matrix by a triangular matrix.

**Lemma 1.** Let \( A = BU \) be an \( m \times n \) matrix where \( U \) is \( n \times n \) upper triangular. Then \( \text{Left}(A) = \text{Left}(\text{Left}(B)U) \).

**Proof.** Let \( \bar{A} = \text{Left}(A), \bar{B} = \text{Left}(B) \). For \( j \leq n - i \), we have \( \bar{A}_{i,j} = \sum_{k=1}^{n} B_{i,k} \cdot U_{k,j} = \sum_{k=1}^{j} B_{i,k} \cdot U_{k,j} \) as \( U \) is upper triangular. Now for \( k \leq j \leq n - i \), \( B_{i,k} = \bar{B}_{i,k} \), which proves that the left triangular part of \( A \) is that of \( \text{Left}(B)U \).

Applying Lemma 1 on \( A^T \) yields Lemma 2

**Lemma 2.** Let \( A = LB \) be an \( m \times n \) matrix where \( L \) is \( m \times m \) lower triangular. Then \( \text{Left}(A) = \text{Left}(L \text{Left}(B)) \).

Lastly, we will extend the notion of quasiseparable order to left triangular matrices, in the natural way: the left quasiseparable order is the maximal rank of any leading \( k \times (n - k) \) sub-matrix. When no confusion may occur, we will abuse the definition and simply call it the quasiseparable order.

2.2 The rank profile matrix

We will use a matrix invariant, introduced in [6, Theorem 1], that summarizes the information on the ranks of any leading sub-matrices of a given input matrix.

**Definition 3.** [6, Theorem 1] The rank profile matrix of an \( m \times n \) matrix \( A \) of rank \( r \) is the unique \( m \times n \) matrix \( R_A \), with only \( r \) non-zero coefficients, all equal to one, located on distinct rows and columns such that any leading sub-matrices of \( R_A \) has the same rank as the corresponding leading sub-matrix in \( A \).

This invariant can be computed in just one Gaussian elimination of the matrix \( A \), at the cost of \( O(mnr^2) \) field operations [6], provided some conditions on the pivoting strategy being used. It is obtained from the corresponding PLUQ decomposition as the product

\[
R_A = P \begin{bmatrix} I_r & 0_{(m-r)\times(n-r)} \end{bmatrix} Q.
\]

We also recall in Theorem 1 an important property of such PLUQ decompositions revealing the rank profile matrix.

**Theorem 1** ([7, Th. 24], [5, Th. 1]). Let \( A = PLUQ \) be a PLUQ decomposition revealing the rank profile matrix of \( A \). Then, \( P [ L \ 0_{m \times (m-r)} ] PT \) is lower triangular and \( Q^T \begin{bmatrix} U \ 0_{(n-r)\times n} \end{bmatrix} Q \) is upper triangular.

**Lemma 3.** The rank profile matrix invariant is preserved by multiplication

1. to the left with an invertible lower triangular matrix,
2. to the right with an invertible upper triangular matrix.

**Proof.** Let \( B = LA \) for an invertible lower triangular matrix \( L \). Then \( \text{rank}(B_{1,1}) = \text{rank}(L_{1,1}A_{1,1}) = \text{rank}(A_{1,1}) \) for any \( i \leq m, j \leq n \). Hence \( R_B = R_A \).

3. COMPUTING THE QUASISEPARABLE ORDERS

Let \( M \) be an \( n \times n \) matrix of which one want to determine the quasiseparable orders \((r_L, r_U)\). Let \( L \) and \( U \) be respectively the lower triangular part and the upper triangular part of \( M \).

Let \( J_0 \) be the unit anti-diagonal matrix. Multiplying on the left by \( J_0 \) reverts the row order while multiplying on the right by \( J_0 \) reverts the column order. Hence both \( J_0L \) and \( UJ_0 \) are left triangular matrices. Remark that the conditions (1) and (2) state that all leading \( k \times (n - k) \) sub-matrices of \( J_0L \) and \( UJ_0 \) have rank no greater than \( r_L \) and \( r_U \) respectively. We will then use the rank profile matrix of these two left triangular matrices to find these parameters.

3.1 From a rank profile matrix

First, note that the rank profile matrix of a left triangular matrix is not necessarily left triangular. For example, the rank profile matrix of \( \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \) is \( \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \). However, only the left triangular part of the rank profile matrix is sufficient to compute the left quasiseparable orders.

Suppose for the moment that the left-triangular part of the rank profile matrix of a left triangular matrix is given (returned by a function LT-RPM). It remains to enumerate all leading \( k \times (n - k) \) sub-matrices and find the one with the largest number of non-zero elements. Algorithm 1 shows how to compute the largest rank of all leading sub-matrices of such a matrix. Run on \( J_0L \) and \( UJ_0 \), it returns successively the quasiseparable orders \( r_L \) and \( r_U \).

This algorithm runs in \( O(n) \) provided that the rank profile matrix \( R \) is stored in a compact way, e.g. using a vector of \( r \) pairs of pivot indices \([(i_1, j_1), \ldots, (i_r, j_r)]\).

3.2 Computing the rank profile matrix of a left triangular matrix

We now deal with the missing component: computing the left triangular part of the rank profile matrix of a left triangular matrix.
Algorithm 1 QS-order

Input: A, an n × n matrix
Output: \( \max(\text{rank}(A_{1,k,k+1:n-k}) : 1 \leq k \leq n-1) \)

\[
R \leftarrow \text{LT-RPM}(A) : \text{The left triangular part of the rank profile matrix of } A
\]

\[
\text{rows} \leftarrow (\text{False}, \ldots, \text{False})
\]

\[
\text{cols} \leftarrow (\text{False}, \ldots, \text{False})
\]

\[
\text{for all } (i, j) \text{ such that } R_{i,j} = 1 \text{ do}
\]

\[
\text{rows}[i] \leftarrow \text{True}
\]

\[
\text{cols}[j] \leftarrow \text{True}
\]

end for

\[
s, r \leftarrow 0
\]

for \( i = 1 \) to \( n \) do

\[
\text{if } \text{rows}[i] \text{ then } r \leftarrow r + 1
\]

\[
\text{if } \text{cols}[n-i+1] \text{ then } r \leftarrow r - 1
\]

\[
s \leftarrow \text{max}(s, r)
\]

end for

return \( s \)

3.2.1 From a PLUQ decomposition

A first approach is to run any Gaussian elimination algorithm that can reveal the rank profile matrix, as described in [6]. In particular, the PLUQ decomposition algorithm of [5] computes the rank profile matrix of \( A \) in \( O(n^2r^2) \) where \( r = \text{rank}(A) \). However this estimate is pessimistic as it does not take into account the left triangular shape of the matrix. Moreover, this estimate does not depend on the left quasiseparable order \( s \) but on the rank \( r \), which may be much higher.

Remark 1. The discrepancy between the rank \( r \) of a left triangular matrix and its quasiseparable order arises from the location of the pivots in its rank profile matrix. Pivots located near the top left corner of the matrix are shared by many leading sub-matrices, and are therefore likely to contribute to the quasiseparable order. On the other hand, pivots near the anti-diagonal can be numerous, but do not add up to a large quasiseparable order. As an illustration, consider the two following extreme cases:

1. a matrix \( A \) with generic rank profile. Then the leading \( r \times r \) sub-matrix of \( A \) has rank \( r \) and the quasiseparable order is \( s \approx r \).

2. the matrix with \( n-1 \) ones right above the anti-diagonal. It has rank \( r = n-1 \) but quasiseparable order 1.

Remark 1 indicates that in the unlucky cases when \( r \gg s \), the computation should reduce to instances of smaller sizes, hence a trade-off should exist between, on one hand, the discrepancy between \( r \) and \( s \), and on the other hand, the dimension \( n \) of the problems. All contributions presented in the remaining of the paper are based on such trade-offs.

3.2.2 A dedicated algorithm

In order to reach a complexity depending on \( s \) and not \( r \), we adapt in Algorithm 2 the tile recursive algorithm of [5], so that the left triangular structure of the input matrix is preserved and can be used to reduce the amount of computation.

Algorithm 2 does not assume that the input matrix is left triangular, as it will be called recursively with arbitrary matrices, but guarantees to return the left triangular part of the rank profile matrix. While the top left quadrant \( A_1 \) is eliminated using any PLUQ decomposition algorithm revealing the rank profile matrix, the top right and bottom left quadrants are handled recursively.

Theorem 2. Given an \( n \times n \) input matrix \( A \) with left quasiseparable order \( s \), Algorithm 2 computes the left triangular part of the rank profile matrix of \( A \) in \( O(n^2s^2) \).

Proof. First remark that

\[
P_1 \begin{bmatrix} D & F \\ \end{bmatrix} = P_1 \begin{bmatrix} L_{n-1}^{-1} & I_{n-1} \\ -M_1L_1^{-1} & L_{n-1} \\ \end{bmatrix} P_1^n P_1 \begin{bmatrix} B_1 \\ B_2 \\ \end{bmatrix} = L A_2.
\]

Hence

\[
L \begin{bmatrix} A_1 \\ A_2 \\ \end{bmatrix} = P_1 \begin{bmatrix} [U_1 \ V_1] Q_1 D \\ 0 \ F \\ \end{bmatrix}.
\]

From Theorem 1, the matrix \( L \) is lower triangular and by Lemma 3 the rank profile matrix of \( [A_1 A_2] \) equals that of \( P_1 \begin{bmatrix} [U_1 \ V_1] Q_1 D \\ 0 \ F \\ \end{bmatrix} \). Now as \( U_1 \) is upper triangular and non-singular, this rank profile matrix is in turn that of \( P_1 \begin{bmatrix} [U_1 \ V_1] Q_1 D \\ 0 \ F \\ \end{bmatrix} \) and its left triangular part is \( [R_1 \ R_2] \).

By a similar reasoning, \( [R_1 \ R_3]^T \) is the left triangular part of the rank profile matrix of \( [A_1 A_3]^T \), which shows that the algorithm is correct.

Let \( s_1 \) be the left quasiseparable order of \( H \) and \( s_2 \) that of \( I \). The number of field operations to run Algorithm 2 is

\[
T(n, s) = \text{or}_{s_1}^{n^2} n^2 + T_{\text{LT-RPM}}(n/2, s_1) + T_{\text{LT-RPM}}(n/2, s_2)
\]
for a positive constant $\alpha$. We will prove by induction that $T(n, s) \leq 2\alpha s^{2-s}n^2$.

Again, since $L$ is lower triangular, the rank profile matrix of $L A_2$ is that of $A_2$ and the quasiseparable orders of the two matrices are the same. Now $H$ is the matrix $L A_2$ with some rows zeroed out, hence $s_1$, the quasiseparable order of $H$ is no greater than that of $A_2$ which is less or equal to $s$. Hence $\max(r_1, s_1, s_2) \leq s$ and we obtain $T(n, s) \leq \alpha s^{2-s}n^2 + 4\alpha s^{2-s}(n/2)^2 = 2\alpha s^{2-s}n^2$. \hfill $\square$

4. MORE COMPACT GENERATORS

Taking advantage of their low rank property, quasiseparable matrices can be represented by a structured representation allowing to compute efficiently with them, as for example in the context of QR or QZ elimination [9, 1].

The most commonly used generator, as described in [8, 1] and in the introduction, represents an $(r_L, r_U)$-quasiseparable matrix of order $n$ by $O(n(r_L^2 + r_U^2))$ field coefficients.\footnote{Note that the statement of $O(n(r_L + r_U))$ for the same generator in [9] is erroneous.}

Alternatively, hierarchically semiseparable representations (HSS) [18, 11] use numerical rank revealing factorizations of the off-diagonal blocks in a divide and conquer approach, reducing the size to $O(\max(r_L, r_U) \log n)$ [11].

A third approach, based on Givens or unitary weights [4], performs another kind of elimination so as to compact the low rank off-diagonal blocks of the input matrix.

We propose, in this section, two alternative generators, based on an exact PLUQ decomposition revealing the rank profile matrix. The first one matches the best space complexity with numerous zeros. The two proposed generators relevant to this fact.

4.1 A binary tree of PLUQ decompositions

Following the divide and conquer scheme of Algorithm 2, we propose a first generator requiring

\[ O(n(r_L \log \frac{n}{r_L} + r_U \log \frac{n}{r_U})) \] \hfill (3)

coefficients.

For a left triangular matrix $A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}$, the sub-matrix $A_1$ is represented by its PLUQ decomposition $(P_1, L_1, U_1, Q_1)$, which requires $2n^2 \frac{n}{2} \leq sn$ field coefficients for $L_1$ and $U_1$ and $2n$ indices for $P$ and $Q$. This scheme is then recursively applied for the representation of $A_2$ and $A_3$. These matrices have quasiseparable order at most $s$, therefore the following recurrence relation for the size of the representation holds:

\[
\begin{cases}
S(n, s) = sn + 2S(n/2, s) & \text{for } s < n/2 \\
S(n, s) = \frac{s}{2} + 2S(n/2, n/4) & \text{for } s \geq n/2
\end{cases}
\]

For $s \geq n/2$, it solves in $S(n, s) = n^2$. Then for $s < n/2$, $S(n, s) = sn + 2sn/2 + \cdots + 2^k S(n/2^k, s)$, for $k$ such that $\frac{n}{2^k} \leq s < \frac{n}{2^{k-1}}$, which is $k = \log_2 \frac{s}{n}$. Hence $S(n, s) = sn \log_2 \frac{s}{n} + sn = O(sn \log \frac{s}{n})$. The estimate $(3)$ is obtained by applying this generator to the upper and lower triangular parts of $(r_L, r_U)$-quasiseparable matrix.

This first generator does not take fully advantage of the rank structure of the matrix: the representation of each anti-diagonal block is independent from the pivots found in the block $A_i$. The second generator, that will be presented in the next section addresses this issue, in order to remove the logarithmic factors in the estimate $(3)$.

4.2 The Bruhat generator

We propose an alternative generator inspired by the generalized Bruhat decomposition [13, 12, 7]. Contrarily to the former one, it is not depending on a specific recursive cutting of the matrix.

Given a left triangular matrix $A$ of quasiseparable order $s$ and a PLUQ decomposition of it, revealing its rank profile matrix $E$, the generator consists in the three matrices

\[
\begin{align*}
\mathcal{L} & = \text{Left}(P \begin{bmatrix} L & 0 \end{bmatrix} Q), \\
\mathcal{E} & = \text{Left}(E), \\
\mathcal{U} & = \text{Left}(P \begin{bmatrix} U & 0 \end{bmatrix} Q).
\end{align*}
\]

Lemma 4 shows that these three matrices suffice to recover the initial left triangular matrix.

\textbf{Lemma 4.} $A = \text{Left}(\mathcal{L}^T \mathcal{U})$

\textbf{Proof.} $A = P \begin{bmatrix} L & 0 \end{bmatrix} Q Q^T \begin{bmatrix} \tilde{U} \\ \tilde{Q} \end{bmatrix} Q$. From Theorem 1, the matrix $Q^T \begin{bmatrix} \tilde{U} \\ \tilde{Q} \end{bmatrix} Q$ is upper triangular and the matrix $P \begin{bmatrix} L & 0 \end{bmatrix} P^T$ is lower triangular. Applying Lemma 1 yields $A = \text{Left}(A) = \text{Left}(Q^T \begin{bmatrix} \tilde{U} \\ \tilde{Q} \end{bmatrix} Q) = \text{Left}(\mathcal{E}^T P \begin{bmatrix} U \\ 0 \end{bmatrix} Q)$, where $E = P \begin{bmatrix} I_i \\ 0 \end{bmatrix}$ $Q$. Then, as $\mathcal{E}^T$ is the matrix $P \begin{bmatrix} L & 0 \end{bmatrix} P^T$ with some coefficients zeroed out, it is lower triangular, hence applying again Lemma 2 yields $A = \text{Left}(\mathcal{E}^T \mathcal{U})$. \hfill (7)

Consider any non-zero coefficient $e_{j,i}$ of $\mathcal{E}^T$ that is not in its the left triangular part, i.e. $j > n - i$. Its contribution to the product $\mathcal{E}^T$, is only of the form $L_{k,j} e_{j,i}$. However the leading coefficient in column $j$ of $P \begin{bmatrix} L & 0 \end{bmatrix} Q$ is precisely at position $(i, j)$. Since $i > n - j$, this means that the $j$-th column of $\mathcal{L}$ is all zero, and therefore $e_{j,i}$ has no contribution to the product. Hence we finally have $A = \text{Left}(\mathcal{E}^T \mathcal{U})$. \hfill $\square$

We now analyze the space required by this generator.

\textbf{Lemma 5.} Consider an $n \times n$ left triangular rank profile matrix $R$ with quasiseparable order $s$. Then a left triangular matrix $L$ all zero except at the positions of the pivots of $R$ and below these pivots, does not contain more than $s(n - s)$ non-zero coefficients.
Let \( p(k) = \text{rank}(R_{1:k,1:n-k}) \). The value \( p(k) \) indicates the number of non-zero columns located in the \( k \times n-k \) leading sub-matrix of \( L \). Consequently the sum \( \sum_{k=1}^{n-1} p(k) \) is an upper bound on the number of non-zero coefficients in \( L \). Since \( p(k) \leq s \), it is bounded by \( sn \). More precisely, there is no more than \( k \) pivots in the first \( k \) columns and the first \( k \) rows, hence \( p(k) \leq k \) and \( p(n-k) \leq k \) for \( k \leq s \). The bound becomes \( s(s+1) + (n-2s-1)s = s(n-s) \). \[\square\]

**Corollary 1.** The generator \((\mathcal{L}, \mathcal{E}, \mathcal{U})\) uses \( 2s(n-s) \) field coefficients and \( O(n) \) additional indices.

**Proof.** The leading column elements of \( \mathcal{L} \) are located at the pivot positions of the left triangular rank profile matrix \( \mathcal{E} \). Lemma 5 can therefore be applied to show that this matrix occupies no more than \( s(n-s) \) non-zero coefficients. The same argument applies to the matrix \( \mathcal{U} \). \[\square\]

Figure 1 illustrates this generator on a left triangular matrix of quasiseparable order 5. As the supports of \( \mathcal{L} \) and \( \mathcal{U} \) are disjoint, the two matrices can be shown on the same left triangular matrix. The pivots of \( \mathcal{E} \) (black) are the leading coefficients of every non-zero row of \( \mathcal{U} \) and non-zero column of \( \mathcal{L} \).

**Corollary 2.** Any \((r_L, r_U)\)-quasiseparable matrix of dimension \( n \times n \) can be represented by a generator using no more than \( 2n(r_L + r_U) + n - 2(r_L^2 - 2r_L^2) \) field elements.

## 4.3 The compact Bruhat generator

The sparse structure of the Bruhat generator makes it not amenable to the use of fast matrix arithmetic. We therefore propose here a slight variation of it, that will be used in section 5 for fast complexity estimates. We will first describe this compact representation for the \( \mathcal{L} \) factor of the Bruhat generator.

First, remark that there exists a permutation matrix \( Q \) moving the non-zero columns of \( \mathcal{L} \) to the first \( r \) positions, sorted by increasing leading row index, i.e. such that \( \mathcal{L}Q \) is in column echelon form. The matrix \( \mathcal{L}Q \) is now compacted, but still has \( r = \text{rank}(A) \) columns, which may exceed \( s \) and thus preventing to reach complexities in terms of \( n \) and \( s \) only. We will again use the argument of Lemma 5 to produce a more compact representation with only \( O(ns) \) non-zero elements, stored in dense blocks. Algorithm 3 shows how to build such a representation composed of a block diagonal matrix and a block sub-diagonal matrix, where all blocks have column dimension \( s \):

\[
\begin{pmatrix}
S_1 & D_2 & S_3 & D_4 & \cdots \\
S_1 & S_2 & D_3 & S_4 & \cdots \\
S_2 & S_3 & D_4 & S_5 & \cdots \\
\vdots & \vdots & \vdots & \ddots & \ddots \\
S_{n-1} & S_{n-2} & S_{n-3} & \cdots & S_n
\end{pmatrix}
\]

**Algorithm 3 Compressing the Bruhat generator**

**Input:** \( \mathcal{L} \): the first matrix of the Bruhat generator

**Output:** \( D, S, T, Q \): the compression of \( \mathcal{L} \)

1: \( Q \leftarrow \text{a permutation s.t. } \mathcal{L}Q \text{ is in column echelon form} \)
2: \( C \leftarrow \mathcal{L}Q [1:] \text{ where } r = \text{rank}(\mathcal{L}) \)
3: Split \( C \) in column slices of width \( s \).
4: \( D \leftarrow \text{Diag}(C_{11}, \ldots, C_{rr}) \)
5: \( C \leftarrow C - D \)
6: \( T \leftarrow I_n \)
7: for \( i = 3 \ldots t \) do
8: for each non-zero column \( j \) of \( [C_{i-2}, \ldots, C_{i-1}] \) do
9: Let \( k \) be a zero column of \( [C_{i-2}, \ldots, C_{i-1}] \)
10: Move col. \( j \) in \( [C_{i-2}, \ldots, C_{i-1}] \) to col. \( k \) in \( [C_{i-1}] \).
11: end for
12: end for
13: \( S \leftarrow C \)
14: Return \( (D, S, T, Q) \)

**Lemma 6.** Algorithm 3 computes a tuple \((D, S, T, Q)\) where \( Q \) is a permutation matrix putting \( \mathcal{L} \) in column echelon form, \( T \in \{0,1\}^{r \times r} \), \( D = \text{Diag}(D_1, \ldots, D_r) \), \( S = \begin{pmatrix} 0 & \ldots & 0 \\ S_2 & \ldots & S_{r-1} \\ \vdots & \ddots & \vdots \\ S_{r-1} & \ldots & S_r \end{pmatrix} \)

where each \( D_i \) and \( S_i \) is \( k_i \times s \) for \( k_i \geq s \) and \( \sum_{i=1}^{r} k_i = n \). This tuple is the compact Bruhat generator for \( \mathcal{L} \) and satisfies \( \mathcal{L} \equiv [D + ST \ 0_{n \times (n-r)}] Q^T \).

**Proof.** First, note that for every \( i \), the dimensions of the blocks \( S_i \) and \( D_i \) are that of the block \( C_{ii} \). This block contains \( s \) pivots, hence \( k_i \geq s \). We then prove that there always exists a zero column to pick at step 10. The loci of the possible non-zero elements in \( \mathcal{L} \) are column segments below a pivot and above the anti-diagonal. From Lemma 5, these segments have the property that each row of \( \mathcal{L} \) is intersected by no more than \( s \) of them. This property is preserved by column permutation, and still holds on the matrix \( \mathcal{C} \). In the first row of \([C_{i1} \ldots C_{ii}\]), there is a pivot located in the block \( C_{ii} \). Hence there is at most \( s-1 \) such segments intersecting \([C_{i1} \ldots C_{i(i-1)}]\). These \( s-1 \) columns can all be gathered in the block \( C_{i,i-1} \) of column dimension \( s \).
trix A Bruhat decomposition \([13, 7]\): the left factor is a column has full rank, it is a permutation matrix.

R pact Bruhat generator for moved at step 11 of Algorithm 3 are in yellow.

S ter) and of the corresponding compact Bruhat gen-

Figure 2: Support of the matrices Diag step 6. For every pair of indices \((j,k)\) selected in loop 8.

According to (7), the reconstruction of the initial ma-

\[ T(n, s) = \frac{K_\omega}{2} s^{-2} n^{-2} \text{ with } K_\omega = \frac{2^{\omega - 2}}{(2^\omega - 2)(2^{\omega - 2} - 1)} C_\omega \]

where \(C_\omega\) is the leading constant of the complexity of matrix multiplication \([5]\).

5.2 The Bruhat generator

We propose in Algorithm 4 an evolution of Algorithm 2 to compute the factors of the Bruhat generator.

Algorithm 4 LT-Bruhat

Input: \(A\): an \(n \times n\) matrix
Output: \((\mathcal{L}, \mathcal{E}, \mathcal{U})\): a Bruhat generator for the left triangular part of \(A\)

1. if \(n = 1\) then return \(((0), [0], [0])\)
2. Split \(A = [A_1 \ A_2]\) where \(A_1\) is \(\lfloor n/2 \rfloor \times \lfloor n/2 \rfloor\)
3. Decompose \(A_1 = P_1 L_1 \begin{bmatrix} U_1 & V_1 \end{bmatrix} Q_1 \quad \triangleright \text{PLU}(A_1)\)
4. \(R_1 \leftarrow P_1 L_1 \begin{bmatrix} I_1 \ 0 \end{bmatrix} Q_1\) where \(r_1 = \text{rank}(A_1)\).
5. \(B_1 \leftarrow P_1^T A_2 \quad \triangleright \text{Perm}(A_2, P_1^T)\)
6. \(C_1 \ C_2 \leftarrow A_2 Q_1^T \quad \triangleright \text{Perm}(A_3, Q_1^T)\)
7. Here \(A = \begin{bmatrix} L_1 \cup U_1 & V_1 \cup B_1 \\ M_1 & 0 \cup B_2 \end{bmatrix} \quad \triangleright \text{TRSM}(L_1, U_1)\)
8. \(D \leftarrow L_1^{-1} B_1 \quad \triangleright \text{TRSM}(L_1, V_1)\)
9. \(E \leftarrow C_1 U_1 \quad \triangleright \text{TRSM}(C_1, U_1)\)
10. \(F \leftarrow B_2 \cup M_1 \quad \triangleright \text{MM}(B_2, M_1)\)
11. \(G \leftarrow C_2 \cup E V_1 \quad \triangleright \text{MM}(C_2, E, V_1)\)
12. Here \(A = \begin{bmatrix} L_1 \cup U_1 & V_1 \cup D \\ M_1 & 0 \cup F \end{bmatrix} \quad \triangleright \text{TRSM}(L_1, U_1)\)
13. \(H \leftarrow P_1 \begin{bmatrix} 0 \cup 0 \times n \times 2 \end{bmatrix} F \quad \triangleright \text{MM}(C_2, E, V_1)\)
14. \(I \leftarrow \begin{bmatrix} 0 \cup 0 \times \frac{n}{2} \end{bmatrix} G \quad \triangleright \text{MM}(C_2, E, V_1)\)
15. \((L_2, \mathcal{E}_2, \mathcal{U}_2) \leftarrow \text{LT-Bruhat}(H)\)
16. \((L_3, \mathcal{E}_3, \mathcal{U}_3) \leftarrow \text{LT-Bruhat}(I)\)
17. \(L \leftarrow \begin{bmatrix} P_1 \ 1 \ P_1 \end{bmatrix} \begin{bmatrix} L_1 \ M_1 \ O \end{bmatrix} \begin{bmatrix} Q_1 \ I_2 \end{bmatrix} \quad \triangleright \text{TRSM}(L_1, \mathcal{L}_2)\)
18. \(U \leftarrow \begin{bmatrix} P_1 \ U_1 \ V_1 \end{bmatrix} \begin{bmatrix} 0 \times \frac{n}{2} \times \frac{n}{2} \end{bmatrix} Q_1 \quad \triangleright \text{Left}(P_1, D)\)
19. \(E \leftarrow \begin{bmatrix} \mathcal{E}_1 \ \mathcal{E}_2 \end{bmatrix} \quad \triangleright \text{MM}(C_2, E, V_1)\)
20. return \((\mathcal{L}, \mathcal{E}, \mathcal{U})\)

Theorem 3. For any \(n \times n\) matrix \(A\) with a left triangular part of quasiseparable order \(s\), Algorithm 4 computes the Bruhat generator of the left triangular part of \(A\) in \(O(s^{\omega - 2} n^2)\) field operations.
PROOF. The correctness of \( \mathcal{C} \) is proven in Theorem 2. We will prove by induction the correctness of \( \mathcal{U} \), noting that the correctness of \( \mathcal{L} \) works similarly.

Let \( H = P_2 P_3 U_2 Q_2 \) and \( I = P_3 L_3 U_3 Q_3 \) be PLUQ decompositions of \( H \) and \( I \) revealing their rank profile matrices. Assume that Algorithm LT-Bruhat is correct in the two recursive calls 15 and 16, that is

\[
\mathcal{U}_2 = \text{Left}(P_2 \begin{bmatrix} U_2 & 0 \\ 0 & Q_2 \end{bmatrix}, \mathcal{L}_2 = \text{Left}(P_3 \begin{bmatrix} U_3 & 0 \\ 0 & Q_3 \end{bmatrix})
\]

At step 7, we have

\[
\begin{bmatrix} A_1 & A_2 \\ A_3 & * \end{bmatrix} = \begin{bmatrix} P_1 & \frac{I_2}{T} \end{bmatrix} \begin{bmatrix} L_1 & M_1 \bar{I}_2 & -r_1 \\ 0 & E \frac{I_2}{T} \end{bmatrix} \begin{bmatrix} \bar{U} & V_1 & \bar{D} \\ 0 & F & Q_1 \end{bmatrix} \begin{bmatrix} \bar{Q}_1 & D \bar{Q}_2 \\ Q_1 & Q_2 \end{bmatrix}
\]

As the first \( r_1 \) rows of \( P^T_2 H \) are zeros, there exists \( \bar{P}_2 \) a permutation matrix and \( L_2 \), a lower triangular matrix, such that \( P^T_2 P_2 L_2 = \begin{bmatrix} 0_{r_1} & \bar{P}_2 L_2 \end{bmatrix} \). Similarly, there exist \( \bar{Q}_3 \), a permutation matrix and \( \bar{U}_1 \), an upper triangular matrix, such that \( U_2 \bar{Q}_3 Q^T_3 = \begin{bmatrix} 0_{r_2} & \bar{U}_1 \bar{Q}_3 \end{bmatrix} \). Hence

\[
\begin{bmatrix} A_1 & A_2 \\ A_3 & * \end{bmatrix} = \begin{bmatrix} P_1 & P_3 \end{bmatrix} \begin{bmatrix} L_1 & M_1 \bar{I}_2 & -r_1 \\ 0 & E \bar{I}_2 & 0 \end{bmatrix} \begin{bmatrix} \bar{U} & V_1 & \bar{D} \\ 0 & F & Q_1 \end{bmatrix} \begin{bmatrix} \bar{Q}_1 & D \bar{Q}_2 \\ Q_1 & Q_2 \end{bmatrix}
\]

Setting \( N_1 = \bar{P}_2^T M_1 \) and \( W_1 = V_1 \bar{Q}_3^T \), we have

\[
\begin{bmatrix} A_1 & A_2 \\ A_3 & * \end{bmatrix} = \begin{bmatrix} P_1 & P_3 \end{bmatrix} \begin{bmatrix} L_1 & N_1 \bar{I}_2 & -r_1 \\ 0 & E \bar{I}_2 & L_3 \end{bmatrix} \begin{bmatrix} \bar{U} & W_1 & \bar{D} \\ 0 & 0 & \bar{Q}_3 \end{bmatrix} \begin{bmatrix} \bar{Q}_1 & D \bar{Q}_2 \\ Q_1 & Q_2 \end{bmatrix}
\]

A PLUQ of \( \begin{bmatrix} A_1 & A_2 \\ A_3 & * \end{bmatrix} \) revealing its rank profile matrix is then obtained from this decomposition by a row block cyclic-shift on the second factor and a column block cyclic shift on the third factor as in [5, Algorithm 1]. Finally,

\[
\begin{bmatrix} U_1 & V_1 & D \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{Q}_1 & D \bar{Q}_2 \\ Q_1 & Q_2 \end{bmatrix}
\]

Hence

\[
\text{Left(PLUQ)} = \begin{bmatrix} P_1 \begin{bmatrix} U_1 & V_1 & D \\ 0 & 0 & 0 \end{bmatrix} Q_1 \end{bmatrix} + \begin{bmatrix} P_3 \begin{bmatrix} U_3 & 0 \\ 0 & Q_1 \end{bmatrix} Q_1 \end{bmatrix}
\]

The complexity analysis is exactly that of Theorem 2. \( \square \)

The computation of a compact Bruhat generator is obtained by combining Algorithm 4 with Algorithm 3.

5.2 Applying a vector

For the three generators proposed earlier, the application of a vector to the corresponding left triangular matrix takes the same amount of field operations as the number of coefficients used for its representation. This yields a cost of \( O(n(r_L \log \frac{n}{r_L} + r_U \log \frac{n}{r_U})) \) field operations for multiplying a vector to an \( (r_L, r_U) \)-quasiseparable matrix using the binary tree PLUQ generator and \( O(n(r_L + r_U)) \) using either one of the Bruhat generator or its compact variant.

5.3 Multiplying two left-triangular matrices

5.3.1 The binary tree PLUQ generator

Let \( T_{RL}(n,s) \) denote the cost of multiplying a dense \( s \times n \) matrix by a left triangular quasiseparable matrix of order \( s \). The natural divide and conquer algorithm yields the recurrence formula:

\[
T_{RL}(n,s) = 2T_{RL}(n/2,s) + O(ns^{\omega-1}) = O(ns^{\omega-1} \log \frac{n}{s}).
\]

Let \( T_{PL}(n,s) \) denote the cost of multiplying a PLUQ decomposition of dimension \( n \) and rank \( s \leq n/2 \) with a left triangular quasiseparable matrix of order \( s \). The product can be done in

\[
T_{PL}(n,s) = T_{RL}(n,s) + O(n^2s^{\omega-2}) = O(n^2s^{\omega-2}).
\]

Lastly, let \( T_{LL}(n,s) \) denote the cost of multiplying two left-triangular matrices of quasiseparability order \( s \). Again the natural recursive algorithm yields:

\[
T_{LL}(n,s) = 2T_{LL}(n/2,s) + 2T_{PL}(n/2,s) + O(n^2s^{\omega-2}) = O(n^2s^{\omega-2}).
\]

5.3.2 The Bruhat generator

Using the decomposition (8), the product of two left triangular matrices writes \( A \times B = C x E x A \times C x E x B \) where \( C_x = D_{C_x} + S_{C_x} T_{C_x} \) and \( E_x = D_{E_x} + T_{E_x} S_{E_x} \) for \( X \in \{A, B\} \). We will compute it using the following parenthesizing:

\[
A \times B = C x (R_x (E_x \times C x) R_x) E_B. \tag{9}
\]

The product \( E_x \times C_x = (D_{E_x} + T_{E_x} S_{E_x}) (D_{C_x} + S_{C_x} T_{C_x}) \) only consists in multiplying together block diagonal or sub-diagonal matrices \( n \times r_B \) or \( r_A \times n \). We will describe the product of two block diagonal matrices (flat times tall); the other cases with sub-diagonal matrices work similarly.

Each term to be multiplied is decomposed in a grid of \( s \times s \) tiles (except at the last row and column positions). In this grid, the non-zero blocks are non longer in a block-diagonal layout: in a flat matrix, the leading block of a block row may lie at the same block column position as the trailing block of its preceding block row, as shown in Figure 3. However, since \( k_i \geq s \) for all \( i \), no more than two

![Figure 3: Aligning a block diagonal matrix (blue) on an \( s \times s \) grid. Each block row of the aligned structure (red) may overlap with the previous and next block rows on at most one \( s \times s \) tile on each side.](image-url)
consecutive block rows of a flat matrix lie in the same block column. Consequently these terms can be decomposed as a sum of two block diagonal matrices aligned on an $s \times s$ grid. Multiplying two such matrices costs $O(s^2 - 1)n$ which is consequently also the cost of computing the product $E_A C_B$. After left and right multiplication by the permutations $R_A$ and $R_B$, this $r_A \times r_B$ dense matrix is multiplied to the left by $C_B$. This costs $O(nr_B s^{w-2})$. Lastly, the right multiplication by $E_R$ of the resulting $n \times r_A$ matrix costs $O(n^2 s^{w-2})$ which dominates the overall cost.

5.4 Multiplying two quasiseparable matrices

Decomposing each multiplicand into its upper, lower and diagonal terms, a product of two quasiseparable matrices writes $A \times B = (L_A + D_A + U_A)(L_B + D_B + U_B)$. Beside the scaling by diagonal matrices, all other operations involve a product between any combination of lower an upper triangular matrices, which in turn translates into products of left triangular matrices and $J_n$ as shown in Table 1. The complexity of section 5.3 directly applies for the computation of upper products. For the other products, a $J_n$ factor has to be added between the $E_A$ and $C_B$ factors in the innermost product of (9). As reverting the row order of $C_B$ does not impact the cost of computing this product, the same complexity applies here too.

<table>
<thead>
<tr>
<th>$\times$</th>
<th>Lower</th>
<th>Upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower</td>
<td>$J_n \times L_A \times J_n \times L_B$</td>
<td>$J_n \times L_A \times L_B \times J_n$</td>
</tr>
<tr>
<td>Upper</td>
<td>$J_n \times L_A \times L_B \times J_n$</td>
<td>$J_n \times L_A \times L_B \times J_n$</td>
</tr>
</tbody>
</table>

Table 1: Reducing products of lower and upper to products of left triangular matrices.

Complexity of computing the rank profile matrix to order respectively $(l_A, u_A)$ and $(l_B, u_B)$ costs $O(n^2 s^{w-2})$ field operations where $s = \max(l_A, u_A, l_B, u_B)$, using either one of the binary tree or the compact Bruhat generator.

6. PERSPECTIVES

The algorithms proposed for multiplying two quasiseparable matrices output a dense $n \times n$ matrix in time $O(n^2 s^{w-2})$; for $s = \max(l_A, u_A, l_B, u_B)$. However, the product is also a quasiseparable matrix, of order $(l_A + l_B, u_A + u_B)$ [8, Theorem 4.1], which can be represented by a Bruhat generator with only $O((l_A + l_B + u_A + u_B))$ coefficients. A first natural question is thus to find an algorithm computing this representation from the generators of $A$ and $B$ in time $O(ns^{w-1})$.

Second, a probabilistic algorithm [7, § 7] reduces the complexity of computing the rank profile matrix to $O(n^2 + r^2)$. It is not clear whether it can be applied to compute a compact Bruhat generator in time $O(n^2 + \max(l_A, u_A)^2)$.

Note (added Sept. 16, 2016.)

Equation (9) for the multiplication of two Bruhat generators is missing the Left operators, and is therefore incorrect. The target complexities can still be obtained by slight modification of the algorithm: computing the inner-most product $E_A \times C_B$ as an unevaluated sum of blocks products. This will be detailed in a follow-up paper.

Acknowledgment

We thank Paola Boito for introducing us to the field of quasiseparable matrices and two anonymous referees for pointing us to the HSS and the Givens weight representations. We acknowledge the financial support from the HPAC project (ANR 11 BS02 013) and from the OpenDreamKit Horizon 2020 European Research Infrastructures project (#676541).

7. REFERENCES

http://linalg.org/.

