Computing with quasiseparable matrices

Clement Pernet

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 $O(n^2 s^{\omega-2})$ where $s = \max(r_L, r_U)$ and ω the exponent of matrix multiplication. We then present two new structured representations, a binary tree of PLUQ decompositions, and the Bruhat generator, using respectively $O(ns \log \frac{n}{s})$ and $O(ns)$ field elements instead of $O(ns^2)$ for the classical generator and $O(ns \log n)$ for the hierarchically semiseparable representations. We present algorithms computing these representations in time $O(n^2 s^{\omega-2})$. 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 $O(n^2 s^{\omega-2})$.

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 $1 \le k \le n-1$,*

$$\begin{aligned} rank(\mathsf{M}_{k+1..n,1..k}) &\le r_L, & (1)\\ rank(\mathsf{M}_{1..k,k+1..n}) &\le r_U. & (2)\end{aligned}$$

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

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

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, 1] 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 $r_L \times r_L$, and $n-1$ matrices $b(i)$ of dimension $r_U \times r_U$ such that

$$\mathsf{M}_{i,j} = \begin{cases} p(i)^T \mathsf{a}^{>}_{ij} q(j), & 1 \le j < i \le n\\ d(i), & 1 \le i = j \le n\\ g(i)^T \mathsf{b}^{<}_{ij} h(j), & 1 \le i < j \le n\end{cases}$$

where $\mathsf{a}^{>}_{ij} = \mathsf{a}(i-1)\ldots\mathsf{a}(j+1)$ for $j > i+1$, $\mathsf{a}_{j+1,j} = 1$, and $\mathsf{b}^{<}_{ij} = \mathsf{b}(i+1)\ldots\mathsf{b}(i-1)$ 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 multiprecision 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 $O(n^2 s^{\omega-2})$ where $s = \max(r_L, r_U)$ and ω the exponent of matrix multiplication. We then present in Section 4 two new structured representations, a binary tree of PLUQ decompositions, and the Bruhat generator, using respectively $O(ns\log\frac{n}{s})$ and $O(ns)$ field elements instead of $O(ns^2)$ for the previously known generators. We present in Section 5 algorithms computing them in time $O(n^2s^{\omega-2})$. These representations support 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 $O(n^2s^{\omega-2})$.

Throughout the paper, $\mathsf{A}_{i..j,k..l}$ will denote the sub-matrix of A of row indices between i and j and column indices between k and l. The matrix of the canonical basis, with a one at position (i,j) will be denoted by $\Delta^{(i,j)}$.

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 $\mathsf{A}_{i,j}=0$ for all $i>n-j$.*

The left triangular part of a matrix A, denoted by $\text{Left}(\mathsf{A})$ 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 $\mathsf{A}=\mathsf{BU}$ be an $m\times n$ matrix where U is $n\times n$ upper triangular. Then $\mathit{Left}(\mathsf{A})=\mathit{Left}(\mathit{Left}(\mathsf{B})\mathsf{U})$.*

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

Applying Lemma 1 on A^T yields Lemma 2

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

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 $\mathcal{R}_\mathsf{A}$, with only r non-zero coefficients, all equal to one, located on distinct rows and columns such that any leading sub-matrices of $\mathcal{R}_\mathsf{A}$ 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 $O(mnr^{\omega-2})$ field operations [6], provided some conditions on the pivoting strategy being used. It is obtained from the corresponding PLUQ decomposition as the product

$$\mathcal{R}_A=\mathsf{P}\begin{bmatrix}\mathsf{I}_r & \\ & 0_{(m-r)\times(n-r)}\end{bmatrix}\mathsf{Q}.$$

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 $\mathsf{A}=\mathsf{PLUQ}$ be a PLUQ decomposition revealing the rank profile matrix of A. Then, $\mathsf{P}\begin{bmatrix}\mathsf{L} & 0_{m\times(m-r)}\end{bmatrix}\mathsf{P}^T$ is lower triangular and $\mathsf{Q}^T\begin{bmatrix}\mathsf{U}\\ 0_{(n-r)\times n}\end{bmatrix}\mathsf{Q}$ 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 $\text{rank}(B_{1..i,1..j})=\text{rank}(L_{1..i,1..i}A_{1..i,1..j})=\text{rank}(A_{1..i,1..j})$ for any $i\le m,j\le n$. Hence $\mathcal{R}_B=\mathcal{R}_A$. $\square$

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_n be the unit anti-diagonal matrix. Multiplying on the left by J_n reverts the row order while multiplying on the right by J_n reverts the column order. Hence both $\mathsf{J}_n\mathsf{L}$ and $\mathsf{U}\mathsf{J}_n$ are left triangular matrices. Remark that the conditions (1) and (2) state that all leading $k\times(n-k)$ sub-matrices of $\mathsf{J}_n\mathsf{L}$ and $\mathsf{U}\mathsf{J}_n$ 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 $\left[\begin{smallmatrix}1&1&0\\1&0&0\\0&0&0\end{smallmatrix}\right]$ is $\left[\begin{smallmatrix}1&0&0\\0&1&0\\0&0&0\end{smallmatrix}\right]$. 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 $\mathsf{J}_n\mathsf{L}$ and $\mathsf{U}\mathsf{J}_n$, 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: $\mathsf{A}$, an $n \times n$ matrix
Output: $\max\{\text{rank}(\mathsf{A}_{1..k,1..n-k}) : 1 \le k \le n-1\}$
  $\mathsf{R} \leftarrow$ LT-RPM($\mathsf{A}$) ▷ The left triangular part of the rank profile matrix of $\mathsf{A}$
  rows $\leftarrow$ (False,...,False)
  cols $\leftarrow$ (False,...,False)
  for all $(i,j)$ such that $\mathsf{R}_{i,j} = 1$ do
    rows[i] $\leftarrow$ True
    cols[j] $\leftarrow$ True
  end for
  $s, r \leftarrow 0$
  for $i = 1 \dots n$ do
    if rows[$i$] then $r \leftarrow r + 1$
    if cols[$n - i + 1$] then $r \leftarrow r - 1$
    $s \leftarrow \max(s, r)$
  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 $O(n^2 r^{\omega-2})$ where $r = \text{rank}(\mathsf{A})$. 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 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 $s = r$.*

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 discrepency 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.

Algorithm 2 LT-RPM: Left Triangular part of the Rank Profile Matrix

```
Input: $\mathsf{A}$: an $n \times n$ matrix
Output: $\mathcal{R}$: the left triangular part of the RPM of $\mathsf{A}$
 1: if $n = 1$ then return $[0]$
 2: Split $\mathsf{A} = \begin{bmatrix} \mathsf{A}_1 & \mathsf{A}_2 \\ \mathsf{A}_3 & \end{bmatrix}$ where $\mathsf{A}_3$ is $\lfloor \frac{n}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor$
 3: Decompose $\mathsf{A}_1 = \mathsf{P}_1 \begin{bmatrix} \mathsf{L}_1 \\ \mathsf{M}_1 \end{bmatrix} \begin{bmatrix} \mathsf{U}_1 & \mathsf{V}_1 \end{bmatrix} \mathsf{Q}_1$
 4: $\mathcal{R}_1 \leftarrow \mathsf{P}_1 \begin{bmatrix} \mathsf{I}_{r_1} & \\ & 0 \end{bmatrix} \mathsf{Q}_1$ where $r_1 = \text{rank}(\mathsf{A}_1)$.
 5: $\begin{bmatrix} \mathsf{B}_1 \\ \mathsf{B}_2 \end{bmatrix} \leftarrow \mathsf{P}_1{}^T \mathsf{A}_2$
 6: $\begin{bmatrix} \mathsf{C}_1 & \mathsf{C}_2 \end{bmatrix} \leftarrow \mathsf{A}_3 \mathsf{Q}_1{}^T$
 7: Here $A = \left[\begin{array}{cc|c} \mathsf{L}_1 \backslash \mathsf{U}_1 & \mathsf{V}_1 & \mathsf{B}_1 \\ \mathsf{M}_1 & 0 & \mathsf{B}_2 \\ \hline \mathsf{C}_1 & \mathsf{C}_2 & \end{array}\right]$.
 8: $\mathsf{D} \leftarrow \mathsf{L}_1{}^{-1} \mathsf{B}_1$
 9: $\mathsf{E} \leftarrow \mathsf{C}_1 \mathsf{U}_1{}^{-1}$
10: $\mathsf{F} \leftarrow \mathsf{B}_2 - \mathsf{M}_1 \mathsf{D}$
11: $\mathsf{G} \leftarrow \mathsf{C}_2 - \mathsf{E} \mathsf{V}_1$
12: Here $\mathsf{A} = \left[\begin{array}{cc|c} \mathsf{L}_1 \backslash \mathsf{U}_1 & \mathsf{V}_1 & \mathsf{D} \\ \mathsf{M}_1 & 0 & \mathsf{F} \\ \hline \mathsf{E} & \mathsf{G} & \end{array}\right]$.
13: $\mathsf{H} \leftarrow \mathsf{P}_1 \begin{bmatrix} 0_{r_1 \times \frac{n}{2}} \\ \mathsf{F} \end{bmatrix}$
14: $\mathsf{I} \leftarrow \begin{bmatrix} 0_{r_1 \times \frac{n}{2}} & \mathsf{G} \end{bmatrix} \mathsf{Q}_1$
15: $\mathcal{R}_2 \leftarrow$ LT-RPM($\mathsf{H}$)
16: $\mathcal{R}_3 \leftarrow$ LT-RPM($\mathsf{I}$)
17: return $\mathcal{R} \leftarrow \begin{bmatrix} \mathcal{R}_1 & \mathcal{R}_2 \\ \mathcal{R}_3 & \end{bmatrix}$
```

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 $O(n^2 s^{\omega-2})$.*

PROOF. First remark that

$$\mathsf{P}_1 \begin{bmatrix} \mathsf{D} \\ \mathsf{F} \end{bmatrix} = \underbrace{\mathsf{P}_1 \begin{bmatrix} \mathsf{L}_1{}^{-1} & \\ -\mathsf{M}_1 \mathsf{L}_1{}^{-1} & \mathsf{I}_{n-r_1} \end{bmatrix} \mathsf{P}_1{}^T}_{\mathsf{L}} \mathsf{P}_1 \begin{bmatrix} \mathsf{B}_1 \\ \mathsf{B}_2 \end{bmatrix} = \mathsf{L}\mathsf{A}_2.$$

Hence

$$\mathsf{L} \begin{bmatrix} \mathsf{A}_1 & \mathsf{A}_2 \end{bmatrix} = \mathsf{P}_1 \left[\begin{array}{c|c} \begin{bmatrix} \mathsf{U}_1 & \mathsf{V}_1 \end{bmatrix} \mathsf{Q}_1 & \mathsf{D} \\ 0 & \mathsf{F} \end{array}\right].$$

From Theorem 1, the matrix L is lower triangular and by Lemma 3 the rank profile matrix of $\begin{bmatrix} \mathsf{A}_1 & \mathsf{A}_2 \end{bmatrix}$ equals that of $\mathsf{P}_1 \left[\begin{array}{c|c} \begin{bmatrix} \mathsf{U}_1 & \mathsf{V}_1 \end{bmatrix} \mathsf{Q}_1 & \mathsf{D} \\ 0 & \mathsf{F} \end{array}\right]$. Now as U_1 is upper triangular and non-singular, this rank profile matrix is in turn that of $\mathsf{P}_1 \left[\begin{array}{c|c} \begin{bmatrix} \mathsf{U}_1 & \mathsf{V}_1 \end{bmatrix} \mathsf{Q}_1 & 0 \\ 0 & \mathsf{F} \end{array}\right]$ and its left triangular part is $\begin{bmatrix} \mathcal{R}_1 & \mathcal{R}_2 \end{bmatrix}$.

By a similar reasoning, $\begin{bmatrix} \mathcal{R}_1 & \mathcal{R}_3 \end{bmatrix}^T$ is the left triangular part of the rank profile matrix of $\begin{bmatrix} \mathsf{A}_1 & \mathsf{A}_3 \end{bmatrix}^T$, 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

$$T(n,s) = \alpha r_1^{\omega-2} n^2 + T_{\text{LT-RPM}}(n/2, s_1) + T_{\text{LT-RPM}}(n/2, s_2)$$

for a positive constant α. We will prove by induction that $T(n,s) \le 2\alpha s^{\omega-2}n^2$.

Again, since L is lower triangular, the rank profile matrix of LA_2 is that of A_2 and the quasiseparable orders of the two matrices are the same. Now H is the matrix LA_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 $\max(r_1, s_1, s_2) \le s$ and we obtain $T(n,s) \le \alpha s^{\omega-2}n^2 + 4\alpha s^{\omega-2}(n/2)^2 = 2\alpha s^{\omega-2}n^2$. $\square$

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[1].

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 $O(\max(r_L, r_U)n\log n)$ [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 of the HSS representation, and improves the time complexity to compute it by a reduction to fast matrix multiplication. The second one also improves on the space complexity of HSS representation by removing the extra $\log n$ factor and shares some similarities with the unitary weight representations of [4].

First, remark that storing a PLUQ decomposition of rank r and dimension $n \times n$ uses $2rn - r^2$ coefficients: each of the L and U factor has dimension $n \times r$ or $r \times n$; the negative r^2 term comes from the lower and upper triangular shapes of L and U. Here again, the rank r can be larger than the quasiseparable order s thus storing directly a PLUQ decomposition is too expensive. But as in Remark 1, the setting where $r \gg s$ is precisely when the pivots are near the anti-diagonal, and therefore the L and U factors have an additional structure, with numerous zeros. The two proposed generators, rely on 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

$$O\left(n\left(r_L \log\frac{n}{r_L} + r_U \log\frac{n}{r_U}\right)\right) \tag{3}$$

coefficients.

For a left triangular matrix $\mathsf{A} = \begin{bmatrix}\mathsf{A}_1 & \mathsf{A}_2\\ \mathsf{A}_3 & \end{bmatrix}$, the sub-matrix A_1 is represented by its PLUQ decomposition $(\mathsf{P}_1, \mathsf{L}_1, \mathsf{U}_1, \mathsf{Q}_1)$, which requires $2r_1\frac{n}{2} \le sn$ 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:

$$\begin{cases} S(n,s) &= sn + 2S(n/2, s) & \text{for } s < n/2\\ S(n,s) &= \frac{n^2}{2} + 2S(n/2, n/4) & \text{for } s \ge n/2 \end{cases}$$

For $s \ge n/2$, 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 $\frac{n}{2^k} \le s < \frac{n}{2^{k-1}}$, which is $k = \lceil \log_2 \frac{n}{s} \rceil$. Hence $S(n,s) = sn\log_2\frac{n}{s} + sn = O(sn\log\frac{n}{s})$. The estimate (3) is obtained by applying this generator to the upper and lower triangular parts of the (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_1. The second generator, that will be presented in the next section adresses 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

$$\mathcal{L} = \text{Left}(\mathsf{P}\begin{bmatrix}\mathsf{L} & 0\end{bmatrix}\mathsf{Q}), \tag{4}$$
$$\mathcal{E} = \text{Left}(\mathsf{E}), \tag{5}$$
$$\mathcal{U} = \text{Left}(\mathsf{P}\begin{bmatrix}\mathsf{U}\\ 0\end{bmatrix}\mathsf{Q}). \tag{6}$$

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

LEMMA 4. $\mathsf{A} = \mathit{Left}(\mathcal{L}\mathcal{E}^T\mathcal{U})$

PROOF. $\mathsf{A} = \mathsf{P}\begin{bmatrix}\mathsf{L} & 0_{m\times(n-r)}\end{bmatrix}\mathsf{Q}\mathsf{Q}^T\begin{bmatrix}\mathsf{U}\\ 0_{(n-r)\times n}\end{bmatrix}\mathsf{Q}$. From Theorem 1, the matrix $\mathsf{Q}^T\begin{bmatrix}\mathsf{U}\\ 0\end{bmatrix}\mathsf{Q}$ is upper triangular and the matrix $\mathsf{P}\begin{bmatrix}\mathsf{L} & 0\end{bmatrix}\mathsf{P}^T$ is lower triangular. Applying Lemma 1 yields $\mathsf{A} = \text{Left}(\mathsf{A}) = \text{Left}(\mathcal{L}\mathsf{Q}^T\begin{bmatrix}\mathsf{U}\\ 0\end{bmatrix}\mathsf{Q}) = \text{Left}(\mathcal{L}\mathsf{E}^T\mathsf{P}\begin{bmatrix}\mathsf{U}\\ 0\end{bmatrix}\mathsf{Q})$, where $\mathsf{E} = \mathsf{P}\begin{bmatrix}\mathsf{I}_r & \\ & 0\end{bmatrix}\mathsf{Q}$. Then, as $\mathcal{L}\mathsf{E}^T$ is the matrix $\mathsf{P}\begin{bmatrix}\mathsf{L} & 0\end{bmatrix}\mathsf{P}^T$ with some coefficients zeroed out, it is lower triangular, hence applying again Lemma 2 yields

$$\mathsf{A} = \text{Left}(\mathcal{L}\mathsf{E}^T\mathcal{U}). \tag{7}$$

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

We now analyze the space required by this generator.

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.*

[1]Note that the statement of $O(n(r_L + r_U))$ for the same generator in [9] is erroneous.

Proof. Let $p(k) = \operatorname{rank}(\mathsf{R}_{1..k,1..n-k})$. 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 $s(s+1)+(n-2s-1)s = s(n-s)$. □

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 $\mathcal{U}$. □

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 $\mathcal{L}$.

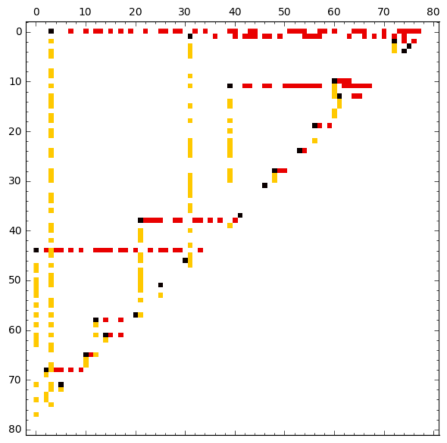

Figure 1: Support of the $\mathcal{L}$ (yellow), $\mathcal{E}$ (black) and $\mathcal{U}$ (red) matrices of the Bruhat generator for a 80×80 left triangular matrix of quasiseparable order 5.

Corollary 2. *Any (r_L, r_U)-quasiseparable matrix of dimension $n \times n$ can be represented by a generator using no more than $2n(r_L + r_U) + n - 2(r_L^2 - 2r_U^2)$ 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 we will use 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 $\mathcal{Q}$ moving the non-zero columns of $\mathcal{L}$ to the first r positions, sorted by increasing leading row index, i.e. such that $\mathcal{LQ}$ is in column echelon form. The matrix $\mathcal{LQ}$ is now compacted, but still has $r = \operatorname{rank}(A)$ 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: $\begin{bmatrix} \mathsf{D}_1 & & & \\ \mathsf{S}_2 & \mathsf{D}_2 & & \\ & \mathsf{S}_3 & \mathsf{D}_3 & \\ & & \ddots & \ddots \\ & & & \mathsf{S}_t \; \mathsf{D}_t \end{bmatrix}$.

Algorithm 3 Compressing the Bruhat generator

```
Input: $\mathcal{L}$: the first matrix of the Bruhat generator
Output: $\mathsf{D}, \mathsf{S}, \mathsf{T}, \mathcal{Q}$: the compression of $\mathcal{L}$
 1: $\mathcal{Q} \leftarrow$ a permutation s.t. $\mathcal{LQ}$ is in column echelon form
 2: $\mathsf{C} \leftarrow \mathcal{LQ}\left[\begin{smallmatrix} \mathsf{I}_r \\ 0 \end{smallmatrix}\right]$ where $r = \operatorname{rank}(\mathcal{L})$
 3: Split $\mathsf{C}$ in column slices of width $s$.
 4:     $\triangleright\ \mathsf{C} = \begin{bmatrix} \mathsf{C}_{11} & & & \\ \mathsf{C}_{21} & \mathsf{C}_{22} & & \\ \vdots & \vdots & \ddots & \\ \mathsf{C}_{t1} & \mathsf{C}_{t2} & \dots & \mathsf{C}_{tt} \end{bmatrix}$ where $\mathsf{C}_{ii}$ is $k_i \times s$.
 5: $\mathsf{D} \leftarrow \operatorname{Diag}(\mathsf{C}_{11}, \dots, \mathsf{C}_{tt})$
 6: $\mathsf{C} \leftarrow \mathsf{C} - \mathsf{D} = \begin{bmatrix} 0 & & & \\ \mathsf{C}_{21} & & & \\ \vdots & \ddots & \ddots & \\ \mathsf{C}_{t1} & \dots & \mathsf{C}_{t,t-1} & 0 \end{bmatrix}$
 7: $\mathsf{T} \leftarrow \mathsf{I}_n$
 8: for $i = 3 \dots t$ do
 9:     for each non zero column $j$ of $\begin{bmatrix} \mathsf{C}_{i,i-2} \\ \cdots \\ \mathsf{C}_{t,i-2} \end{bmatrix}$ do
10:         Let $k$ be a zero column of $\begin{bmatrix} \mathsf{C}_{i,i-1} \\ \cdots \\ \mathsf{C}_{t,i-1} \end{bmatrix}$
11:         Move col. $j$ in $\begin{bmatrix} \mathsf{C}_{i,i-2} \\ \vdots \\ \mathsf{C}_{t,i-2} \end{bmatrix}$ to col. $k$ in $\begin{bmatrix} \mathsf{C}_{i,i-1} \\ \vdots \\ \mathsf{C}_{t,i-1} \end{bmatrix}$.
12:         $\mathsf{T} \leftarrow (\mathsf{I}_n + \Delta^{(k,j)} - \Delta^{(k,k)}) \times \mathsf{T}$
13:     end for
14: end for
15: $\mathsf{S} \leftarrow \mathsf{C} = \begin{bmatrix} 0 & & & \\ \mathsf{C}_{21} & 0 & & \\ & \ddots & \ddots & \\ & & \mathsf{C}_{t,t-1} & 0 \end{bmatrix}$
16: Return $(\mathsf{D}, \mathsf{S}, \mathsf{T}, \mathcal{Q})$
```

Lemma 6. *Algorithm 3 computes a tuple $(\mathsf{D}, \mathsf{S}, \mathsf{T}, \mathcal{Q})$ where $\mathcal{Q}$ is a permutation matrix putting $\mathcal{L}$ in column echelon form, $\mathsf{T} \in \{0,1\}^{r \times r}$, $\mathsf{D} = \operatorname{Diag}(\mathsf{D}_1, \dots, \mathsf{D}_t)$, $\mathsf{S} = \begin{bmatrix} 0 & & & \\ \mathsf{S}_2 & 0 & & \\ & \ddots & \ddots & \\ & & \mathsf{S}_t & \end{bmatrix}$ where each D_i and S_i is $k_i \times s$ for $k_i \geq s$ and $\sum_{i=1}^{t} k_i = n$. This tuple is the compact Bruhat generator for $\mathcal{L}$ and satisfies $\mathcal{L} = \begin{bmatrix} \mathsf{D} + \mathsf{ST} & 0_{n \times (n-r)} \end{bmatrix} \mathcal{Q}^T$.*

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 C. In the first row of $\begin{bmatrix} \mathsf{C}_{i1} & \dots & \mathsf{C}_{ii} \end{bmatrix}$, there is a pivot located in the block C_{ii}. Hence there is at most $s-1$ such segments intersecting $\begin{bmatrix} \mathsf{C}_{i1} & \dots & \mathsf{C}_{i,i-1} \end{bmatrix}$. These $s-1$ columns can all be gathered in the block $\mathsf{C}_{i,i-1}$ of column dimension s.

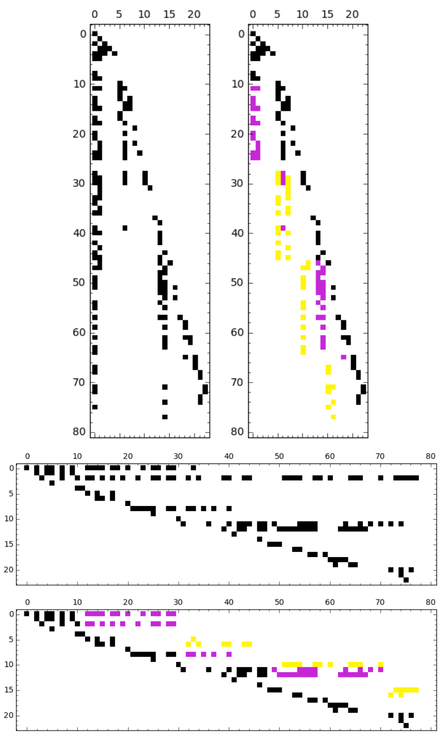

Figure 2: Support of the matrices $\mathcal{LQ}$ (left), $\mathcal{PU}$ (center) and of the corresponding compact Bruhat generator (right and bottom) for the matrix of Figure 1. In the compact Bruhat generator: D is in black and S in magenta and yellow; those rows and columns moved at step 11 of Algorithm 3 are in yellow.

There only remains to show that ST is the matrix C of step 6. For every pair of indices (j,k) selected in loop 8, right multiplication by $(\mathsf{I}_n + \Delta^{(k,j)} - \Delta^{(k,k)})$ adds up column k to column j and zeroes out column k. On matrix S, this has the effect of reverting each operation done at step 11 in the reverse order of the loop 8. $\square$

A compact representation of $\mathcal{U}$ is obtained in Lemma 7 by running Algorithm 3 on $\mathcal{U}^T$ and transposing its output.

Lemma 7. *There exist a tuple $(\mathsf{D},\mathsf{S},\mathsf{T},\mathcal{P})$ called the compact Bruhat generator for $\mathcal{U}$ such that $\mathcal{P}$ is a permutation matrix putting $\mathcal{U}$ in row echelon form, $\mathsf{T} \in \{0,1\}^{r\times r}$, $\mathsf{D} =$ Diag$(\mathsf{D}_1,\ldots,\mathsf{D}_t)$, $\mathsf{S} = \begin{bmatrix} 0 & \mathsf{S}_2 & & \\ & \ddots & \ddots & \\ & & 0 & \mathsf{S}_t \end{bmatrix}$ where each D_i and S_i is $s \times k_i$ for $k_i \geq s$ and $\sum_{i=1}^t k_i = n$ and $\mathcal{U} = \mathcal{P}^T \begin{bmatrix} \mathsf{D}+\mathsf{TS} \\ 0_{(n-r)\times n}\end{bmatrix}$.*

According to (7), the reconstruction of the initial matrix A, from the compact Bruhat generators, writes

$$\mathsf{A} = (\mathsf{D}_{\mathcal{L}} + \mathsf{S}_{\mathcal{L}}\mathsf{T}_{\mathcal{L}})\mathsf{R}(\mathsf{D}_{\mathcal{U}} + \mathsf{T}_{\mathcal{U}}\mathsf{S}_{\mathcal{U}}) \quad (8)$$

where R is the leading $r \times r$ sub-matrix of $\mathcal{Q}^T\mathcal{E}^T\mathcal{P}^T$. As it has full rank, it is a permutation matrix.

This factorization is a compact version of the generalized Bruhat decomposition [13, 7]: the left factor is a column echelon form, the right factor a row echelon form.

5. COST OF COMPUTING WITH THE NEW GENERATORS

5.1 Computation of the generators

5.1.1 *The binary tree generators*

Let $T_1(n,s)$ denote the cost of the computation of the binary tree generator for an $n\times n$ matrix of order of quasiseparability s. It satisfies the recurrence relation $T_1(n,s) = K_\omega s^{\omega-2}\left(\frac{n}{2}\right)^2 + 2T_1(n/2,s)$, which solves in

$$T(n,s) = \frac{K_\omega}{2}s^{\omega-2}n^2 \text{ with } K_\omega = \frac{2^{\omega-2}}{(2^\omega-2)(2^{\omega-2}-1)}\mathsf{C}_\omega$$

where C_ω is the leading constant of the complexity of matrix multiplication [5].

5.1.2 *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 $\mathsf{A} = \begin{bmatrix}\mathsf{A}_1 & \mathsf{A}_2\\ \mathsf{A}_3 & \end{bmatrix}$ where $\mathsf{A}_3$ is $\lfloor\frac{n}{2}\rfloor \times \lfloor\frac{n}{2}\rfloor$
 3: Decompose $\mathsf{A}_1 = \mathsf{P}_1\begin{bmatrix}\mathsf{L}_1\\ \mathsf{M}_1\end{bmatrix}\begin{bmatrix}\mathsf{U}_1 & \mathsf{V}_1\end{bmatrix}\mathsf{Q}_1$        ▷ PLUQ($\mathsf{A}_1$)
 4: $\mathsf{R}_1 \leftarrow \mathsf{P}_1\begin{bmatrix}\mathsf{I}_{r_1} & \\ & 0\end{bmatrix}\mathsf{Q}_1$ where $r_1 = \mathrm{rank}(\mathsf{A}_1)$.
 5: $\begin{bmatrix}\mathsf{B}_1\\ \mathsf{B}_2\end{bmatrix} \leftarrow \mathsf{P}_1{}^T\mathsf{A}_2$        ▷ PermR($\mathsf{A}_2, \mathsf{P}_1{}^T$)
 6: $\begin{bmatrix}\mathsf{C}_1 & \mathsf{C}_2\end{bmatrix} \leftarrow \mathsf{A}_3\mathsf{Q}_1{}^T$        ▷ PermC($\mathsf{A}_3, \mathsf{Q}_1{}^T$)
 7: Here $A = \left[\begin{array}{cc|c}\mathsf{L}_1\backslash\mathsf{U}_1 & \mathsf{V}_1 & \mathsf{B}_1\\ \mathsf{M}_1 & 0 & \mathsf{B}_2\\ \hline \mathsf{C}_1 & \mathsf{C}_2 & \end{array}\right]$.
 8: $\mathsf{D} \leftarrow \mathsf{L}_1{}^{-1}\mathsf{B}_1$        ▷ TRSM($\mathsf{L}_1, \mathsf{B}_1$)
 9: $\mathsf{E} \leftarrow \mathsf{C}_1\mathsf{U}_1{}^{-1}$        ▷ TRSM($\mathsf{C}_1, \mathsf{U}_1$)
10: $\mathsf{F} \leftarrow \mathsf{B}_2 - \mathsf{M}_1\mathsf{D}$        ▷ MM($\mathsf{B}_2, \mathsf{M}_1, \mathsf{D}$)
11: $\mathsf{G} \leftarrow \mathsf{C}_2 - \mathsf{E}\mathsf{V}_1$        ▷ MM($\mathsf{C}_2, \mathsf{E}, \mathsf{V}_1$)
12: Here $\mathsf{A} = \left[\begin{array}{cc|c}\mathsf{L}_1\backslash\mathsf{U}_1 & \mathsf{V}_1 & \mathsf{D}\\ \mathsf{M}_1 & 0 & \mathsf{F}\\ \hline \mathsf{E} & \mathsf{G} & \end{array}\right]$.
13: $\mathsf{H} \leftarrow \mathsf{P}_1\begin{bmatrix}0_{r_1\times\frac{n}{2}}\\ \mathsf{F}\end{bmatrix}$
14: $\mathsf{I} \leftarrow \begin{bmatrix}0_{r_1\times\frac{n}{2}} & \mathsf{G}\end{bmatrix}\mathsf{Q}_1$
15: $(\mathcal{L}_2,\mathcal{E}_2,\mathcal{U}_2) \leftarrow$ LT-Bruhat(H)
16: $(\mathcal{L}_3,\mathcal{E}_3,\mathcal{U}_3) \leftarrow$ LT-Bruhat(I)
17: $\mathcal{L} \leftarrow \mathrm{Left}\left(\begin{bmatrix}\mathsf{P}_1 & \\ & \mathsf{I}_{\frac{n}{2}}\end{bmatrix}\begin{bmatrix}\mathsf{L}_1 & \\ \mathsf{M}_1 & 0\\ \mathsf{E} & 0\end{bmatrix}\begin{bmatrix}\mathsf{Q}_1 & \\ & \mathsf{I}_{\frac{n}{2}}\end{bmatrix}\right) + \begin{bmatrix}0 & \mathcal{L}_2\\ \mathcal{L}_3 & \end{bmatrix}$
18: $\mathcal{U} \leftarrow \begin{bmatrix}\mathsf{P}_1\begin{bmatrix}\mathsf{U}_1 & \mathsf{V}_1\\ 0 & 0\end{bmatrix}\mathsf{Q}_1 & \mathrm{Left}(\mathsf{P}_1\begin{bmatrix}\mathsf{D}\\ 0\end{bmatrix})\\ 0 & 0\end{bmatrix} + \begin{bmatrix}0 & \mathcal{U}_2\\ \mathcal{U}_3 & \end{bmatrix}$
19: $\mathcal{E} \leftarrow \begin{bmatrix}\mathcal{E}_1 & \mathcal{E}_2\\ \mathcal{E}_3 & \end{bmatrix}$
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 $\mathcal{E}$ 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 $H = P_2L_2U_2Q_2$ and $I = P_3L_3U_3Q_3$ 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

$$\mathcal{U}_2 = \mathrm{Left}(P_2 \begin{bmatrix} U_2 \\ 0 \end{bmatrix} Q_2), \qquad \mathcal{U}_3 = \mathrm{Left}(P_3 \begin{bmatrix} U_3 \\ 0 \end{bmatrix} Q_3),$$
$$\mathcal{L}_2 = \mathrm{Left}(P_2 \begin{bmatrix} L_2 & 0 \end{bmatrix} Q_2), \quad \mathcal{L}_3 = \mathrm{Left}(P_3 \begin{bmatrix} L_3 & 0 \end{bmatrix} Q_3).$$

At step 7, we have

$$\begin{bmatrix} A_1 & A_2 \\ A_3 & * \end{bmatrix} = \begin{bmatrix} P_1 & \\ & I_{\frac{n}{2}} \end{bmatrix} \left[\begin{array}{cc|c} L_1 & & \\ M_1 & I_{\frac{n}{2}-r_1} & \\ \hline E & 0 & I_{\frac{n}{2}} \end{array}\right] \times$$
$$\left[\begin{array}{cc|c} U_1 & V_1 & D \\ & 0 & F \\ \hline & G & \end{array}\right] \begin{bmatrix} Q_1 & \\ & I_{\frac{n}{2}} \end{bmatrix}$$

As the first r_1 rows of $P_1^T H$ are zeros, there exists $\bar{P}_2$ a permutation matrix and $\bar{L}_2$, a lower triangular matrix, such that $P_1^T P_2 L_2 = \begin{bmatrix} 0_{r_1 \times \frac{n}{2}} \\ \bar{P}_2 \bar{L}_2 \end{bmatrix}$. Similarly, there exsist $\bar{Q}_3$, a permutation matrix and $\bar{U}_3$, an upper triangular matrix, such that $U_3 Q_3 Q_1^T = \begin{bmatrix} 0_{\frac{n}{2} \times r_1} & \bar{U}_3 \bar{Q}_3 \end{bmatrix}$. Hence

$$\begin{bmatrix} A_1 & A_2 \\ A_3 & * \end{bmatrix} = \begin{bmatrix} P_1 & \\ & P_3 \end{bmatrix} \left[\begin{array}{cc|c} L_1 & & \\ M_1 & \bar{P}_2 \bar{L}_2 & \\ \hline P_3^T E & 0 & L_3 \end{array}\right] \times$$
$$\left[\begin{array}{cc|c} U_1 & V_1 & DQ_2^T \\ & 0 & U_2 \\ & \bar{U}_3 \bar{Q}_3 & \end{array}\right] \begin{bmatrix} Q_1 & \\ & Q_2 \end{bmatrix}$$

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

$$\begin{bmatrix} A_1 & A_2 \\ A_3 & * \end{bmatrix} = \begin{bmatrix} P_1 \begin{bmatrix} I_{r_1} & \\ & \bar{P}_2 \end{bmatrix} & \\ & P_3 \end{bmatrix} \left[\begin{array}{ccc} L_1 & & \\ N_1 & \bar{L}_2 & \\ \hline E & 0 & L_3 \end{array}\right] \times$$
$$\left[\begin{array}{cc|c} U_1 & W_1 & DQ_2^T \\ & 0 & U_2 \\ & \bar{U}_3 & \end{array}\right] \begin{bmatrix} \begin{bmatrix} I_{r_1} & \\ & \bar{Q}_3 \end{bmatrix} Q_1 & \\ & Q_2 \end{bmatrix}.$$

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 cylic-shift on the second factor and a column block cyclic shift on the third factor as in [5, Algorithm 1].

Finally,

$$P \begin{bmatrix} U \\ 0 \end{bmatrix} Q = \begin{bmatrix} P_1 & \\ & I_{\frac{n}{2}} \end{bmatrix} \begin{bmatrix} U_1 & V_1 & D \\ & 0 & \bar{P}_2 U_2 Q_2 \\ & P_3 \bar{U}_3 \bar{Q}_3 & \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} Q_1 & \\ & I_{\frac{n}{2}} \end{bmatrix}$$
$$= \begin{bmatrix} P_1 \begin{bmatrix} U_1 & V_1 \\ 0 & 0 \end{bmatrix} Q_1 & P_1 \begin{bmatrix} D \\ 0 \end{bmatrix} \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} & P_2 \begin{bmatrix} U_2 \\ 0 \end{bmatrix} Q_2 \\ P_3 \begin{bmatrix} U_3 \\ 0 \end{bmatrix} Q_3 & \end{bmatrix}.$$

Hence

$$\mathrm{Left}(PUQ) = \begin{bmatrix} P_1 \begin{bmatrix} U_1 & V_1 \\ 0 & 0 \end{bmatrix} Q_1 & \mathrm{Left}(P_1 \begin{bmatrix} D \\ 0 \end{bmatrix}) \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} & \mathcal{U}_2 \\ \mathcal{U}_3 & \end{bmatrix}.$$

The complexity analysis is exactly that of Theorem 2. □

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 $T_{\mathrm{RL}}(n, s)$ 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:

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

Let $T_{\mathrm{PL}}(n, s)$ 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

$$T_{\mathrm{PL}}(n, s) \quad = \quad T_{\mathrm{RL}}(n, s) + O(n^2 s^{\omega-2}) = O(n^2 s^{\omega-2}).$$

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

$$\begin{aligned} T_{\mathrm{LL}}(n, s) &= 2T_{\mathrm{LL}}(n/2, s) + 2T_{\mathrm{PL}}(n/2, s) + O(n^2 s^{\omega-2}) \\ &= O(n^2 s^{\omega-2}) \end{aligned}$$

5.3.2 *The Bruhat generator*

Using the decomposition (8), the product of two left triangular matrices writes $A \times B = C_A R_A E_A \times C_B R_B E_B$ where $C_X = D_{\mathcal{L}_X} + S_{\mathcal{L}_X} T_{\mathcal{L}_X}$ and $E_X = D_{\mathcal{U}_X} + T_{\mathcal{U}_X} S_{\mathcal{U}_X}$ for $X \in \{A, B\}$. We will compute it using the following parenthesizing:

$$A \times B = C_A(R_A(E_A \times C_B)R_B)E_B. \qquad (9)$$

The product $E_A \times C_B = (D_{\mathcal{U}_A} + T_{\mathcal{U}_A} S_{\mathcal{U}_A})(D_{\mathcal{L}_B} + S_{\mathcal{L}_B} T_{\mathcal{L}_B})$ 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

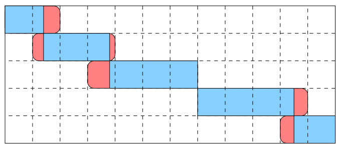

Figure 3: Aligning a block diagonal matrix (blue) on an $s \times s$ grid. Each block row of the aligned structure (red) may overlap with the previous and next block rows on at most one $s \times s$ tile on each side.

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 $O(s^{\omega-1}n)$ which is consequently also the cost of computing the product $\mathsf{E_A C_B}$. After left and right multiplication by the permutations $\mathsf{R_A}$ and $\mathsf{R_B}$, this $r_\mathsf{A} \times r_\mathsf{B}$ dense matrix is multiplied to the left by $\mathsf{C_A}$. This costs $O(nr_B s^{\omega-2})$. Lastly, the right multiplication by E_B of the resulting $n \times r_\mathsf{A}$ matrix costs $O(n^2 s^{\omega-2})$ 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 $\mathsf{A} \times \mathsf{B} = (\mathsf{L_A} + \mathsf{D_A} + \mathsf{U_A})(\mathsf{L_B} + \mathsf{D_B} + \mathsf{U_B})$. 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 shows in Table 1. The complexity of section 5.3 directly applies for the computation of Upper × Lower and Lower × Upper products. For the other products, a J_n factor has to be added between the $\mathsf{E_A}$ and $\mathsf{C_B}$ factors in the innermost product of (9). As reverting the row order of $\mathsf{C_B}$ does not impact the cost of computing this product, the same complexity applies here too.

×	Lower	Upper
Lower	J_n × Left × J_n × Left	J_n × Left × Left × J_n
Upper	Left × J_n × J_n × Left	Left × J_n × Left × J_n

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

THEOREM 4. *Mutliplying two quasiseparable matrices of order respectively $(l_\mathsf{A}, u_\mathsf{A})$ and $(l_\mathsf{B}, u_\mathsf{B})$ costs $O(n^2 s^{\omega-2})$ field operations where $s = \max(l_\mathsf{A}, u_\mathsf{A}, l_\mathsf{B}, u_\mathsf{B})$, 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 $O(n^2 s^{\omega-2})$ for $s = \max(l_\mathsf{A}, u_\mathsf{A}, l_\mathsf{B}, u_\mathsf{B})$. However, the product is also a quasiseparable matrix, of order $(l_\mathsf{A} + l_\mathsf{B}, u_\mathsf{A} + u_\mathsf{B})$ [8, Theorem 4.1], which can be represented by a Bruhat generator with only $O(n(l_\mathsf{A} + l_\mathsf{B} + u_\mathsf{A} + u_\mathsf{B}))$ coefficients. A first natural question is thus to find an algorithm computing this representation from the generators of A and B in time $O(ns^{\omega-1})$.

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

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 $\mathsf{E_A} \times \mathsf{C_B}$ 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

[1] P. Boito, Y. Eidelman, and L. Gemignani. Implicit QR for companion-like pencils. *Math. of Computation*, 85(300):1753–1774, 2016.

[2] Tony F. Chan. Rank revealing QR factorizations. *Linear Algebra and its Applications*, 88:67–82, April 1987.

[3] S. Chandrasekaran and I. Ipsen. On Rank-Revealing Factorisations. *SIAM Journal on Matrix Analysis and Applications*, 15(2):592–622, April 1994.

[4] S. Delvaux and M. Van Barel. A Givens-Weight Representation for Rank Structured Matrices. *SIAM J. on Matrix Analysis and Applications*, 29(4):1147–1170, November 2007.

[5] Jean-Guillaume Dumas, Clément Pernet, and Ziad Sultan. Simultaneous computation of the row and column rank profiles. In Manuel Kauers, editor, *Proc. ISSAC'13*, pages 181–188. ACM Press, 2013.

[6] Jean-Guillaume Dumas, Clément Pernet, and Ziad Sultan. Computing the rank profile matrix. In *Proc. ISSAC'15*, pages 149–156, New York, NY, USA, 2015. ACM.

[7] Jean-Guillaume Dumas, Clément Pernet, and Ziad Sultan. Fast computation of the rank profile matrix and the generalized bruhat decomposition. Technical report, 2015. `arXiv:1601.01798`.

[8] Y. Eidelman and I. Gohberg. On a new class of structured matrices. *Integral Equations and Operator Theory*, 34(3):293–324, September 1999.

[9] Yuli Eidelman, Israel Gohberg, and Vadim Olshevsky. The QR iteration method for hermitian quasiseparable matrices of an arbitrary order. *Linear Algebra and its Applications*, 404:305 – 324, 2005.

[10] Tsung-Min Hwang, Wen-Wei Lin, and Eugene K. Yang. Rank revealing LU factorizations. *Linear Algebra and its Applications*, 175:115–141, October 1992.

[11] K Lessel, M. Hartman, and Shivkumar Chandrasekaran. A fast memory efficient construction algorithm for hierarchically semi-separable representations. Technical report, 2015. http://scg.ece.ucsb.edu/publications/MemoryEfficientHSS.pdf.

[12] Gennadi Ivanovich Malaschonok. Fast generalized Bruhat decomposition. In *CASC'10*, volume 6244 of *LNCS*, pages 194–202. Springer-Verlag, Berlin, Heidelberg, 2010.

[13] Wilfried Manthey and Uwe Helmke. Bruhat canonical form for linear systems. *Linear Algebra and its Applications*, 425(2–3):261 – 282, 2007. Special Issue in honor of Paul Fuhrmann.

[14] C.-T. Pan. On the existence and computation of rank-revealing LU factorizations. *Linear Algebra and its Applications*, 316(1–3):199–222, September 2000.

[15] The LinBox Group. *LinBox: Linear algebra over black-box matrices*, v1.4.1 edition, 2016. http://linalg.org/.

[16] R. Vandebril, M. Van Barel, G. Golub, and N. Mastronardi. A bibliography on semiseparable matrices. *CALCOLO*, 42(3):249–270, 2005.

[17] Raf Vandebril, Marc Van Barel, and Nicola Mastronardi. *Matrix computations and semiseparable matrices: linear systems*, volume 1. The Johns Hopkins University Press, 2007.

[18] Jianlin Xia, Shivkumar Chandrasekaran, Ming Gu, and Xiaoye S. Li. Fast algorithms for hierarchically semiseparable matrices. *Numerical Linear Algebra with Applications*, 17(6):953–976, December 2010.