LBfM Lehrstuhl B für Mathematik | RWTH AACHEN UNIVERSITY

Bachelor thesis

Bruhat Decomposition in Unitary and Symplectic Groups over Finite Fields

Daniel Rademacher

August 30, 2019

supervised by Prof. Dr. Alice Niemeyer
and Dominik Bernhardt M. Sc.

The present work was submitted to

Lehrstuhl B für Mathematik

RWTH Aachen University

Contents

1 Introduction

In this bachelor thesis, we pursue two goals whose results are relevant to a larger project from the area of computational group theory, namely the matrix group recognition problem [MGR]. The first goal is to find a Bruhat decomposition for the unitary and symplectic group over finite fields and to prove the correctness of an algorithm for computing these decompositions.
Then our second goal is to be able to write every element of the symplectic and unitary group as a word in specific generators, namely the Leedham-Green-O'Brien standard generators (LGO) [GB]. The basic procedure is described in the paper "Straight-line programs with memory and matrix Bruhat decomposition" by A. C. Niemeyer, T. Popiel and C. E. Praeger [NPP].

Before we embark on describing how to achieve our goals, we use the second chapter to give basic definitions and to fix notation. In the third chapter we recapitulate the paper "Straight-line programs with memory and matrix Bruhat decomposition" by A. C. Niemeyer, T. Popiel and C. E. Praeger [NPP] and discuss the general approach. In the fourth chapter, we consider the unitary group. In the first three parts of this chapter we see that the Bruhat decomposition of the special linear group consisting of monomial matrices and lower unitriangular matrices also exists for the unitary group. We also want to realize this decomposition there. For this we use a monomorphism from the special linear group into the unitary group to get a general impression of the matrices that implement the desired decomposition. However, since these are not sufficient, we will look for more matrices that would be useful and show that they lie in the unitary group. The algorithm requires a case distinction, depending on whether the dimension of the underlying vector space is even or odd. If the dimension is even, computing the Bruhat decomposition is easier, so we deal with this situation first and then extend the algorithm to the case with odd dimension. In the final part of the fourth chapter, we express every element of the special unitary group as a word of specified generators. Our specified generators are to become the LGO standard generators. Therefore, we look at the LGO standard generators and use a base change to transform the matrices into a form that matches our previous definition. Then we start with the monomial matrices from the algorithm for the Bruhat decomposition. We can assume that we can find two unitriangular matrices b_1, b_2 such that for a matrix a in the unitary group we have

$$b_1 \cdot a \cdot b_2 = m$$

where m is a monomial matrix in the same group. So if we can write the matrices b_1, b_2 and m in terms of standard generators, we know exactly what a looks like, since

$$a = b_1^{-1} \cdot m \cdot b_2^{-1}.$$

We therefore start with the matrices b_1 and b_2. Since b_1 and b_2 are the product of matrices computed by the Bruhat decomposition algorithm, we look at the matrices used there, write them and can thus directly represent b_1 and b_2.
We know that it is also necessary to be able to describe monomial matrices in the unitary group as words. To achieve this we proceed again as in the paper "Straight-line programs with memory and matrix Bruhat decomposition" by A. C. Niemeyer, T. Popiel and C. E. Praeger [NPP] and write a monomial matrix m' in terms of the LGO generators such that

$$m' \cdot m = h$$

where h is a diagonal matrix. Finally, we write the diagonal matrix h as a product of the LGO generators and obtain

$$a = b_1^{-1} \cdot m'^{-1} \cdot h \cdot b_2^{-1}.$$

The fifth chapter deals with the symplectic group and is structured analogously to the fourth chapter. The results for these groups are slightly different, but the procedure and the methods are for the most part identical.

All algorithms were implemented by me in GAP during this bachelor thesis for the first time and are available in the package "BruhatDecomposition". I also analysed the maximum length and number of slots needed for the constructed MSLP of the implementation. The main theorems about the existence of an MSLP for the Bruhat decomposition are Theorem 4.34 for the unitary group with even dimension, Theorem 4.36 for the unitary group with odd dimension and Theorem 5.24 for the symplectic group.

To show the connection to the matrix group recognition problem, we consider the following situation. Let $a_1, \ldots, a_k \in \mathrm{GL}(n, q)$ and $G = \langle a_1, \ldots, a_k \rangle$. The aim of the matrix group recognition project is to get information about a matrix group like G, such as the order. We also want to be able to decide if a matrix $a \in \mathrm{GL}(n, q)$ is in the matrix group G and, if so, to write a as a word in the generators $a_1, \ldots, a_k$.
In order to solve this problem, Aschbacher's Theorem is employed to seek a homomorphism from a given group into a group lying in one of the Aschbacher families. The problem of determining information about the original group G is then reduced to determining information about two new groups, namely the image of G under the homomorphism and the kernel. This process can be repeated with the two new groups, forming a tree structure. The process ends when only certain groups remain as leaves in which the classical groups play an important role. To decide whether any matrix is in the special unitary group or the symplectic group, and if so, how to write the element with particular generators, can be solved with the results of this thesis.

This problem has also been solved and implemented for the classical groups by Csaba Schneider in Magma with the work of Elliot Costi. However, they used a completely different method which did not involve the Bruhat decomposition.

Index of Symbols

Symbol	Meaning
G, H, U	Groups
g	Element of a group
$\varphi, \phi, \tau, \mu, \kappa, \theta$	Maps
Φ	Bilinear form
$\mathrm{Bifo}(V)$	The set of all bilinear forms on a vector space V
$^-$	A field automorphism of order 2
V	Vector space
B	Basis of a vector space
GL	General linear group
$\mathrm{End}(V)$	Group of homomorphisms from V to V
SL	Special linear group
U	Unitary group
SU	Special unitary group
Sp	Symplectic group
a, b, c	Matrices
I_n	$\mathrm{diag}(1, \ldots, 1)$
$E_{i,j}$	Has a 0 in every position except for a 1 in position (i, j)
n	Dimension of a vector space
m	Mostly $n = 2m$ or $n = 2m + 1$
$e_1, \ldots, e_n$	Standard basis of $\mathbb{F}_q^n$
h	Diagonal matrix
p	Prime number
f	Natural number (Mostly $q = p^f$)
q	Prime power
ν, w, z	Elements of a vector space
$\mathbb{F}$	A finite field
$\mathbb{F}_0$	The fixed field of $\mathbb{F}$ of the field automorphism $^-$
$\imath, \jmath$	Elements of a field
ω, γ	Primitive elements of a field
ℓ	Commonly used as an exponent
$a_{i,-}$	ith row of a
$a_{-,j}$	jth column of a
$\cdot^L$	Multiplication from left
$\cdot^R$	Multiplication from right
$I_{i,j}(\imath)$	Transvection in the special linear group
$T_{i,j}(\imath)$	Transvection in the special unitary group
$S_{i,j}(\imath)$	Transvection in the symplectic group
s, t, δ, u, v, x, y	LGO standard generators (see Chapter 4.4 and Chapter 5.4)
J_n	Gram-matrix in special unitary group
P_n	Gram-matrix in symplectic group
$M - N$	$M \setminus N$ for sets M and N
v^φ	$\varphi(v)$
V^*	Dual space of a vector space V
i, j, k	Indices

2 Background

This chapter introduces basic definitions and expressions that are important in all subsequent chapters. First, we discuss the mathematical aspects, while the second section of this chapter deals with the programming side.

2.1 Preliminaries

This subsection introduces the basic mathematical definitions and expressions of this thesis.

First we give some definitions that are needed later to describe the classical groups. For the remainder of this chapter $\mathbb{F}$ denotes a finite field of order q.

Definition 2.1: Let V be an $\mathbb{F}$-vector space.
A map $\Phi\colon V \times V \to \mathbb{F}, (\nu, w) \mapsto \Phi(\nu, w)$ is called a *bilinear form* on V, if Φ is linear in each component, that means

(i) $\Phi(\imath\nu_1 + \jmath\nu_2, w) = \imath \cdot \Phi(\nu_1, w) + \jmath \cdot \Phi(\nu_2, w)$ for all $\imath, \jmath \in \mathbb{F}$ and $\nu_1, \nu_2, w \in V$,

(ii) $\Phi(\nu, \imath w_1 + \jmath w_2) = \imath \cdot \Phi(\nu, w_1) + \jmath \cdot \Phi(\nu, w_2)$ for all $\imath, \jmath \in \mathbb{F}$ and $\nu, w_1, w_2 \in V$.

We define $\mathrm{Bifo}(V)$ as the set of all bilinear forms on V.

Remark 2.2: In the following we will refer to a bilinear form only as a *form*.

In order to define the unitary group, one needs a unitary form and a field automorphism of order 2.

Definition 2.3: Let $\mathbb{F}$ be a field with a field automorphism $\bar{}$ of order 2 and V an $\mathbb{F}$-vector space. A map $\Phi\colon V \times V \to \mathbb{F}, (\nu, w) \mapsto \Phi(\nu, w)$ is called a *unitary form* on V, if

(i) $\Phi(\imath\nu_1 + \jmath\nu_2, w) = \imath \cdot \Phi(\nu_1, w) + \jmath \cdot \Phi(\nu_2, w)$ for all $\imath, \jmath \in \mathbb{F}$ and $\nu_1, \nu_2, w \in V$,

(ii) $\Phi(\nu, w) = \overline{\Phi(w, \nu)}$ for all $\nu, w \in V$.

The symplectic group can be defined in terms of bilinear forms with certain properties.

Definition 2.4: Let V be an $\mathbb{F}$-vector space.

1.) A bilinear form $\Phi \in \mathrm{Bifo}(V)$ or a unitary form Φ is called *non-singular* or *non-degenerate* if for each $\nu \in V - \{0\}$ there is a $w \in V$ with $\Phi(\nu, w) \neq 0$.

2.) A bilinear form $\Phi \in \mathrm{Bifo}(V)$ is called *alternating* if $\Phi(\nu, \nu) = 0$ for all $\nu \in V$.

3.) A non-singular and alternating bilinear form $\Phi \in \mathrm{Bifo}(V)$ is called *symplectic*.

There are special endomorphisms on an $\mathbb{F}$-vector space with a bilinear form, which are used to define the classical groups. To define these, we need another definition beforehand.

Definition 2.5: Let V be an n-dimensional $\mathbb{F}$-vector space. Define

1.) $\mathrm{End}(V) \coloneqq \{\varphi \mid \varphi\colon V \to V,\ \varphi \text{ is a vector space homormorphism}\}$,

2.) $\mathrm{GL}(V) \coloneqq \{\varphi \mid \varphi\colon V \to V,\ \varphi \text{ is a homormorphism and bijective}\}$,

3.) $\mathrm{GL}(n,q) \coloneqq \{a \in \mathbb{F}_q^{n\times n} \mid a \text{ is invertible}\}$.

Remark 2.6:

1.) Notice that $\mathrm{GL}(n,q)$ is well-defined since every finite field of order $q = p^f$ is unique up to isomorphism.

2.) $\mathrm{End}(V)$ and $\mathrm{GL}(V)$ are groups with composition. $\mathrm{GL}(n,q)$ is a group with matrix multiplication.

3.) Note that in the whole thesis we act from the right.

There is an important connection between these groups.

Theorem 2.7: Let V be an n-dimensional $\mathbb{F}$-vector space. Let $B \in V^n$ be a basis of V. Then the map

$$\mathrm{End}(V) \to \mathbb{F}^{n\times n}, \varphi \mapsto {}^B\varphi^B$$

is an isomorphism.

Corollary 2.8: Let V be an n-dimensional $\mathbb{F}$-vector space. Let $B \in V^n$ be a basis of V. Then

$$\mathrm{GL}(V) \to \mathrm{GL}(n,q), \varphi \mapsto {}^B\varphi^B$$

is an isomorphism.

We are now in a position to define the special endomorphisms.

Definition 2.9: Let (V, Φ) be an $\mathbb{F}$-vector space equipped with a form Φ.
A map $\varphi \in \mathrm{End}(V)$ is called an *isometry* from V to V if $\Phi(\nu^\varphi, w^\varphi) = \Phi(\nu, w)$ for all $\nu, w \in V$.
The group of all bijective isometries of (V, Φ) is denoted by $I(V, \mathbb{F}, \Phi)$.
The subgroup $S(V, \mathbb{F}, \Phi) = I(V, \mathbb{F}, \Phi) \cap \mathrm{SL}(V)$ is called the group of *special isometries* of V.

At this point it is possible to define the classical groups SU and Sp.

Definition 2.10: Let V be an n-dimensional $\mathbb{F}_q$-vector space with a non-singular form Φ.

1.) If Φ is symplectic then the group $I(V, \mathbb{F}_q, \Phi)$ of bijective isometries is equal to the group $S(V, \mathbb{F}_q, \Phi)$ of all special isometries and is called the *symplectic group* and denoted $\mathrm{Sp}(V)$. (The fact that $I(V, \mathbb{F}_q, \Phi)$ and $S(V, \mathbb{F}_q, \Phi)$ are equal can be found in Corollary 8.6 [Taylor].)

2.) If $\mathbb{F}_{q^2}$ is a field admitting a field automorphism $\bar{}$ of order 2, $\mathrm{Fix}(\mathbb{F}_{q^2}) = \mathbb{F}$ and Φ is a unitary form then the group $I(V', \mathbb{F}_{q^2}, \Phi)$ of bijective isometries is called the *unitary group* $\mathrm{U}(V)$ where V' is an n-dimensional $\mathbb{F}_{q^2}$-vector space. The subgroup $S(V', \mathbb{F}_{q^2}, \Phi)$ of all invertible special isometries is called *special unitary group* and denoted by $\mathrm{SU}(V)$.

Remark 2.11:

1.) From now on, we only consider non-singular bilinear forms.

2.) Note that the unitary and symplectic group is independent of the chosen bilinear form. [Taylor, page 69 and page 116/117]

To understand the Bruhat decomposition, it is also important to define (B, N) pairs.

Definition 2.12: Let G be a group. A *(B, N) pair* is a pair of subgroups B and N of G such that the all of following conditions hold:

1.) $G = BN$,

2.) $H = B \cap N$ is a normal subgroup of N,

3.) The group $W = N/H$ is generated by elements $\{w_i \mid i \in I\}$ such that $w_i^2 = 1$ for all $i \in I$ and $I \neq \emptyset$ is an index set,

4.) For $i \in I$ and any element $w \in W$ we have $w_i B w \subseteq B w_i w B \cup B w B$,

5.) No generator w_i for $i \in I$ normalizes B.

This definition looks complicated, so we give a brief example in order to understand (B, N) pairs better. We need two more definitions for this example, which are also needed later.

Definition 2.13: Let $n \in \mathbb{N}$ and $a \in \mathbb{F}^{n\times n}$. Then a is called a *monomial matrix* if there is exactly one non-zero element in each row and column of a.

Definition 2.14: Let $n \in \mathbb{N}$ and $a \in \mathbb{F}^{n\times n}$. Then a is called a *lower-unitriangular matrix* if both of the following conditions hold:

(i) $a_{i,i} = 1$ for all $i \in \{1, 2, \ldots, n\}$,

(ii) $a_{i,j} = 0$ for all $i, j \in \{1, 2, \ldots, n\}$ and $j > i$.

Example 2.15: Let $n \in \mathbb{N}$, V an n-dimensional $\mathbb{F}$-vector space, $G = \mathrm{GL}(V)$ the general linear group and $a \in G$.
As $\det(a) \neq 0$ each row and column of a contains a non-zero entry. Multiplying a matrix a by a lower-unitriangular matrix from left effects elementary row operations and from right effects elementary column operations. Using the Gaussian elimination procedure, every matrix in G can be transformed into reduced row echelon form using only elementary row and column operations. Such a matrix has only one non-zero entry in each row and column, so it is a monomial matrix. In particular, there are unitriangular matrices $b_1, b_2 \in G$ such that $b_1 a b_2 = w$ where w is a monomial matrix.
Let B be the group of lower-unitriangular matrices and N the group of monomial matrices. It follows that (B, N) is a (B, N) pair for G.

Finally we can define the Bruhat decomposition.

Definition 2.16: Let G be a group with a (B, N) pair, $H = B \cap N$ and $W = N/H$. The *Bruhat decomposition* of G is the decomposition of G into

$$G = BWB = \prod_{w\in W} BwB.$$

We now define transvections in a group G. Transvections take the role of elementary row and column operations in the Gaussian elimination. The proof of Lemma 3.2 is a good example of this.

Definition 2.17: Let V be an n-dimensional $\mathbb{F}$-vector space. Let $H \leq V$ such that $\dim(H) = n - 1$. An element $\tau \in \mathrm{GL}(V)$ is called *transvection* with respect to the *hyperplane* H if

1.) $w^\tau = w$ for all $w \in H$,

2.) $\nu^\tau - \nu \in H$ for all $\nu \in V$.

Transvections play an important role as they generate the SL, SU and Sp. We show this for SL and Sp at the beginning of the corresponding chapter and give a reference to the prove for SU. This gives us an idea of how to construct a (B, N) pair for the group.
In order to express elements of a group in terms of particular generators, it is only possible to work with transvections which lie in the corresponding group.

Definition 2.18: Let V be an n-dimensional $\mathbb{F}$-vector space and $\tau \in \mathrm{GL}(V)$ be a transvection.

1.) τ is called a *symplectic transvection* if $\tau \in \mathrm{Sp}(V)$.

2.) τ is called a *unitary transvection* if $\tau \in \mathrm{SU}(V)$.

Transvections can be completely characterized as follows:

Lemma 2.19: Let V be an n-dimensional $\mathbb{F}$-vector space. Let $\tau \in \mathrm{GL}(V)$ be a transvection with respect to the hyperplane $H \leq V$. Let $\mu \colon V \to \mathbb{F} \in V^*$ with $H = \ker(\mu)$. Then the following holds:

1.) There is a vector $u \in H - \{0\}$ such that $\nu^\tau = \nu - \nu^\mu \cdot u$ for all $\nu \in V$.

2.) If $z \in V - \{0\}$ with $z \in \ker(\mu)$, then $\tau_{z,\mu} \colon V \to V, \nu \mapsto \nu - \nu^\mu \cdot z$ is a transvection.

Proof. [Taylor, page 20]. □

When working with transvections, the following properties are commonly used.

Lemma 2.20: Let V be an n-dimensional $\mathbb{F}$-vector space. Let $H \leq V$ be a hyperplane and $\mu_1, \mu_2 \in V^*$ with $\ker(\mu_1) = \ker(\mu_2) =: U$, $\jmath \in \mathbb{F}^*$ and $z_1, z_2 \in H - \{0\}$. Then all of the following hold:

1.) $\tau_{z_1,\mu_1}\tau_{z_2,\mu_1} = \tau_{z_1+z_2,\mu_1}$,

2.) $\tau_{z_1,\mu_1}\tau_{z_1,\mu_2} = \tau_{z_1,\mu_1+\mu_2}$,

3.) $\tau_{\jmath z_1,\mu_1} = \tau_{z_1,\jmath\mu_1}$.

Proof. [Taylor, page 21 Theorem 4.2]. □

2.2 Straight Line Programs

In this subsection we present an efficient way to evaluate a group element in specified generators of the group using simple operations in the group. So it is possible to encode group elements as words over a given generating set, which can be evaluated efficiently.
We start by defining which operations can be used. There are no loops, no conditional statements, no comparisons or other special techniques from programming languages. We restrict ourselves to multiplying and inverting. (See [NPP].)

Remark 2.21: MSLPs and the Bruhat decomposition of the classical groups play an important role in the matrix group recognition project. In this project some matrices $a_1, \ldots, a_k \in \mathrm{GL}(n, q)$ over a finite field $\mathbb{F}_q$ are given. We consider the group $G = \langle a_1, \ldots, a_k \rangle$ and want to know the order of G, for example.
Moreover we could ask for a matrix $a \in \mathrm{GL}(n, q)$ whether $a \in G$ and if yes, how a can be expressed in terms of the matrices $a_1, \ldots, a_k$. For this, the results of this thesis can be used. More information about this project can be found in [MGR].

Definition 2.22: Let G be a group, $b \in \mathbb{N}_0$ and $M = [m_1, \ldots, m_b]$ an ordered list of b elements of G. A modification $\mathcal{I}$ of M is called an *instruction* if it has one of the following forms:

(i) $m_k \leftarrow m_i$ with $i, k \in \{1, \ldots, b\}$. This instruction stores m_i in the list M in slot k.

(ii) $m_k \leftarrow m_i \cdot m_j$ with $i, j, k \in \{1, \ldots, b\}$. This instruction stores $m_i \cdot m_j$ in the list M in slot k.

(iii) $m_k \leftarrow m_i^{-1}$ with $i, k \in \{1, \ldots, b\}$. This instruction stores m_i^{-1} in the list M in slot k.

(iv) Show(A) where $A \subseteq \{1, \ldots, b\}$. Here no action is required. Instead the specified slots are displayed.

A single instruction does not seem interesting, but we can define a sequence of instructions.

Definition 2.23: Let G be a group, $b \in \mathbb{N}_0$ and $M = [m_1, \ldots, m_b]$ an ordered list of b elements of G. A *straight-line program with memory* (MSLP) is a sequence $S = [\mathcal{I}_1, \ldots, \mathcal{I}_n]$ of instructions $\mathcal{I}_r$ with $1 \leq r \leq n$.
The number $b \in \mathbb{N}_0$ is called the *memory quota* of S and S is said to be a b-MSLP. The number $n \in \mathbb{N}_0$ is the length of S. The empty sequence is permitted with length 0.

In the evaluation process M is used as the memory for the elements that are needed. The instruction (i) of Definition 2.22 can be used to overwrite a slot and minimise the memory quota. The idea of an MSLP is to reduce the memory required for evaluating a particular

group word, by repeatedly overwriting memory slots which are no longer used. The length n of an MSLP describes the number of operations during the entire evaluation.

An obvious way to record a word is to simply store the sequence of elements in the word. This, however, is often not very efficient if we want to evaluate the word afterwards by multiplying the elements of the word by specific group elements, e.g. matrices. The aim of an SLP is to give an algorithm how to evaluate this word efficiently, for example by computing and storing subwords only once that occur repeatedly.

Remark 2.24: The instructions of Definition 2.22 use elements of $\{1, \ldots, b\}$. The memory M is secondary in the description. This implies that an MSLP is independent of the group and the chosen elements for the memory. So it is possible to execute exactly the same constructed steps from one group in another group. We use this later to transform a monomial matrix into a diagonal matrix.

The following example helps to understand the previous remark and MSLPs.

Example 2.25: In most computations with groups, elements need to be raised to some powers. To shorten this, fast exponentiation can be used.
Let G be a group, $g \in G$ and $n \in \mathbb{N}$. We can express n in binary form as $n = \sum_{i=0}^{m} a_i \cdot 2^i$ with $a_i \in \{0, 1\}$ for $0 \leq i \leq m$ and $m \in \mathbb{N}$. Then is

$$g^n = g^{(\sum_{i=0}^{m} a_i \cdot 2^i)} = \prod_{i=0}^{m} (g^{2^i})^{a_i}.$$

A way to construct an MSLP for fast exponentiation with memory quota 2 is given in Algorithm 1 in pseudo code.

For example, the algorithm produces the following MSLP for $n = 8$:

$$\begin{aligned} L = [&[m_2 \leftarrow g], [m_1 \leftarrow 1_G], [m_2 \leftarrow m_2 \cdot m_2], [m_2 \leftarrow m_2 \cdot m_2], \\ &[m_2 \leftarrow m_2 \cdot m_2], [m_1 \leftarrow m_1 \cdot m_2], [m_2 \leftarrow m_2 \cdot m_2], [m_1 \leftarrow m_1]] \end{aligned}$$

In this case we need 8 instructions which is as much as n. However, for a growing n, this variant is advisable since the length of the SLPs converges to $\log_2(n)$.
Now let G be $\mathbb{F}_5$ and $g = 2 \in G$. If we evaluate the MSLP L with g and 1_G, we get $g^8 = 1_G$ in slot 1. But we can also choose $G = C_9$ and $g = (1, \ldots, 9) \in G$. If we evaluate the MSLP L with g and (), we get $g^8 = g^{-1}$ in slot 1. So we can use a given MSLP on every group and every selection of elements for the memory.

Algorithm 1: Fast exponentiation

Input: $k \in \mathbb{N}_0$ a natural number
Output: An MSLP which computes g^k with fast exponentiation for a group element of G

function Power(k)
- $\mathcal{I}_1 := (m_2 \leftarrow g)$
- $\mathcal{I}_2 := (m_1 \leftarrow 1_G)$ // First two instructions to store g and 1_G of a group G
- $i := 3$
- **while** $k > 0$ **do**
  - **if** k *odd* **then**
    - $\mathcal{I}_i := (m_1 \leftarrow m_1 \cdot m_2)$ // The factor in the binary representation is not 0
    - $i = i + 1$
  - $\mathcal{I}_i := (m_2 \leftarrow m_2 \cdot m_2)$ // square the element in slot 2
  - $k := \text{Floor}(\frac{k}{2})$
  - $i := i + 1$
- $\mathcal{I}_i = (m_1 \leftarrow m_1)$
- **return** $[\mathcal{I}_1, \ldots, \mathcal{I}_i]$

Remark 2.26: Let $n, f \in \mathbb{F}$, $q = p^f$ a prime power, $G \in \{\mathrm{SU}(n,q), \mathrm{Sp}(n,q)\}$, $G = \langle a_1, \ldots, a_k \rangle$ and $a \in G$. Since $a \in G$, we can write a as a word in the generators $a_1, \ldots, a_k$ by multiplying these elements and their inverse elements. Therefore, we can compute an MSLP S that, if we evaluate S with the generators $a_1, \ldots, a_k$, we achieve a. The construction of such an MSLP while be referred as the word problem in the following.

3 Bruhat Decomposition in the Special Linear Group

In their paper "Straight-line programs with memory and matrix Bruhat decomposition" [NPP], A. C. Niemeyer, T. Popiel and C. E. Praeger described and implemented an algorithm to compute the Bruhat decomposition in the special linear group SL. (See also [Taylor].)
In the following section we want to describe the method they used because we use a similar strategy in the special unitary group SU and the symplectic group Sp. First we show that SL is generated by transvections.
In this chapter $\mathbb{F}$ is also a finite field of order q.

Definition 3.1: Let $n \in \mathbb{N}$ and $\mathbb{F}$ a field. We call

$$I_n = \mathrm{diag}(1, \ldots, 1)$$

the *identity matrix* and define $E_{i,j} \in \mathbb{F}^{n \times n}$ for $i, j \in \{1, \ldots, n\}$ as follows:

$$(E_{i,j})_{a,b} = \begin{cases} 1, & \text{if } i = a \text{ and } j = b, \\ 0, & \text{else.} \end{cases}$$

Lemma 3.2: Let $n, f \in \mathbb{F}$, $q = p^f$ a prime power. Then $\mathrm{SL}(n, q)$ is generated by transvections.

Proof. First we look at matrices of the form $I_{i,j}(\imath) = I_n + \imath \cdot E_{i,j}$ for $\imath \in \mathbb{F} - \{0\}$ and $i, j \in \{1, \ldots, n\}$ with $i \neq j$ and show that these are transvections as described in Definition 2.17. Let $e_1, \ldots, e_n$ be the standard basis of $V = \mathbb{F}_q^n$ and let $H = \langle e_1, e_2, \ldots, e_{i-1}, e_{i+1}, \ldots, e_n \rangle$. We show that $I_{i,j}(\imath)$ is a transvection with respect to the hyperplane H.

1.) Let $w \in H$. Then it follows

$$w \cdot (I_n + \imath \cdot E_{i,j}) = w \cdot I_n + w \cdot \imath \cdot E_{i,j} = w + 0 = w.$$

2.) Let $\nu \in V$. We can assume $\nu = w + \jmath \cdot e_i$ for $\jmath \in \mathbb{F} - \{0\}$ and $w \in H$. Then we have

$$\nu(I_n + \imath \cdot E_{i,j}) - \nu = \nu I_n + \nu \imath E_{i,j} - \nu = \nu - \nu + w \imath E_{i,j} + \imath \jmath e_i E_{i,j} = \imath \jmath e_i E_{i,j} \in H.$$

So $I_{i,j}(\imath) = I_n + \imath \cdot I_{i,j}$ is a transvection. Moreover $\det(I_{i,j}(\imath)) = 1$ for $i \neq j$ which implies $I_{i,j}(\imath) \in \mathrm{SL}(n, q)$.
Now we see that every matrix in $\mathrm{SL}(n, q)$ can be expressed as a product of these transvections. The proof is done by induction on n.
For $n = 1$ there is nothing to show. So assume that $n > 1$ and let $(a_{i,j})_{i,j=1,..,n} \in \mathrm{SL}(n, q)$.

1.) If $a_{1,2} = 0$ then choose some $j \in \{1, \ldots, n\}$ with $a_{1,j} \neq 0$ and add column j to column 2 by multiplying $(a_{i,j})_{n \times n}$ from right with $I_n + E_{j,2}$ to get $a_{1,2} \neq 0$.

2.) Use $I_n + (1 - a_{1,1})/a_{1,2} \cdot E_{2,1}$ from right to add column 2 $(1 - a_{1,1})/a_{1,2}$ times to column 1 to get $a_{1,1} = 1$.

3.) For $j > 1$ use transvections as row and column operations to get $a_{1,j} = 0$ and $a_{j,1} = 0$.

Now every entry of the first row and column is zero except $a_{1,1}$.
By induction the claim follows. □

Remark 3.3: In SL, let B denote the subgroup of lower-unitriangular matrices and N the subgroup of monomial matrices. Then (B, N) forms a (B, N) pair for SL.
[Taylor, page 28/29]

We can use the following algorithm to compute such a Bruhat decomposition in SL. Since we use only lower-unitriangular matrices, we can not guarantee that there is a 1 in the desired places, as in the Lemma 3.2. The algorithm is from the paper "Straight-line programs with memory and matrix Bruhat decomposition" [NPP].

Algorithm 2: UnitriangularDecompositionSL

Input: $a \in \mathrm{SL}(n, q)$.
Output: A monomial matrix $h \in \mathrm{SL}(n, q)$ and two lower unitriangular matrices $u_1, u_2 \in \mathrm{SL}(n, q)$ such that $h = u_1 \cdot a \cdot u_2$.

function UnitriangularDecompositionSL(a)
- $u_1 \coloneqq I_n$, $u_2 \coloneqq I_n$
- // we go through the columns of a from right to left
- **for** $c \in [n, \ldots, 2]$ **do**
  - // search in column c the first non-zero entry from the top
  - Search the first entry $i \in [1, \ldots, n]$ with $a_{i,c} \neq 0$
  - Set $r \coloneqq i$ and $piv \coloneqq a_{r,c}$
  - **for** $i \in [r + 1, ..., n]$ **do**
    - $\alpha \coloneqq -\frac{a_{i,c}}{a_{r,c}}$
    - // multiply transvection $E_{i,c}(\alpha)$ from left
    - $a_{i,-} \coloneqq a_{i,-} + \alpha a_{r,-}$ and $u_{1i,-} \coloneqq u_{1i,-} + \alpha u_{1r,-}$
  - **for** $j \in [c - 1, ..., 1]$ **do**
    - $\alpha \coloneqq -\frac{a_{r,j}}{a_{r,c}}$
    - // multiply transvection $E_{r,j}(\alpha)$ from right
    - $a_{-,j} \coloneqq a_{-,j} + \alpha a_{-,c}$ and $u_{2-,j} \coloneqq u_{2-,j} + \alpha u_{2-,r}$
- **return** $[a, u_1, u_2]$

The word problem of SL was subsequently solved in [NPP] as follows.
Let $a \in \mathrm{SL}$. The description of the matrix a with special generators consists of 3 steps.
First we calculate the Bruhat decomposition of the matrix a. It is important that all operations performed during the algorithm can be described with the special generators. With Algorithm 2, this step can be performed in SL. However, the description of the used matrices with the special generators has been omitted.
Now, if the monomial matrix of the Bruhat decomposition can be described with the special generators, a description of a with the special generators is found. For this purpose, the monomial matrix of the Bruhat decomposition should be represented in a product of another monomial matrix and a diagonal matrix. The second step is, therefore, to find a monomial matrix in SL, so that the product of the monomial matrices forms a diagonal matrix. So in the final step, only the description of the diagonal matrix with the special generators remains.

The same procedure with these 3 steps we now want to perform in the special unitary group and the symplectic group.

4 Bruhat Decomposition in SU

This chapter deals with the special unitary group.
First, we look at how the group can be described using a Gram-matrix to gain an understanding of the elements in this group. In the second subsection, we focus on the transvections in the special unitary group, as we aim for a similar procedure as in the special linear group. In the third subsection we describe an algorithm to obtain the Bruhat decomposition in the special unitary group and prove the correctness. Finally, we discuss the implementation to realize this procedure as an MSLP.

4.1 Mathematical Background for the Bruhat Decomposition in SU

We start with some mathematical background for this group, which we need in the rest of the chapter.

Theorem 4.1: Let V be an n-dimensional $\mathbb{F}$-vector space. Then $\mathrm{SU}(V)$ is generated by unitary transvections.

Proof. [Taylor, Theorem 10.20]. □

The matrix $J_n \in \mathbb{F}^{n\times n}$ defined below is required throughout this chapter.

Definition 4.2: Let $n \in \mathbb{N}$ and define $J_n := (J_n)_{i,j\in\{1,\dots,n\}}$ as follows:

$$(J_n)_{i,j} = \begin{cases} 1, & \text{if } i+j = n+1, \\ 0, & \text{else.} \end{cases}$$

We call J_n the *anti-diagonal matrix*.

Example 4.3: For $n = 4$ we have

$$J_4 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Remark 4.4: Observe that $J_n^2 = I_n$.

We need another definition to understand what the elements of SU look like.

Definition 4.5: Let $\mathbb{F}$ be a field admitting a field automorphism $\overline{\ }$ of order 2, $n \in \mathbb{N}$ and $a \in \mathbb{F}^{n\times n}$. We define

$$a^* := \overline{(a^{\mathrm{T}})} = (\overline{a})^{\mathrm{T}}.$$

The next result shows that we may assume that $\mathrm{U}(n,q)$ preserves a form with Gram-matrix J_n. We need this for later proofs.

Theorem 4.6: Let $n, f \in \mathbb{N}$, $q = p^f$ be a prime power and $\mathbb{F}_{q^2}$ a field admitting a field automorphism $\overline{\ }$ of order 2. Then

$$\mathrm{U}(n,q) = \{a \in \mathrm{GL}(n,q^2) \mid aJ_na^* = J_n\}.$$

Proof. We can find a basis $\{e_1, f_1, \ldots, e_n, f_n\}$ for n even and a basis $\{e_1, f_1, \ldots, e_n, f_n, w\}$ for n odd such that (e_i, f_i) is a hyperbolic pair for $i \in \{1, \ldots, n\}$ and $\Phi(w,w) = 1$. [Taylor, page 116/117]
We can order this basis into $(e_1, e_2, \ldots, f_2, f_1)$ for even n and $(e_1, e_2, \ldots, e_n, w, f_n, \ldots, f_2, f_1)$ for n odd such that the corresponding matrix has the form J_n.
Therefore, $a \in \mathrm{U}(n,q)$ if and only if $aJ_na^* = J_n$. □

Corollary 4.7: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power, $\mathbb{F}$ be a field admitting a field automorphism $\overline{\ }$ of order 2 and V an n-dimensional $\mathbb{F}_q$-vector space. Then

$$\mathrm{SU}(n,q) = \{a \in \mathrm{SL}(n,q^2) \mid aJ_na^* = J_n\}.$$

Proof. We have $\mathrm{SU}(V) = \mathrm{U}(V) \cap \mathrm{SL}(V)$. □

The next result describes what some elements of SU look like. Then we use this to construct matrices, which lie in SU and have entries in positions other than the upper left and lower right block.

Definition 4.8: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power and $a, b \in \mathrm{GL}(n,q)$. We set

$$\mathrm{Diag}(a, I_0, b) = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \quad \text{and} \quad \mathrm{Diag}(a, I_1, b) = \begin{pmatrix} a & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & b \end{pmatrix}.$$

Lemma 4.9: For a natural number n let $\ell := 1$ if n is odd and $\ell := 0$ otherwise. Further, let $m := \lfloor \frac{n}{2} \rfloor$. We set $a^{-*} := (a^*)^{-1} = (a^{-1})^*$. Then

$$\kappa\colon \mathrm{GL}(m,q^2) \hookrightarrow \mathrm{U}(n,q), a \mapsto \mathrm{Diag}(a, I_\ell, J_m a^{-*} J_m)$$

is a monomorphism.

Proof. Let $a, b \in \mathrm{GL}(m,q^2)$. Then we have

$$(ab)^{-*} = (b^* a^*)^{-1} = a^{-*} b^{-*}.$$

Therefore

$$\begin{aligned}
\kappa(ab) &= \mathrm{Diag}(ab, I_\ell, J_m (ab)^{-*} J_m) \\
&= \mathrm{Diag}(ab, I_\ell, J_m a^{-*} b^{-*} J_m) \\
&= \mathrm{Diag}(ab, I_\ell, J_m a^{-*} J_m J_m b^{-*} J_m) \\
&= \mathrm{Diag}(a, I_\ell, J_m a^{-*} J_m)\,\mathrm{Diag}(b, I_\ell, J_m b^{-*} J_m) = \kappa(a)\kappa(b).
\end{aligned}$$

Hence, κ is a group homomorphism. Moreover, we see that the homomorphism κ is injective by inspecting the top left block of $\mathrm{Diag}(a, I_\ell, J_m a^{-*} J_m)$. It remains to show that $\kappa(a) \in \mathrm{U}(n,q)$ for $a \in \mathrm{GL}(m,q^2)$. First we verify this for even dimensions, by using Corollary 4.7.

$$\begin{aligned}
\kappa(a) J_n \kappa(a)^* &= \begin{pmatrix} a & 0 \\ 0 & J_m a^{-*} J_m \end{pmatrix} \begin{pmatrix} 0 & J_m \\ J_m & 0 \end{pmatrix} \begin{pmatrix} a^* & 0 \\ 0 & J_m a^{-1} J_m \end{pmatrix} \\
&= \begin{pmatrix} 0 & a J_m \\ J_m a^{-*} & 0 \end{pmatrix} \begin{pmatrix} a^* & 0 \\ 0 & J_m a^{-1} J_m \end{pmatrix} \\
&= \begin{pmatrix} 0 & a a^{-1} J_m \\ J_m a^{-*} a^* & 0 \end{pmatrix} \\
&= J_n.
\end{aligned}$$

Using the same argument, we check this for odd dimension.

$$\begin{aligned}
\kappa(a) J_n \kappa(a)^* &= \begin{pmatrix} a & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & J_m a^{-*} J_m \end{pmatrix} \begin{pmatrix} 0 & 0 & J_m \\ 0 & 1 & 0 \\ J_m & 0 & 0 \end{pmatrix} \begin{pmatrix} a^* & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & J_m a^{-1} J_m \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & a J_m \\ 0 & 1 & 0 \\ J_m a^{-*} & 0 & 0 \end{pmatrix} \begin{pmatrix} a^* & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & J_m a^{-1} J_m \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & a a^{-1} J_m \\ 0 & 1 & 0 \\ J_m a^{-*} a^* & 0 & 0 \end{pmatrix} \\
&= J_n.
\end{aligned}$$

Hence, $\kappa(a) \in \mathrm{U}(n,q)$. □

Corollary 4.10: For a natural number n let $\ell \coloneqq 1$ if n is odd and $\ell \coloneqq 0$ otherwise. Further, let $m \coloneqq \lfloor \frac{n}{2} \rfloor$. Then

$$\kappa : \mathrm{SL}(m, q^2) \hookrightarrow \mathrm{SU}(n, q) : a \mapsto \mathrm{Diag}(a, I_\ell, J_m a^{-*} J_m)$$

is a monomorphism.

Proof. For a matrix $a \in \mathrm{SL}(m, q^2)$ we have $\det(a) = 1$, $\det(a^{-*}) = 1$, $\det(J_m) = \pm 1$ and hence

$$\begin{aligned} \det(\mathrm{Diag}(a, I_\ell, J_m a^{-*} J_m)) &= \det(a) \cdot \det(I_\ell) \cdot \det(J_m a^{-*} J_m) \\ &= \det(a) \cdot \det(I_\ell) \cdot \det(a^{-*}) \cdot \det(J_m)^2 \\ &= 1. \end{aligned}$$

So the result follows directly by the previous lemma. □

For the remainder of the chapter, we keep the notation from the previous lemma and corollary.

4.2 Transvections in SU

The focus of this chapter is on unitary transvections. We describe transvections and prove that they lie in the special unitary group.

Lemma 4.11: Let $a \in \mathrm{SL}(m, q^2)$ be a monomial matrix. Then $\kappa(a)$ is a monomial matrix.

Proof. The monomial matrices form a group under matrix multiplication. Therefore, a^{-1} is a monomial matrix and so is $(a^{-1})^* = a^{-*}$. Multiplying a monomial matrix by J_m on the left, induces a permutation of the rows and, hence, $J_m a^{-*}$ is again a monomial matrix. Multiplying a monomial matrix by J_m on the right, induces a permutation of the columns and, hence, $(J_m a^{-*})J_m$ is again a monomial matrix. Hence, $J_m a^{-*} J_m$ is again a monomial matrix. Clearly, $\mathrm{Diag}(a, I_\ell, J_m a^{-*} J_m)$ is a monomial matrix. □

We can also investigate the image of transvections under κ.

Lemma 4.12: Let $I_n + \imath \cdot E_{i,j} = I_{i,j}(\imath) \in \mathrm{SL}(m, q^2)$ for $\imath \in \mathbb{F}_{q^2}$, $i, j \in \{1, \ldots, n\}$ and $j < i$ be a lower-unitriangular matrix. Then $\kappa(I_{i,j}(\imath))$ is a lower-unitriangular matrix.

Proof. It is enough to check that $J_m \cdot (I_{i,j}(\imath))^{-*} \cdot J_m$ is a lower-unitriangular matrix.
We see that $(I_{i,j}(\imath))^{-*} = I_{j,i}(\overline{-\imath})$ is an upper-unitriangular matrix and $J_m \cdot I_{j,i}(\overline{-\imath}) \cdot J_m = I_{n+1-j,n+1-i}(\overline{-\imath})$. Now $j < i$ if and only if $n + 1 - j > n + 1 - i$. Hence, $J_m \cdot (I_{i,j}(\imath))^{-*} \cdot J_m$ is a lower-unitriangular matrix. □

Unfortunately, we can not use these matrices to eliminate entries in the lower left or upper right block of a matrix. But now we have an idea what elements of SU look like in general.

Theorem 4.13: Let $n, f \in \mathbb{N}$ and $q = p^f$ a prime power. Define

$$T_{i,j}(\imath) \coloneqq I_n + E_{i,j}(\imath) + E_{n-j+1,n-i+1}(-\bar{\imath})$$

for $i, j \in \{1, \ldots, n\}$, $j < i$, $i + j \neq n + 1$ and $\imath \in \mathbb{F}_{q^2}$. Moreover let $i \neq \frac{n+1}{2}$ if n is odd. Then $T_{i,j}(\imath) \in \mathrm{SU}(n, q)$.

Proof. We use Corollary 4.7 and check that $T_{i,j}(\imath) \cdot J_n \cdot T_{i,j}(\imath)^* = J_n$. We have

$$\begin{aligned} T_{i,j}(\imath) \cdot J_n \cdot T_{i,j}(\imath)^* &= (I_n + E_{i,j}(\imath) + E_{n-j+1,n-i+1}(-\bar{\imath})) \cdot J_n \cdot T_{i,j}(\imath)^* \\ &= (J_n + E_{i,n-j+1}(\imath) + E_{n-j+1,i}(-\bar{\imath})) \cdot T_{i,j}(\imath)^* \\ &= (J_n + E_{i,n-j+1}(\imath) + E_{n-j+1,i}(-\bar{\imath})) \cdot \overline{T_{i,j}(\imath)}^{\mathrm{T}} \\ &= (J_n + E_{i,n-j+1}(\imath) - E_{n-j+1,i}(\bar{\imath})) \cdot (I_n + E_{j,i}(\bar{\imath}) - E_{n-i+1,n-j+1}(\imath)) \\ &= J_n + E_{i,n-j+1}(\imath) - E_{n-j+1,i}(\bar{\imath}) + E_{n-j+1,i}(\bar{\imath}) - E_{i,n-j+1}(\imath) \\ &= J_n. \end{aligned}$$

Hence $T_{i,j}(\imath) \in \mathrm{SU}(n,q)$. □

Note that for the previous theorem to hold we assumed certain conditions on i and j. The following example shows that the imposed conditions are necessary.

Example 4.14: Let $f \in \mathbb{N}$ and $q = p^f$ a prime power. First we look at matrices with a field element in the lower left corner. For example we could define for $\imath \in \mathbb{F}_{q^2} - \{0\}$ the matrix

$$T_{5,1}(\imath) \coloneqq \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \imath & 0 & 0 & 0 & 1 \end{pmatrix} \in \mathrm{GL}(5, q^2).$$

This corresponds to the case of Theorem 4.13 when $i+j = n+1$ since $i+j = 5+1 = 6 = n+1$. We notice that

$$T_{5,1}(\imath) \cdot J_5 \cdot T_{5,1}(\imath)^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & \imath + \bar{\imath} \end{pmatrix}.$$

So $T_{5,1}(\imath) \in \mathrm{SU}(5,q)$ if and only if $\imath + \bar{\imath} = 0$. Notice that not every element of $\imath \in \mathbb{F}_{q^2}$ fulfils $\imath + \bar{\imath} = 0$.
We also have to look at matrices which have an entry in the central row or column. For example we could define for $\imath \in \mathbb{F}_{q^2} - \{0\}$ the matrix

$$T_{3,1}(\imath) \coloneqq \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \imath & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -\bar{\imath} & 0 & 1 \end{pmatrix} \in \mathrm{GL}(5, q^2).$$

This corresponds to the case of Theorem 4.13 when n is odd and $i = \frac{n+1}{2}$ since $n = 5$ and $i = 3 = \frac{6}{2} = \frac{n+1}{2}$. We notice that

$$T_{3,1}(\imath) \cdot J_5 \cdot T_{3,1}(\imath)^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & \imath\bar{\imath} \end{pmatrix}.$$

Therefore, $T_{3,1}(\imath) \in \mathrm{SU}(5, q)$ if and only if $\imath = 0$. Since we want to use matrices with entries in the central row and column, we can try to fix this by choosing

$$T_{3,1}(\imath) \coloneqq \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \imath & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \jmath & 0 & -\bar{\imath} & 0 & 1 \end{pmatrix} \in \mathrm{GL}(5, q^2).$$

In this case we notice that

$$T_{3,1}(\imath) \cdot J_5 \cdot T_{3,1}(\imath)^* = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & \imath\bar{\imath} + \jmath + \bar{\jmath} \end{pmatrix}.$$

So $T_{3,1}(\imath) \in \mathrm{SU}(5, q)$ if and only if $\imath$ and $\jmath$ fulfil the equation $\imath\bar{\imath} + \jmath + \bar{\jmath} = 0$. We show that this equation always has a solution in Lemma 4.15.

We want to prove the claim from the previous example.

Lemma 4.15: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power and n odd. Let $j \in \{1, \ldots, \frac{n+1}{2} - 1\}$ and $i \in \{\frac{n+1}{2} + 1, \ldots, n\}$. Then there exists $\imath_1, \jmath_1 \in \mathbb{F}_{q^2}$ such that $T_{i,\frac{n+1}{2}}(\imath_1) \in \mathrm{SU}(n, q)$ and $\imath_2, \jmath_2 \in \mathbb{F}_{q^2}$ such that $T_{\frac{n+1}{2},j}(\imath_2) \in \mathrm{SU}(n, q)$.

Proof. We prove both assertions at once by showing that there is a solution for the equation

$$\imath\bar{\imath} + \jmath + \bar{\jmath} = 0$$

for $\imath, \jmath \in \mathbb{F}_{q^2}$. Observe that

$$\overline{(\imath\bar{\imath})} = \bar{\imath}\overline{(\bar{\imath})} = \bar{\imath}\imath = \imath\bar{\imath}.$$

Therefore, $\imath\bar{\imath} \in \mathrm{Fix}(\mathbb{F}_{q^2})$ and so $-\imath\bar{\imath} \in \mathrm{Fix}(\mathbb{F}_{q^2})$. By Hilbert's Theorem 90 [NTWA, Theorem 4] there exists a $\jmath \in \mathbb{F}_{q^2}$ such that $-\imath\bar{\imath} = \jmath + \bar{\jmath}$. Hence

$$\imath\bar{\imath} + \jmath + \bar{\jmath} = 0.$$

□

We use the following notation to work with transvections in the unitary group.

Definition 4.16: Let $n, f \in \mathbb{N}$ and $q = p^f$ a prime power. For even n, $\imath \in \mathbb{F}_{q^2} - \{0\}$, $i, j \in \{1, \ldots, n\}$ and $j < i$ we set

$$T_{i,j}(\imath) = \begin{cases} I_n + E_{i,j}(\imath) - E_{n-j+1,n-i+1}(\bar{\imath}), & \text{if } i + j \neq n + 1, \\ I_n + E_{i,j}(\imath), & \text{else.} \end{cases}$$

For n odd, $\imath \in \mathbb{F}_{q^2} - \{0\}$, $i, j \in \{1, \ldots, n\}$ and $j < i$ we set

$$T_{i,j}(\imath) = \begin{cases} I_n + E_{i,j}(\imath) - E_{n-j+1,n-i+1}(\bar{\imath}), & \text{if } i + j \neq n + 1 \text{ and } i, j \neq \frac{n+1}{2}, \\ I_n + E_{i,j}(\imath), & \text{if } i + j = n + 1, \\ I_n + E_{i,j}(\imath) - E_{n-j+1,n-i+1}(\bar{\imath}) + E_{n-j+1,j}(\jmath), & \text{if } i = \frac{n+1}{2}, \\ I_n + E_{i,j}(\imath) - E_{n-j+1,n-i+1}(\bar{\imath}) + E_{i,n-i+1}(\jmath), & \text{else.} \end{cases}$$

In the last two cases we choose $\jmath \in \mathbb{F}_{q^2} - \{0\}$ such that $\imath\bar{\imath} + \jmath + \bar{\jmath} = 0$.

4.3 Algorithm for the Bruhat Decomposition in SU

We now describe an algorithm to compute the Bruhat decomposition for elements of SU. First we give a brief outline of the algorithm explaining its main ideas, then give a pseudo code for the algorithm and prove the correctness. Since the second case of Example 4.14 can only occur for odd n, we make a case distinction for the Bruhat decomposition depending on the dimension of the underlying vector space. We start with the case where n is even.

Suppose $a \in \mathrm{SU}(n, q)$ and we wish to compute the Bruhat decomposition of a.

The algorithm iterates over the following basic step:
Suppose a is an element of $\mathrm{SU}(n, q)$ with n even and $k \in \{\frac{n}{2}, \ldots, n\}$ is the least element such that for all $j \in \{k+1, \ldots, n\}$ the jth column has only one non-zero entry, say $a_{i,j}$, and $a_{i,j}$ is the only non-zero entry of ith row. This means that we can find a kth column of a such that every column right of this column of a has only one non-zero entry and this entry is also the only non-zero entry in his row. Therefore, every column right of column k has the desired monomial form.
In the basic step, we search the least $i \in \{1, \ldots, n\}$ such that $a_{i,k} \neq 0$ and multiply lower unitriangular matrices, in particular the transvections from Definition 4.16, from left and right such that column k also has exactly one non-zero element $a_{i,k}$ and $a_{i,k}$ is the only non-zero entry in row i. Moreover, we can ensure that the columns $k+1, \ldots, n$ and the corresponding rows stay in monomial form.

Having described the basic step we now describe the main algorithm. Our first aim is to transform a into a matrix in which all entries in the nth column below a pivot element $a_{i,n}$ are zero and all entries in the ith row to the left of $a_{i,n}$ are also zero. We achieve this by multiplying a by matrices $T_{i,j}(\imath)$ as in Definition 4.16.
We go through all the columns from right to left. So we start with the nth column. Now we go through this column from top to bottom and look for the first non-zero entry $a_{i,n}$. We now want to use this entry to clear all entries to the left and below. First we go through the entries $a_{k,n}$ for $k \in \{i+1, \ldots, n\}$. If $a_{k,n} \neq 0$ we multiply the matrix $T_{i,k}(-\frac{a_{k,n}}{a_{i,n}})$ from the left side such that this entry becomes 0. Hence, the only non-zero entry in the nth column is $a_{i,n}$.
Now we consider all the entries to the left of $a_{i,n}$, that is, the $a_{i,k}$ for $k \in \{1, \ldots, n-1\}$. Similarly, we multiply the matrices $T_{k,n}(-\frac{a_{i,k}}{a_{i,n}})$ from the right side to get rid of the entries, leaving only one non-zero entry in the row. This is the first use of the basic step.
We continue this procedure with the basic step until the matrix gets the desired monomial form.

It remains to understand, why lower unitriangular matrices are enough in the course of the algorithm.

Assume that a has a form where for $k \in \{\frac{n}{2}, \dots, n\}$ every column right of kth column has been modified by the basic step and, therefore, has the described monomial form. We want to use the basic step on the kth column.
We search our pivot element in the kth column. For this we go through the kth column from top to bottom and select the first non-zero element, say $a_{i,k}$. Since $a_{i,k}$ is the first non-zero entry from the top, we do not have to clear entries above $a_{i,k}$. Therefore, we only have to clear everything below $a_{i,k}$ with lower unitriangular matrices.
Assume that there is a non-zero entry right of $a_{i,k}$ say $a_{i,j}$ for $j \in \{k+1, \dots, n\}$. Then $a_{i,k}$ would be zero since we used the basic step on $a_{i,j}$. Therefore, we only have to clear everything left of $a_{i,k}$ with lower unitriangular matrices.

We give the algorithm UnitriangularDecompositionSUEven in pseudo code and Example 4.17 demonstrates a computation. Let $a \in \mathrm{SU}(n, q)$.

Algorithm 3: UnitriangularDecompositionSUEven

Input: $a \in \mathrm{SU}(n, q)$.
Output: A monomial matrix $w \in \mathrm{SU}(n, q)$ and two lower unitriangular matrices $u_1, u_2 \in \mathrm{SU}(n, q)$ such that $w = u_1 \cdot a \cdot u_2$.

function UnitriangularDecompositionSUEven(a)
- $u_1 \coloneqq I_n$, $u_2 \coloneqq I_n$
- **for** $c \in [n, \dots, \frac{n}{2}]$ **do**
  - Search first entry $i \in [1, \dots, n]$ with $a_{i,c} \neq 0$
  - Set $r \coloneqq i$ and $piv \coloneqq a_{r,c}$
  - **for** $i \in [r+1, ..., n]$ **do**
    - $\alpha \coloneqq -\frac{a_{i,c}}{a_{r,c}}$
    - **if** $r + i \neq n + 1$ **then**
      - $a_{i,-} \coloneqq a_{i,-} + \alpha a_{r,-}$; $u_{1i,-} \coloneqq u_{1i,-} + \alpha u_{1r,-}$
      - $a_{n-r+1,-} \coloneqq a_{n-r+1,-} - \overline{\alpha} a_{n-i+1,-}$; $u_{1n-r+1,-} \coloneqq u_{1n-r+1,-} - \overline{\alpha} u_{1n-i+1,-}$
    - **else**
      - $a_{i,-} \coloneqq a_{i,-} + \alpha a_{r,-}$; $u_{1i,-} \coloneqq u_{1i,-} + \alpha u_{1r,-}$
  - **for** $j \in [c-1, ..., 1]$ **do**
    - $\alpha \coloneqq -\frac{a_{r,j}}{a_{r,c}}$
    - **if** $c + j \neq n + 1$ **then**
      - $a_{-,j} \coloneqq a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} \coloneqq u_{2-,j} + \alpha u_{2-,c}$
      - $a_{-,n-c+1} \coloneqq a_{-,n-c+1} - \overline{\alpha} a_{-,n-j+1}$; $u_{2-,n-c+1} \coloneqq u_{2-,n-c+1} - \overline{\alpha} u_{2-,n-j+1}$
    - **else**
      - $a_{-,j} \coloneqq a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} \coloneqq u_{2-,j} + \alpha u_{2-,c}$
- **return** $[a, u_1, u_2]$

To better understand how and why this algorithm works, we give an example.

Example 4.17: Let $\omega \in \mathbb{F}_{49}$ be a primitive element. We demonstrate how Algorithm 3 works on the example of

$$a := \begin{pmatrix} \omega^{29} & \omega^{2} & \omega^{21} & \omega^{19} \\ \omega^{23} & 1 & 0 & \omega^{35} \\ \omega^{34} & \omega^{31} & \omega^{6} & 1 \\ 2 & \omega^{41} & \omega^{17} & \omega^{41} \end{pmatrix} \in \mathrm{SU}(4,7).$$

The algorithm first considers the 4th column. We start to pick the entry $a_{1,4} = \omega^{19}$ and clear everything below this element in column 4.

$$\begin{pmatrix} \omega^{29} & \omega^{2} & \omega^{21} & \omega^{19} \\ \omega^{23} & 1 & 0 & \omega^{35} \\ \omega^{34} & \omega^{31} & \omega^{6} & 1 \\ 2 & \omega^{41} & \omega^{17} & \omega^{41} \end{pmatrix} \xrightarrow{\cdot^{L} T_{2,1}(\omega^{5})} \begin{pmatrix} \omega^{29} & \omega^{2} & \omega^{21} & \omega^{19} \\ 4 & \omega^{41} & \omega^{13} & 0 \\ \omega^{34} & \omega^{31} & \omega^{6} & 1 \\ \omega^{31} & 5 & \omega^{7} & \omega^{23} \end{pmatrix}$$
$$\xrightarrow{\cdot^{L} T_{3,1}(\omega^{5})} \begin{pmatrix} \omega^{29} & \omega^{2} & \omega^{21} & \omega^{19} \\ 4 & \omega^{41} & \omega^{13} & 0 \\ \omega^{2} & 0 & \omega^{1} & 0 \\ \omega^{45} & \omega^{6} & \omega^{25} & \omega^{23} \end{pmatrix}.$$

Now we clear the entry $a_{4,4} = \omega^{23}$ and column 4 is finished. However, something important happens at the same time, which we show in the following.

$$\begin{pmatrix} \omega^{29} & \omega^{2} & \omega^{21} & \omega^{19} \\ 4 & \omega^{41} & \omega^{13} & 0 \\ \omega^{2} & 0 & \omega^{1} & 0 \\ \omega^{45} & \omega^{6} & \omega^{25} & \omega^{23} \end{pmatrix} \xrightarrow{\cdot^{L} T_{4,1}(\omega^{28})} \begin{pmatrix} \omega^{29} & \omega^{2} & \omega^{21} & \omega^{19} \\ 4 & \omega^{41} & \omega^{13} & 0 \\ \omega^{2} & 0 & \omega^{1} & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix}.$$

We observe that as soon as we have ensured that column 4 has only one non-zero entry, then row 4 also has only one non-zero entry. Thus, in the further course of the algorithm, we no longer need to add something to the last row. This preserves the cleared last column.
We continue the algorithm and clear row 1 with $a_{1,4} = \omega^{19}$.

$$\begin{pmatrix} \omega^{29} & \omega^{2} & \omega^{21} & \omega^{19} \\ 4 & \omega^{41} & \omega^{13} & 0 \\ \omega^{2} & 0 & \omega^{1} & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^{R} T_{4,3}(\omega^{26})} \begin{pmatrix} \omega^{7} & \omega^{2} & 0 & \omega^{19} \\ \omega^{14} & \omega^{41} & \omega^{13} & 0 \\ \omega^{2} & 0 & \omega^{1} & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix}.$$

Again we observe that as soon as we have ensured that row 1 has only one non-zero entry, then column 1 also has only one non-zero entry.

$$\begin{pmatrix} \omega^{7} & \omega^{2} & 0 & \omega^{19} \\ \omega^{14} & \omega^{41} & \omega^{13} & 0 \\ \omega^{2} & 0 & \omega^{1} & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^{R}T_{4,2}(\omega^{7})} \begin{pmatrix} \omega^{7} & 0 & 0 & \omega^{19} \\ 0 & \omega^{41} & \omega^{13} & 0 \\ 0 & 0 & \omega^{1} & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix}$$
$$\xrightarrow{\cdot^{R}T_{4,1}(\omega^{12})} \begin{pmatrix} 0 & 0 & 0 & \omega^{19} \\ 0 & \omega^{41} & \omega^{13} & 0 \\ 0 & 0 & \omega^{1} & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix}.$$

At this stage, the first and last row as well as the first and last column each contain one non-zero entry. We can look at $\begin{pmatrix} \omega^{41} & \omega^{13} \\ 0 & \omega^{1} \end{pmatrix}$ and proceed by induction. We select $a_{2,3} = \omega^{13}$ and clear column 3 and row 2.

$$\begin{pmatrix} 0 & 0 & 0 & \omega^{19} \\ 0 & \omega^{41} & \omega^{13} & 0 \\ 0 & 0 & \omega^{1} & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^{L}T_{3,2}(\omega^{12})} \begin{pmatrix} 0 & 0 & 0 & \omega^{19} \\ 0 & \omega^{41} & \omega^{13} & 0 \\ 0 & \omega^{5} & 0 & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix}$$
$$\xrightarrow{\cdot^{R}T_{3,2}(\omega^{4})} \begin{pmatrix} 0 & 0 & 0 & \omega^{19} \\ 0 & 0 & \omega^{13} & 0 \\ 0 & \omega^{5} & 0 & 0 \\ \omega^{11} & 0 & 0 & 0 \end{pmatrix} =: h.$$

Therefore

$$h = T_{3,2}(\omega^{12})T_{4,1}(\omega^{28})T_{3,1}(\omega^{5})T_{2,1}(\omega^{5})aT_{4,3}(\omega^{26})T_{4,2}(\omega^{7})T_{4,1}(\omega^{12})T_{3,2}(\omega^{4}),$$

where h is a monomial matrix. Hence, we have the Bruhat decomposition of a.

The effect we have seen in the previous example on both rows as well as columns is paramount for the correctness of the algorithm, since it ensures that the rows and columns that have already been cleared are preserved. Furthermore, we can stop the algorithm after we have used the basic step on half of the columns of the matrix, since we already know that the remaining matrix has monomial form. Therefore, we prove that the observed effect appears for all matrices in the special unitary group. We start with a cleared row.

Lemma 4.18: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power, $j \in \{1, \dots, n\}$ and $a \in \mathrm{SU}(n, q)$ such that there exists a $k \in \{1, \dots, n\}$ with $a_{k,j} \neq 0$ and $a_{k,i} = 0$ for all $i \in \{1, \dots, n\} - \{j\}$. Then

$$a_{i,n-j+1} = \begin{cases} (\overline{a_{k,j}})^{-1}, & \text{if } i = n-k+1, \\ 0, & \text{else.} \end{cases}$$

Proof. We set $t := n-j+1$. Since $a \in \mathrm{SU}(n, q)$ we know by Corollary 4.7 that $a \cdot J_n \cdot a^* = J_n$. Hence

$$\begin{aligned}
&a \cdot J_n \cdot a^* \\
&= \begin{pmatrix}
* & \dots & * & a_{1,t} & * & \dots & * & * & * & \dots & * \\
* & \dots & * & a_{2,t} & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & a_{i-1,t} & * & \dots & * & * & * & \dots & * \\
0 & \dots & 0 & 0 & 0 & \dots & 0 & a_{k,j} & 0 & \dots & 0 \\
* & \dots & * & a_{i+1,t} & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & a_{n,t} & * & \dots & * & * & * & \dots & *
\end{pmatrix} \cdot J_n \cdot a^* \\
&= \begin{pmatrix}
* & \dots & * & * & * & \dots & * & a_{1,t} & * & \dots & * \\
* & \dots & * & * & * & \dots & * & a_{2,t} & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & a_{i-1,t} & * & \dots & * \\
0 & \dots & 0 & a_{k,j} & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\
* & \dots & * & * & * & \dots & * & a_{i+1,t} & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & a_{n,t} & * & \dots & *
\end{pmatrix} \cdot \begin{pmatrix}
* & \dots & * & 0 & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & 0 & * & \dots & * \\
\overline{a_{1,t}} & \dots & \overline{a_{k-1,t}} & 0 & \overline{a_{k+1,t}} & \dots & \overline{a_{n,t}} \\
* & \dots & * & 0 & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & 0 & * & \dots & * \\
* & \dots & * & \overline{a_{k,j}} & * & \dots & * \\
* & \dots & * & 0 & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & 0 & * & \dots & *
\end{pmatrix} \\
&= \begin{pmatrix}
0 & 0 & 0 & \dots & 0 & 1 \\
0 & 0 & 0 & \dots & 1 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & 0 & 0 \\
a_{k,j}\overline{a_{1,t}} & a_{k,j}\overline{a_{2,t}} & a_{k,j}\overline{a_{3,t}} & \dots & a_{k,j}\overline{a_{n-1,t}} & a_{k,j}\overline{a_{n,t}} \\
0 & 0 & 0 & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 1 & 0 & \dots & 0 & 0 \\
1 & 0 & 0 & \dots & 0 & 0
\end{pmatrix} \\
&= J_n.
\end{aligned}$$

Therefore, the claim follows. □

We show the same result for a column with one non-zero entry.

Lemma 4.19: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power, $j \in \{1, \dots, n\}$ and $a \in \mathrm{SU}(n, q)$ such that there exists a $k \in \{1, \dots, n\}$ with $a_{k,j} \neq 0$ and $a_{i,j} = 0$ for all $i \in \{1, \dots, n\} - \{k\}$. Then

$$a_{n-k+1,i} = \begin{cases} (\overline{a_{k,j}})^{-1}, & \text{if } i = n - j + 1, \\ 0, & \text{else.} \end{cases}$$

Proof. We know by Corollary 4.7 that $aJ_n a^* = J_n$. Since $J_n^{-1} = J_n$ we see that

$$(aJ_n a^*)^{-1} = (J_n)^{-1} \iff a^{-*} J_n a^{-1} = J_n.$$

Hence, $a^{-*} \in \mathrm{SU}(n, q)$ and so $a^* \in \mathrm{SU}(n, q)$. Therefore, the claim follows by Lemma 4.18. □

Now we deal with the first problem of Example 4.14. This corresponds to a transvection $T_{i,j}(\imath)$ where $i + j = n + 1$. The next theorem shows that the transvection lies in SU when we have cleared every entry between the pivot element and the entry we want to remove.

Lemma 4.20: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power and $a \in \mathrm{SU}(n, q)$ such that there exists a $j \in \{\lceil \frac{n}{2} \rceil, \dots, n\}$, $i \in \{1, \dots, n\}$ with $a_{i,j} \neq 0$, $a_{i,n-j+1} \neq 0$ and $a_{i,k} = 0$ for all $k \in \{n - j + 2, \dots, n\} - \{j\}$. Then $T_{j,n-j+1}(-\frac{a_{i,n-j+1}}{a_{i,j}}) \in \mathrm{SU}(n, q)$.

Proof. We know by Corollary 4.7 that $aJ_n a^* = J_n$. We set $\ell \coloneqq n - j + 1$. Hence

$$a \cdot J_n \cdot a^* = \begin{pmatrix} * & \dots & * & * & * & \dots & * & * & * & \dots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \dots & * & * & * & \dots & * & * & * & \dots & * \\ * & \dots & * & a_{i,\ell} & 0 & \dots & 0 & a_{i,j} & 0 & \dots & 0 \\ * & \dots & * & * & * & \dots & * & * & * & \dots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \dots & * & * & * & \dots & * & * & * & \dots & * \end{pmatrix} \cdot J_n \cdot a^*$$

$$
= \begin{pmatrix}
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
0 & \dots & 0 & a_{i,j} & 0 & \dots & 0 & a_{i,\ell} & * & \dots & * \\
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & * & * & \dots & *
\end{pmatrix} \cdot
\begin{pmatrix}
* & * & \dots & * & * & * & \dots & * & * \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
* & * & \dots & * & * & * & \dots & * & * \\
* & * & \dots & * & \overline{a_{i,\ell}} & * & \dots & * & * \\
* & * & \dots & * & 0 & * & \dots & * & * \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
* & * & \dots & * & 0 & * & \dots & * & * \\
* & * & \dots & * & \overline{a_{i,j}} & * & \dots & * & * \\
* & * & \dots & * & 0 & * & \dots & * & * \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
* & * & \dots & * & 0 & * & \dots & * & *
\end{pmatrix}
$$

$$
= \begin{pmatrix}
0 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 1 \\
0 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 1 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & & \vdots & & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 0 \\
0 & 0 & 0 & \dots & 0 & a_{i,j}\overline{a_{i,\ell}} + a_{i,\ell}\overline{a_{i,j}} & & & 0 & \dots & 0 & 0 \\
0 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & & \vdots & & \vdots & \ddots & \vdots & \vdots \\
0 & 1 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 0 \\
1 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 0
\end{pmatrix}
$$

$$= J_n.$$

Therefore, $a_{i,j}\overline{a_{i,\ell}} + a_{i,\ell}\overline{a_{i,j}} = (a \cdot J_n \cdot a^*)_{i,i} = (J_n)_{i,i} = 0$ and we have $a_{i,\ell} + a_{i,j} \cdot \frac{(\overline{a_{i,\ell}})}{(\overline{a_{i,j}})} = a_{i,\ell} + a_{i,j} \cdot (\overline{\frac{a_{i,\ell}}{a_{i,j}}}) = 0$. Moreover we know that $a_{i,\ell} + a_{i,j} \cdot (-\frac{a_{i,n-j+1}}{a_{i,j}}) = 0$. Since $\mathbb{F}_{q^2}$ is a finite field there can be only one solution, namely $(-\frac{a_{i,n-j+1}}{a_{i,j}}) = \overline{(\frac{a_{i,\ell}}{a_{i,j}})}$.
Again we use Corollary 4.7 and show that

$$T_{j,\ell}(-\tfrac{a_{i,\ell}}{a_{i,j}}) J_n T_{j,\ell}(-\tfrac{a_{i,\ell}}{a_{i,j}})^* = T_{j,\ell}(\overline{(\tfrac{a_{i,\ell}}{a_{i,j}})}) J_n T_{j,\ell}(\overline{(\tfrac{a_{i,\ell}}{a_{i,j}})})^* = J_n.$$

We have

$$\begin{aligned}
T_{j,\ell}(\overline{(\frac{a_{i,\ell}}{a_{i,j}})}) J_n T_{j,\ell}(\overline{(\frac{a_{i,\ell}}{a_{i,j}})})^* &= (I_n + \overline{(\frac{a_{i,\ell}}{a_{i,j}})} E_{j,\ell}) J_n (I_n + (\frac{a_{i,\ell}}{a_{i,j}}) E_{\ell,j}) \\
&= (J_n + \overline{(\frac{a_{i,\ell}}{a_{i,j}})} E_{j,j}) (I_n + (\frac{a_{i,\ell}}{a_{i,j}}) E_{\ell,j}) \\
&= J_n + (\overline{(\frac{a_{i,\ell}}{a_{i,j}})} + (\frac{a_{i,\ell}}{a_{i,j}})) E_{j,j} \\
&= J_n + ((-\frac{a_{i,n-j+1}}{a_{i,j}}) + (\frac{a_{i,\ell}}{a_{i,j}})) E_{j,j} \\
&= J_n + 0 = J_n.
\end{aligned}$$

Hence, the claim follows. □

Again we prove the same result for columns.

Lemma 4.21: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power and $a \in \mathrm{SU}(n,q)$ such that there exists a $j \in \{1,\dots,n\}$, $i \in \{1,\dots,\lfloor \frac{n}{2} \rfloor\}$ with $a_{i,j} \neq 0$, $a_{n-i+1,j} \neq 0$ and $a_{k,j} = 0$ for all $k \in \{1,\dots,n-i\} - \{i\}$. Then $T_{n-i+1,i}(-\frac{a_{n-i+1,j}}{a_{i,j}}) \in \mathrm{SU}(n,q)$.

Proof. Analogous to Lemma 4.19 we know that $a^* \in \mathrm{SU}(n,q)$. Let $b \coloneqq a^*$ and $\ell \coloneqq n-i+1$. We know by Corollary 4.7 that $bJ_nb^* = J_n$. Hence

$$
\begin{aligned}
& b \cdot J_n \cdot b^* \\
&= \begin{pmatrix}
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
0 & \dots & 0 & \overline{a_{i,j}} & 0 & \dots & 0 & \overline{a_{\ell,j}} & * & \dots & * \\
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & * & * & \dots & *
\end{pmatrix} \cdot J_n \cdot b^* \\
&= \begin{pmatrix}
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
* & \dots & * & \overline{a_{\ell,j}} & 0 & \dots & 0 & \overline{a_{i,j}} & 0 & \dots & 0 \\
* & \dots & * & * & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & * & * & \dots & *
\end{pmatrix} \cdot \begin{pmatrix}
* & * & \dots & * & 0 & * & \dots & * & * \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
* & * & \dots & * & 0 & * & \dots & * & * \\
* & * & \dots & * & a_{i,j} & * & \dots & * & * \\
* & * & \dots & * & 0 & * & \dots & * & * \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
* & * & \dots & * & 0 & * & \dots & * & * \\
* & * & \dots & * & a_{\ell,j} & * & \dots & * & * \\
* & * & \dots & * & * & * & \dots & * & * \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
* & * & \dots & * & * & * & \dots & * & *
\end{pmatrix} \\
&= \begin{pmatrix}
0 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 1 \\
0 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 1 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & & \vdots & & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 0 \\
0 & 0 & 0 & \dots & 0 & a_{i,j}\overline{a_{\ell,j}} + a_{\ell,j}\overline{a_{i,j}} & & & 0 & \dots & 0 & 0 \\
0 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & & \vdots & & \vdots & \ddots & \vdots & \vdots \\
0 & 1 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 0 \\
1 & 0 & 0 & \dots & 0 & & 0 & & 0 & \dots & 0 & 0
\end{pmatrix} \\
&= J_n.
\end{aligned}
$$

Therefore, $a_{i,j}\overline{a_{\ell,j}} + a_{\ell,j}\overline{a_{i,j}} = (b \cdot J_n \cdot b^*)_{j,j} = (J_n)_{j,j} = 0$ and we have $a_{\ell,j} + a_{i,j} \cdot \frac{(\overline{a_{\ell,j}})}{(\overline{a_{i,j}})} = a_{\ell,j} + a_{i,j} \cdot (\overline{\frac{a_{\ell,j}}{a_{i,j}}}) = 0$. Moreover we know that $a_{\ell,j} + a_{i,j} \cdot (-\frac{a_{\ell,j}}{a_{i,j}}) = 0$. Since

$\mathbb{F}_{q^2}$ is a finite field there can be only one solution, namely $(-\frac{a_{\ell,j}}{a_{i,j}}) = \overline{(\frac{a_{\ell,j}}{a_{i,j}})}$.
Again we use Corollary 4.7 and show that

$$T_{\ell,i}(-\tfrac{a_{\ell,j}}{a_{i,j}})J_nT_{\ell,i}(-\tfrac{a_{\ell,j}}{a_{i,j}})^* = T_{\ell,i}(\overline{(\tfrac{a_{\ell,j}}{a_{i,j}})})J_nT_{\ell,i}(\overline{(\tfrac{a_{\ell,j}}{a_{i,j}})})^* = J_n.$$

We have

$$\begin{aligned}
T_{\ell,i}(\overline{(\frac{a_{\ell,j}}{a_{i,j}})})J_nT_{\ell,i}(\overline{(\frac{a_{\ell,j}}{a_{i,j}})})^* &= (I_n + \overline{(\frac{a_{\ell,j}}{a_{i,j}})}E_{\ell,i})J_n(I_n + (\frac{a_{\ell,j}}{a_{i,j}})E_{i,\ell}) \\
&= (J_n + \overline{(\frac{a_{\ell,j}}{a_{i,j}})}E_{\ell,\ell})(I_n + (\frac{a_{\ell,j}}{a_{i,j}})E_{i,\ell}) \\
&= J_n + (\overline{(\frac{a_{\ell,j}}{a_{i,j}})} + (\frac{a_{\ell,j}}{a_{i,j}}))E_{\ell,\ell} \\
&= J_n + ((-\frac{a_{\ell,j}}{a_{i,j}}) + (\frac{a_{\ell,j}}{a_{i,j}}))E_{\ell,\ell} \\
&= J_n + 0 = J_n.
\end{aligned}$$

Hence, the claim follows. □

Now we have everything to proof the correctness of the algorithm UnitriangularDecompositionSUEven.

Theorem 4.22: The algorithm UnitriangularDecompositionSUEven (Algorithm 3) works correctly and terminates.

Proof. Let $a \in \mathrm{SU}(n, q)$.
We start with column n and pick the least i such that the entry $a_{i,n} \neq 0$. Such an entry must exist, since $\det(a) = 1$. By Theorem 4.13 we can use the matrices $T_{i,j}$ to clear all entries $a_{k,n}$ for $k \in \{i+1, \ldots, n\}$ and $k + i \neq n + 1$. By Lemma 4.21 we can also eliminate the entry $a_{k,n}$ with $k + i = n + 1$. Therefore, everything above and below the entry $a_{i,n}$ in column n is zero and we used only lower-unitriangular matrices. Again we use Theorem 4.13 and clear every entry $a_{i,k}$ left from $a_{i,n}$ for $k \in \{1, \ldots, n-1\}$ and $k + i \neq n + 1$. By Lemma 4.20 we can also eliminate the entry $a_{i,k}$ with $k + n = n + 1$. Therefore, the row i also contains only one non-zero entry.
By Lemma 4.18 and Lemma 4.19 it follows that row $n - i + 1$ as well as column $n - n + 1$ contain only one non-zero entry. Hence, we can ensure that over the course of the algorithm one non-zero entry remains in the cleared row and column.
Therefore, the claim follows iteratively. □

Computing the Bruhat decomposition for odd n is a little bit more difficult. We also start to describe the algorithm and then show the missing claims.

The main idea is identical to the case where n is even. Here, however, we have to take care of the center column and row, as the Example 4.14 has shown. In order to avoid problems in the further algorithm, this is always the first entry which is set to 0 by a matrix multiplication from left or right, if this is not the selected element.
The further course is then analogous to the case of the even dimension.

We give the algorithm UnitriangularDecompositionSUOdd in pseudo code.
Let $a \in \mathrm{SU}(n, q)$.

Algorithm 4: UnitriangularDecompositionSUOdd

Input: $a \in \mathrm{SU}(n, q)$.
Output: A monomial matrix $w \in \mathrm{SU}(n, q)$ and two lower unitriangular matrices $u_1, u_2 \in \mathrm{SU}(n, q)$ such that $w = u_1 \cdot a \cdot u_2$.

function UnitriangularDecompositionSUOdd(a)
  $u_1 \coloneqq I_n$, $u_2 \coloneqq I_n$
  for $c \in [n, \ldots, \frac{n+1}{2}]$ **do**
    Search first entry $i \in [1, \ldots, n]$ with $a_{i,c} \neq 0$
    Set $r \coloneqq i$ and $piv \coloneqq a_{r,c}$
    if $a_{\frac{n+1}{2},c}$ *is not zero* **then**
      Set $i = \frac{n+1}{2}$ and $\alpha \coloneqq -\frac{a_{i,c}}{a_{r,c}}$
      if $r + i \neq n + 1$ **then**
        Find x with $\alpha\overline{\alpha} + x + \overline{x} = 0$
        $a_{n-r+1,-} \coloneqq a_{n-r+1,-} + x a_{r,-}$; $u_{1n-r+1,-} \coloneqq u_{1n-r+1,-} + x u_{1r,-}$
        $a_{n-r+1,-} \coloneqq a_{n-r+1,-} - \overline{\alpha} a_{n-i+1,-}$; $u_{1n-r+1,-} \coloneqq u_{1n-r+1,-} - \overline{\alpha} u_{1n-i+1,-}$
        $a_{i,-} \coloneqq a_{i,-} + \alpha a_{r,-}$; $u_{1i,-} \coloneqq u_{1i,-} + \alpha u_{1r,-}$
      else
        $a_{i,-} \coloneqq a_{i,-} + \alpha a_{r,-}$; $u_{1i,-} \coloneqq u_{1i,-} + \alpha u_{1r,-}$
    for $i \in [r + 1, ..., n]$ **do**
      $\alpha \coloneqq -\frac{a_{i,c}}{a_{r,c}}$
      if $r + i \neq n + 1$ **then**
        $a_{i,-} \coloneqq a_{i,-} + \alpha a_{r,-}$; $u_{1i,-} \coloneqq u_{1i,-} + \alpha u_{1r,-}$
        $a_{n-r+1,-} \coloneqq a_{n-r+1,-} - \overline{\alpha} a_{n-i+1,-}$; $u_{1n-r+1,-} \coloneqq u_{1n-r+1,-} - \overline{\alpha} u_{1n-i+1,-}$
      else
        $a_{i,-} \coloneqq a_{i,-} + \alpha a_{r,-}$; $u_{1i,-} \coloneqq u_{1i,-} + \alpha u_{1r,-}$

if $a_{r,\frac{n+1}{2}}$ *is not zero* **then**
  Set $j = \frac{n+1}{2}$ and $\alpha := -\frac{a_{r,j}}{a_{r,c}}$;
  if $r + j \neq n + 1$ **then**
    Find x with $\alpha\overline{\alpha} + x + \overline{x} = 0$;
    $a_{-,n-c+1} := a_{-,n-c+1} + xa_{-,c}$; $u_{2-,n-c+1} := u_{2-,n-c+1} + xu_{2-,c}$;
    $a_{-,n-c+1} := a_{-,n-c+1} - \overline{\alpha}a_{-,n-j+1}$; $u_{2-,n-c+1} := u_{2-,n-c+1} - \overline{\alpha}u_{2-,n-j+1}$;
    $a_{-,j} := a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} := u_{2-,j} + \alpha u_{2-,c}$;
  else
    $a_{-,j} := a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} := u_{2-,j} + \alpha u_{2-,c}$;
for $j \in [c-1, ..., 1]$ **do**
  $\alpha := -\frac{a_{r,j}}{a_{r,c}}$;
  if $c + j \neq n + 1$ **then**
    $a_{-,j} := a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} := u_{2-,j} + \alpha u_{2-,c}$;
    $a_{-,n-c+1} := a_{-,n-c+1} - \overline{\alpha}a_{-,n-j+1}$; $u_{2-,n-c+1} := u_{2-,n-c+1} - \overline{\alpha}u_{2-,n-j+1}$;
  else
    $a_{-,j} := a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} := u_{2-,j} + \alpha u_{2-,c}$;
return $[a, u_1, u_2]$

To see the difference to the even case, we give an example.

Example 4.23: Let $\omega \in \mathbb{F}_{16}$ be a primitive element. We demonstrate how Algorithm 4 works on the example of

$$a := \begin{pmatrix} \omega^4 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \\ \omega^5 & \omega^1 & \omega^4 & \omega^8 & \omega^{12} \\ \omega^9 & \omega^4 & \omega^2 & \omega^{10} & \omega^{11} \\ \omega^7 & \omega^{13} & \omega^1 & \omega^6 & \omega^{12} \\ \omega^6 & \omega^8 & \omega^5 & \omega^{13} & \omega^3 \end{pmatrix} \in \mathrm{SU}(5,4).$$

First we pick the entry $a_{1,5} = \omega^{14}$ and start to clear the entry in the middle.

$$\begin{pmatrix} \omega^4 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \\ \omega^5 & \omega^1 & \omega^4 & \omega^8 & \omega^{12} \\ \omega^9 & \omega^4 & \omega^2 & \omega^{10} & \omega^{11} \\ \omega^7 & \omega^{13} & \omega^1 & \omega^6 & \omega^{12} \\ \omega^6 & \omega^8 & \omega^5 & \omega^{13} & \omega^3 \end{pmatrix} \xrightarrow{\cdot {}^L T_{3,1}(\omega^{12})} \begin{pmatrix} \omega^4 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \\ \omega^5 & \omega^1 & \omega^4 & \omega^8 & \omega^{12} \\ \omega^3 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^7 & \omega^{13} & \omega^1 & \omega^6 & \omega^{12} \\ \omega^6 & \omega^3 & \omega^{14} & \omega^{13} & \omega^9 \end{pmatrix}$$

We continue to clear everything below this element in column 5.

$$\begin{pmatrix} \omega^4 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \\ \omega^5 & \omega^1 & \omega^4 & \omega^8 & \omega^{12} \\ \omega^3 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^7 & \omega^{13} & \omega^1 & \omega^6 & \omega^{12} \\ \omega^6 & \omega^3 & \omega^{14} & \omega^{13} & \omega^9 \end{pmatrix} \xrightarrow{\cdot^L T_{2,1}(\omega^{13})} \begin{pmatrix} \omega^4 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \\ \omega^1 & \omega^8 & 0 & \omega^{13} & 0 \\ \omega^3 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^7 & \omega^{13} & \omega^1 & \omega^6 & \omega^{12} \\ \omega^8 & \omega^{11} & \omega^6 & 0 & \omega^{14} \end{pmatrix}$$

$$\xrightarrow{\cdot^L T_{4,1}(\omega^{13})} \begin{pmatrix} \omega^4 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \\ \omega^1 & \omega^8 & 0 & \omega^{13} & 0 \\ \omega^3 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^{12} & \omega^9 & 1 & \omega^2 & 0 \\ 0 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \end{pmatrix} \xrightarrow{\cdot^L T_{5,1}(\omega^0)} \begin{pmatrix} \omega^4 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \\ \omega^1 & \omega^8 & 0 & \omega^{13} & 0 \\ \omega^3 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^{12} & \omega^9 & 1 & \omega^2 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix}$$

We observe the same effect as for even n. We continue the algorithm and clear row 1 with $a_{1,5} = \omega^{14}$. Again we start to clear the entry in the middle.

$$\begin{pmatrix} \omega^4 & \omega^{12} & \omega^6 & \omega^5 & \omega^{14} \\ \omega^1 & \omega^8 & 0 & \omega^{13} & 0 \\ \omega^3 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^{12} & \omega^9 & 1 & \omega^2 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^R T_{5,3}(\omega^7)} \begin{pmatrix} \omega^8 & \omega^{12} & 0 & \omega^5 & \omega^{14} \\ \omega^1 & \omega^8 & 0 & \omega^{13} & 0 \\ \omega^7 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^1 & \omega^9 & 1 & \omega^2 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix}$$

And now we can clear the rest of the first row.

$$\begin{pmatrix} \omega^8 & \omega^{12} & 0 & \omega^5 & \omega^{14} \\ \omega^1 & \omega^8 & 0 & \omega^{13} & 0 \\ \omega^7 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^1 & \omega^9 & 1 & \omega^2 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^R T_{5,4}(\omega^6)} \begin{pmatrix} \omega^{14} & \omega^{12} & 0 & 0 & \omega^{14} \\ \omega^5 & \omega^8 & 0 & \omega^{13} & 0 \\ \omega^{11} & \omega^{14} & \omega^6 & \omega^4 & 0 \\ \omega^9 & \omega^9 & 1 & \omega^2 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\xrightarrow{\cdot^R T_{5,2}(\omega^{13})} \begin{pmatrix} \omega^{14} & 0 & 0 & 0 & \omega^{14} \\ 0 & \omega^8 & 0 & \omega^{13} & 0 \\ 0 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ 0 & \omega^9 & 1 & \omega^2 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^R T_{5,1}(\omega^0)} \begin{pmatrix} 0 & 0 & 0 & 0 & \omega^{14} \\ 0 & \omega^8 & 0 & \omega^{13} & 0 \\ 0 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ 0 & \omega^9 & 1 & \omega^2 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Now we can continue with the next column. We select $a_{2,4} = \omega^{13}$ and clear the entry in the middle of column 4.

$$\begin{pmatrix} 0 & 0 & 0 & 0 & \omega^{14} \\ 0 & \omega^8 & 0 & \omega^{13} & 0 \\ 0 & \omega^{14} & \omega^6 & \omega^4 & 0 \\ 0 & \omega^9 & 1 & \omega^2 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^L T_{3,2}(\omega^6)} \begin{pmatrix} 0 & 0 & 0 & 0 & \omega^{14} \\ 0 & \omega^8 & 0 & \omega^{13} & 0 \\ 0 & 0 & \omega^6 & 0 & 0 \\ 0 & \omega^{13} & 0 & \omega^8 & 0 \\ \omega^4 & 0 & 0 & 0 & 0 \end{pmatrix}$$

We can continue to clear the rest of column 4.

$$\begin{pmatrix} 0 & 0 & 0 & 0 & \omega^{14} \\ 0 & \omega^{8} & 0 & \omega^{13} & 0 \\ 0 & 0 & \omega^{6} & 0 & 0 \\ 0 & \omega^{13} & 0 & \omega^{8} & 0 \\ \omega^{4} & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^{L}T_{4,2}(\omega^2)} \begin{pmatrix} 0 & 0 & 0 & 0 & \omega^{14} \\ 0 & \omega^{8} & 0 & \omega^{13} & 0 \\ 0 & 0 & \omega^{6} & 0 & 0 \\ 0 & \omega^{8} & 0 & 0 & 0 \\ \omega^{4} & 0 & 0 & 0 & 0 \end{pmatrix}$$

Since the middle of row 2 is already 0, we can continue with the rest.

$$\begin{pmatrix} 0 & 0 & 0 & 0 & \omega^{14} \\ 0 & \omega^{8} & 0 & \omega^{13} & 0 \\ 0 & 0 & \omega^{6} & 0 & 0 \\ 0 & \omega^{8} & 0 & 0 & 0 \\ \omega^{4} & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^{R}T_{4,2}(\omega^2)} \begin{pmatrix} 0 & 0 & 0 & 0 & \omega^{14} \\ 0 & 0 & 0 & \omega^{13} & 0 \\ 0 & 0 & \omega^{6} & 0 & 0 \\ 0 & \omega^{8} & 0 & 0 & 0 \\ \omega^{4} & 0 & 0 & 0 & 0 \end{pmatrix}$$

Analogous to the case where n is even, we have the Bruhat decomposition of a.

We also want to show the correctness of the algorithm.

Theorem 4.24: The algorithm UnitriangularDecompositionSUOdd (Algorithm 4) works correctly and terminates.

Proof. Let $a \in \mathrm{SU}(n, q)$.
We start with column n and pick the least i such that the entry $a_{i,n} \neq 0$. Such an entry must exist, since $\det(a) = 1$. Now we clear the entry in the middle of the column by Lemma 4.15. By Theorem 4.13 we can use the matrices $T_{i,j}$ to clear all entries $a_{k,n}$ for $k \in \{i+1, \ldots, n\} - \{\frac{n+1}{2}\}$ and $k + i \neq n + 1$. By Lemma 4.21 we can also eliminate the entry $a_{k,n}$ with $k + i = n + 1$. Therefore, everything above and below the entry $a_{i,n}$ is zero and we used only lower-unitriangular matrices. We can clear the entry in the middle of the row i by Lemma 4.15. Again we use Theorem 4.13 and clear every entry $a_{i,k}$ left from $a_{i,n}$ for $k \in \{1, \ldots, n-1\} - \{\frac{n+1}{2}\}$ and $k + i \neq n + 1$. By Lemma 4.20 we can also eliminate the entry $a_{i,k}$ with $k + n = n + 1$. Therefore, the row i also contains only one non-zero entry.
By Lemma 4.18 and Lemma 4.19 it follows that row $n - i + 1$ as well as column $n - n + 1$ contain only one non-zero entry. Hence, we can ensure that in the course of the algorithm one non-zero entry remains in the cleared row and column.
Therefore, the claim follows iteratively. □

4.4 Implementation of the Bruhat Decomposition in SU

In this subsection we discuss the implementation of the algorithms for the Bruhat decomposition in SU (Algorithm 3 and Algorithm 4) and how it can be described with an MSLP. In addition, we discuss how a monomial matrix can be described as the product of particular generators.

First we describe the Leedham-Green-O'Brien standard generators [GB] of $\mathrm{SU}(n,q)$, because we want to use them to solve the word problem described in chapter 2.

Definition 4.25: Let q be a prime power, say $q = p^f$, $n \in \mathbb{N}$, $\gamma \in \mathbb{F}_{q^2}$ a specified primitive element and $\alpha = \gamma^{(q+1)/2}$. The *Leedham-Green-O'Brien standard generators* [GB] of $\mathrm{SU}(n,q)$ are the following:

$$
\begin{aligned}
s_0 &= \begin{pmatrix} 0 & \alpha \\ \alpha^{-q} & 0 \end{pmatrix} \text{ and } \widetilde{s} = \begin{pmatrix} s_0 & 0 \\ 0 & I_{n-2} \end{pmatrix}, \\
t_0 &= \begin{pmatrix} 1 & \alpha \\ 0 & 1 \end{pmatrix} \text{ and } \widetilde{t} = \begin{pmatrix} t_0 & 0 \\ 0 & I_{n-2} \end{pmatrix}, \\
\widetilde{\delta} &= \mathrm{diag}(\gamma^{q+1}, \gamma^{-(q+1)}, 1, \ldots, 1), \\
\widetilde{v} &= \begin{cases} \begin{pmatrix} 0 & I_{n-2} \\ I_2 & 0 \end{pmatrix}, & n \text{ even}, \\ \begin{pmatrix} 0 & I_{n-2} & 0 \\ I_2 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, & n \text{ odd}, \end{cases} \\
u_0 &= \begin{pmatrix} 0 & I_2 \\ I_2 & 0 \end{pmatrix} \text{ and } \widetilde{u} = \begin{pmatrix} u_0 & 0 \\ 0 & I_{n-4} \end{pmatrix}, \\
\widetilde{x} &= \begin{cases} \begin{pmatrix} x_0 & 0 \\ 0 & I_{n-4} \end{pmatrix}, & n \text{ even with } x_0 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{pmatrix}, \\ \begin{pmatrix} x_0 & 0 \\ 0 & I_{n-3} \end{pmatrix}, & n \text{ odd with } x_0 = \begin{pmatrix} 1 & 1 & -2^{-1} \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{pmatrix}, \end{cases}
\end{aligned}
$$

$$\widetilde{y} = \begin{cases} \begin{pmatrix} y_0 & 0 \\ 0 & I_{n-4} \end{pmatrix}, & n \text{ even with } y_0 = \begin{pmatrix} \gamma & 0 & 0 & 0 \\ 0 & \gamma^{-q} & 0 & 0 \\ 0 & 0 & \gamma^{-1} & 0 \\ 0 & 0 & 0 & \gamma^{q} \end{pmatrix}, \\ \begin{pmatrix} y_0 & 0 \\ 0 & I_{n-3} \end{pmatrix}, & n \text{ odd with } y_0 = \begin{pmatrix} \gamma & 0 & 0 \\ 0 & \gamma^{q-1} & 0 \\ 0 & 0 & \gamma^{-q} \end{pmatrix}. \end{cases}$$

Leedham-Green and O'Brien use a different basis in their paper. Therefore, we have to perform a basis transformation. We start with an even dimension. Let $\{e_1, \ldots, e_n\}$ be the standard basis of $\mathbb{F}_{q^2}^n$. We use the following matrix

$$T_n^{\mathrm{SUE}} = \begin{pmatrix} e_1 & e_n & e_2 & e_{n-1} & \ldots & e_{\frac{n}{2}} & e_{\frac{n}{2}+1} \end{pmatrix} \in \mathrm{GL}(n, q^2).$$

Example 4.26: For $n = 6$ we have

$$T_6^{\mathrm{SUE}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

When we use the matrix T_n^{SUE} to change the basis, we obtain the generating set.

Definition 4.27: Let q be a prime power, say $q = p^f$, n a positive even integer, $\gamma \in \mathbb{F}_{q^2}$ a specified primitive element and $\alpha = \gamma^{(q+1)/2}$. The *Leedham-Green-O'Brien standard generators* [GB] of $\mathrm{SU}(n, q)$ transformed with T_n^{SUE} are the following:

A transposition : $$s = \begin{pmatrix} 0 & 0 & \alpha \\ 0 & I_{n-2} & 0 \\ \alpha^{-q} & 0 & 0 \end{pmatrix},$$

A transvection : $$t = I_n + \alpha \cdot E_{1,n},$$

A diagonal matrix with 2 non-identity entries : $$\delta = \mathrm{diag}(\gamma^{q+1}, 1, \ldots, 1, \gamma^{-(q+1)}),$$

Product of two cycles of length $\frac{n}{2}$: $v = \begin{pmatrix} 0 & I_{\frac{n}{2}-1} & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & I_{\frac{n}{2}-1} & 0 \end{pmatrix}$,

Two 2 cycles : $u = \begin{pmatrix} J_2 & 0 & 0 \\ 0 & I_{n-4} & 0 \\ 0 & 0 & J_2 \end{pmatrix}$,

A upper unitriangular matrix : $x = I_n + E_{1,2} + (-1) \cdot E_{n-1,n}$,

A diagonal matrix with 4 non-identity entries : $y = \mathrm{diag}(\gamma, \gamma^{-1}, 1, \ldots, 1, \gamma^q, \gamma^{-q})$.

Example 4.28: We get the following generating set for $n = 6$ and $q = 7$:

$$s = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \gamma^4 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \gamma^{20} & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad t = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \gamma^4 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad \delta = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \end{pmatrix},$$

$$v = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad u = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad x = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 6 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

$$y = \begin{pmatrix} \gamma^1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \gamma^{47} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \gamma^7 & 0 \\ 0 & 0 & 0 & 0 & 0 & \gamma^{41} \end{pmatrix}.$$

Now we consider the case that n is odd. Let $\{e_1, \ldots, e_n\}$ be the standard basis of $\mathbb{F}_{q^2}^n$. We apply the matrix

$$T_n^{\mathrm{SUO}} = \begin{pmatrix} e_1 & e_n & e_2 & e_{n-1} & \ldots & e_{\frac{n+1}{2}-1} & e_{\frac{n+1}{2}+1} & e_{\frac{n+1}{2}} \end{pmatrix} \in \mathrm{GL}(n, q^2)$$

to effect a basis transformation on $\widetilde{s}, \widetilde{t}, \widetilde{\delta}, \widetilde{v}, \widetilde{u}$ and we apply

$$T_n^{\mathrm{SUOX}} = \begin{pmatrix} e_n & e_{\frac{n+1}{2}} & e_1 & e_{\frac{n+1}{2}+1} & e_2 & \ldots & e_{n-1} & e_{\frac{n+1}{2}-1} \end{pmatrix} \in \mathrm{GL}(n, q^2)$$

to effect a basis transformation on $\widetilde{x}$ and $\widetilde{y}$.

Example 4.29: For $n = 7$ we have

$$T_7^{\mathrm{SUO}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \text{ and } T_7^{\mathrm{SUOX}} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

When we use the matrices T_n^{SUO} and T_n^{SUOX} to change the basis, we obtain the following generating set.

Definition 4.30: Let q be a prime power, say $q = p^f$, n a positive odd integer, $\gamma \in \mathbb{F}_{q^2}$ a specified primitive element and $\alpha = \gamma^{(q+1)/2}$. The *Leedham-Green-O'Brien standard generators* [GB] of $\mathrm{SU}(n, q)$ transformed with T_n^{SUO} and T_n^{SUOX} are the following:

A transposition : $$s = \begin{pmatrix} 0 & 0 & \alpha \\ 0 & I_{n-2} & 0 \\ \alpha^{-q} & 0 & 0 \end{pmatrix},$$

A transvection : $$t = I_n + \alpha \cdot E_{1,n},$$

A diagonal matrix : $$\delta = \mathrm{diag}(\gamma^{q+1}, 1, \ldots, 1, \gamma^{-(q+1)}),$$

Product of two cycles of length $\frac{n}{2}$: $$v = \begin{pmatrix} 0 & I_{\frac{n+1}{2}-2} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & I_{\frac{n+1}{2}-2} & 0 \end{pmatrix},$$

Two 2 cycles : $$u = \begin{pmatrix} J_2 & 0 & 0 \\ 0 & I_{n-4} & 0 \\ 0 & 0 & J_2 \end{pmatrix},$$

A transvection : $$x = I_n + (-1) \cdot E_{\frac{n+1}{2},1} + (-2^{-1}) \cdot E_{n,1} + E_{n,\frac{n+1}{2}},$$

A diagonal matrix : $$y = \mathrm{diag}(\gamma^{-q}, 1, \ldots, 1, \gamma) + (\gamma^{q-1} - 1) E_{\frac{n+1}{2},\frac{n+1}{2}}.$$

Example 4.31: We get the following generating set for $n = 5$ and $q = 7$:

$$s = \begin{pmatrix} 0 & 0 & 0 & 0 & \gamma^4 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \gamma^{20} & 0 & 0 & 0 & 0 \end{pmatrix}, \quad t = \begin{pmatrix} 1 & 0 & 0 & 0 & \gamma^4 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad \delta = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 5 \end{pmatrix},$$

$$v = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad u = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad x = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 6 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 3 & 0 & 1 & 0 & 1 \end{pmatrix},$$

$$y = \begin{pmatrix} \gamma^{41} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \gamma^6 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \gamma \end{pmatrix}.$$

Remark 4.32: Notice that $u^{-1} = u$.

Now we can continue as in the paper "Straight-line programs with memory and matrix Bruhat decomposition" from A. C. Niemeyer, T. Popiel and C. E. Praeger [NPP].

First we describe the matrices for the Algorithm UnitriangularDecompositionSUEven and the Algorithm UnitriangularDecompositionSUOdd with the LGO generators. Again we have to make a case distinction since the generating set for SU is different for odd and even dimensions. We start with the even dimension.

Theorem 4.33: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element of $\mathbb{F}_{q^2}$. Let $(s, t, \delta, v, u, x, y)$ be the LGO standard generators as in Definition 4.27. We set

$$u_1 \coloneqq u,$$
$$u_i \coloneqq v^{-1} u_{i-1} v.$$

Then all matrices of the Algorithm UnitriangularDecompositionSUEven (Algorithm 3) can be written as a word of the LGO standard generators as shown in the following tables:

Entry in the top left or bottom right block.

Matrix	**Word in terms of the LGO generators**
$T_{2,1}(\omega^\ell)$	$(y^{-\ell}v^{-1}y^{-\ell}v)u^{-1}xu(y^{-\ell}v^{-1}y^{-\ell}v)^{-1}$
$T_{i+1,i}(\omega^\ell)$	$v^{-1}T_{i,i-1}(\omega^\ell)v$
$T_{i,j}(\omega^\ell)$	$u_jT_{i,j+1}(\omega^\ell)u_j$

Entry on the anti diagonal.

Matrix	**Word in terms of the LGO generators**
$T_{n,1}(\alpha^{-q}(\omega^{q+1})^\ell)$	$y^{-\ell}s^{-1}tsy^\ell$
$T_{i-1,j+1}(\alpha^{-q}(\omega^{q+1})^\ell)$	$v^{-1}T_{i,j}((\omega^{q+1})^\ell)v$

Entry in the bottom left block.

Matrix	**Word in terms of the LGO generators**
$T_{n,2}(\omega^\ell)$	$s^{-1}u^{-1}(y^{-(\ell+\frac{q+1}{2})}v^{-1}y^{-(\ell+\frac{q+1}{2})}v)u^{-1}xu(y^{-(\ell+\frac{q+1}{2})}v^{-1}y^{-(\ell+\frac{q+1}{2})}v)^{-1}us$
$T_{n,i}(\omega^\ell)$	$u^{-1}v^{-1}T_{n,i-1}(\omega^\ell)vu$
$T_{j,i}(\omega^\ell)$	$u_{n-j+1}T_{j-1,i}(\omega^\ell)u_{n-j+1}$

Notice that $\alpha^{-q}(\omega^{q+1})^\ell$ generates all solutions of $\imath + \bar{\imath} = 0$ for $\imath \in \mathbb{F}_{q^2} - \{0\}$.

Proof. We start with the expression $T_{2,1}(\omega^\ell) = (y^{-\ell}v^{-1}y^{-\ell}v)u^{-1}xu(y^{-\ell}v^{-1}y^{-\ell}v)^{-1}$. Let $\{e_1, \dots, e_n\}$ be the standard basis of $\mathbb{F}_{q^2}^n$. First we see that

$$v^{-1} : \begin{cases} e_i & \mapsto e_{i-1} \quad \text{for } i \in \{2, \dots, m\}, \\ e_1 & \mapsto e_m \\ e_n & \mapsto e_{m+1} \\ e_i & \mapsto e_{i+1} \quad \text{for } i \in \{m+1, \dots, n-1\}. \end{cases}$$

Let $h = \operatorname{diag}(b_1, \dots, b_n) \in \mathbb{F}_{q^2}^{n \times n}$ be any diagonal matrix. We have

$$\begin{aligned} &e_iv^{-1}hv = e_{i-1}hv = b_{i-1}e_{i-1}v = b_{i-1}e_i, && \text{for } i \in \{2, \dots, m\} \\ &e_1v^{-1}hv = e_mhv = b_me_mv = b_me_1, \\ &e_iv^{-1}hv = e_{i+1}hv = b_{k+1}e_{k+1}v = b_{k+1}e_k, && \text{for } i \in \{m+1, \dots, n-1\} \\ &e_nv^{-1}hv = e_{m+1}hv = b_{m+1}e_{m+1}v = b_{m+1}e_n, \end{aligned}$$

and therefore $v^{-1}hv = \operatorname{diag}(b_m, b_1, b_2, \dots, b_{m-1}, b_{m+2}, b_{m+3}, \dots, b_n, b_{m+1})$. In particular for y this shows

$$v^{-1}y^{-\ell}v = \operatorname{diag}(1, \omega^{-\ell}, \omega^\ell, 1, \dots, 1, \omega^{-\ell q}, \omega^{\ell q}, 1).$$

Hence

$$y^{-\ell}v^{-1}y^{-\ell}v = \operatorname{diag}(\omega^{-\ell}, 1, \omega^\ell, 1, \dots, 1, \omega^{-\ell q}, 1, \omega^{\ell q}).$$

Now we have

$$u^{-1}xu = I_n + E_{2,1} + (-1) \cdot E_{n,n-1}$$

and, therefore, $(y^{-\ell}v^{-1}y^{-\ell}v)u^{-1}xu(y^{-\ell}v^{-1}y^{-\ell}v)^{-1}$ can be verified directly by matrix multiplication of the upper-left 3×3 block and lower-right 3×3 block. First we look at the upper-left 3×3 block and notice that

$$\begin{pmatrix} \omega^{-\ell} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \omega^{\ell} \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \omega^{\ell} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \omega^{-\ell} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ \omega^{\ell} & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In the lower-right 3 x 3 block we have

$$\begin{pmatrix} \omega^{-\ell q} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \omega^{\ell q} \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} \omega^{\ell q} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \omega^{-\ell q} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\omega^{q\ell} & 1 \end{pmatrix}$$

And, therefore, the first claim follows. Now we have for $i \in \{2, \dots, m-1\}$

$$\begin{aligned} &e_{i+1}v^{-1}T_{i,i-1}(\omega^{\ell})v = e_iT_{i,i-1}(\omega^{\ell})v = (e_i + e_{i-1}\omega^{\ell})v = e_{i+1} + e_i\omega^{\ell}, \\ &e_kv^{-1}T_{i,i-1}(\omega^{\ell})v = e_{k-1}T_{i,i-1}(\omega^{\ell})v = e_{k-1}v = e_k, \qquad \text{for } k \in \{2, \dots, m\} - \{i+1\}, \\ &e_1v^{-1}T_{i,i-1}(\omega^{\ell})v = e_mT_{i,i-1}(\omega^{\ell})v = e_mv = e_1 \end{aligned}$$

and similarly, this can be checked for the bottom right block.
Hence $v^{-1}T_{i,i-1}(\omega^{\ell})v = T_{i+1,i}(\omega^{\ell})$. Moreover we have for $j \in \{1, \dots, i-2\}$

$$\begin{aligned} &e_iu_{j-1}T_{i,j}(\omega^{\ell})u_{j-1} = e_iT_{i,j}(\omega^{\ell})u_{j-1} = (e_i + \omega^{\ell}e_j)u_{j-1} = e_i + \omega^{\ell}e_{j-1}, \\ &e_{j-1}u_{j-1}T_{i,j}(\omega^{\ell})u_{j-1} = e_jT_{i,j}(\omega^{\ell})u_{j-1} = e_ju_{j-1} = e_{j-1}, \\ &e_ju_{j-1}T_{i,j}(\omega^{\ell})u_{j-1} = e_{j-1}T_{i,j}(\omega^{\ell})u_{j-1} = e_{j-1}u_{j-1} = e_j, \\ &e_ku_{j-1}T_{i,j}(\omega^{\ell})u_{j-1} = e_kT_{i,j}(\omega^{\ell})u_{j-1} = e_ku_{j-1} = e_k, \qquad \text{for } k \in \{1, \dots, m\} - \{i, j, j-1\} \end{aligned}$$

and similarly, this can be checked for the bottom right block.
So $u_{j-1}T_{i,j}(\omega^{\ell})u_{j-1} = T_{i,j-1}(\omega^{\ell})$.

For an entry on the anti diagonal we start with $y^{-\ell}s^{-1}tsy^{\ell}$. We have $s^{-1}ts = I_n + \alpha^{-q}E_{n,1}$ and therefore

$$y^{-\ell}s^{-1}tsy^{\ell} = I_n + \alpha^{-q}\omega^{\ell+q\ell}E_{n,1}.$$

Now $\omega^{\ell+q\ell} = (\omega^{q+1})^{\ell}$. Since $\omega^{q^2-1} = 1$ it follows that $(\omega^{q+1})^{q-1} = \omega^{q^2-1} = 1$. Moreover, we have that $q\ell + \ell < q^2 - 1$ for all $\ell < q-1$ and thus the order of ω^{q+1} is $q-1$. We show that $\alpha^{-q}(\omega^{q+1})^{\ell}$ generates all solutions of $\imath + \overline{\imath} = 0$ for $\imath \in \mathbb{F}_{q^2} - \{0\}$.
Let $\mathbb{F}_0 = \mathrm{Fix}_{\mathbb{F}_{q^2}}(\,^-)$. By Hilbert's Theorem 90 [NTWA, Theorem 4] we know that

$$\varphi\colon \mathbb{F} \to \mathbb{F}_0, x \mapsto x + \overline{x}$$

is surjective. Let $0 \neq \jmath_0 \in \mathbb{F}_{q^2}$ be a solution for $\jmath_0 + \overline{\jmath_0} = 0$. (This element has to exist, since the entry $(s^{-1}ts)_{n,1}$ is a non-trivial solution). Then $k\jmath_0$ is also a solution for $k \in \mathbb{F}_0$ since

$$k\jmath_0 + \overline{k\jmath_0} = k\jmath_0 + \overline{k}\overline{\jmath_0} = k\jmath_0 + k\overline{\jmath_0} = k(\jmath_0 + \overline{\jmath_0}) = k \cdot 0 = 0.$$

Therefore, we have at least q solutions. Let $\jmath_\xi \in \mathbb{F}_{q^2}$ be a solution for $\jmath_\xi + \overline{\jmath_\xi} = \xi$ for $\xi \in \mathbb{F}_0 - \{0\}$. This element has to exist since φ is surjective. Then $k\jmath_0 + \jmath_\xi$ is also a solution for $k \in \mathbb{F}_0$ since

$$\begin{aligned} k\jmath_0 + \jmath_\xi + \overline{k\jmath_0 + \jmath_\xi} &= k\jmath_0 + \jmath_\xi + \overline{k} \cdot \overline{\jmath_0} + \overline{\jmath_\xi} \\ &= k\jmath_0 + \jmath_\xi + k \cdot \overline{\jmath_0} + \overline{\jmath_\xi} \\ &= k(\jmath_0 + \overline{\jmath_0}) + \jmath_\xi + \overline{\jmath_\xi} \\ &= k \cdot 0 + \xi = \xi. \end{aligned}$$

Therefore, we also have at least q solutions. Now $\mathrm{Order}(\mathbb{F}_0) = q$ and we have q solutions for each element of $\mathbb{F}_0$, so we have found the image of every element in $\mathbb{F}_{q^2}$, since $\mathrm{Order}(\mathbb{F}_{q^2}) = q^2 = q \cdot q$. Hence, $\alpha^{-q}(\omega^{q+1})^\ell$ generates all solutions of $\imath + \bar{\imath} = 0$ except 0.
For $k < m$ we have that $e_k v^{-1} = e_{k'}$ for $k' \in \{1, \ldots, m\}$ and thus $e_k v^{-1} T_{i,j}(\alpha^{-q}(\omega^{q+1})^\ell)v = e_k$. Moreover, we set $\beta = \alpha^{-q}(\omega^{q+1})^\ell$ and have for $i \in \{n, \ldots, m+2\}, j = n - i + 1$

$$\begin{aligned} &e_{i-1}v^{-1}T_{i,j}(\beta)v = e_i T_{i,j}(\beta)v = (e_i + \beta e_j)v = e_{i-1} + \beta e_{j+1}, \\ &e_k v^{-1}T_{i,j}(\beta)v = e_{k+1}T_{i,j}(\beta)v = e_{k+1}v = e_k, \\ &e_n v^{-1}T_{i,j}(\beta)v = e_{m+1}T_{i,j}(\beta)v = e_{m+1}v = e_n \end{aligned}$$

for $k \in \{m+1, \ldots, n-1\} - \{i-1\}$. Therefore $v^{-1}T_{i,j}(\beta)v = T_{i-1,j+1}(\beta)$.

We already know that $T_{2,1}(\omega^\ell) = (y^{-\ell}v^{-1}y^{-\ell}v)u^{-1}xu(y^{-\ell}v^{-1}y^{-\ell}v)^{-1}$ and therefore $uT_{2,1}(\omega^\ell)u = T_{1,2}(\omega^\ell)$. Now $T_{1,2}(\omega^\ell)s$ induces a column operation on $T_{1,2}(\omega^\ell)$ where we replace the last column by α times the first column and the first column by α^{-q} times the last column. Hence

$$T_{1,2}(\omega^\ell)s = \begin{pmatrix} 0 & \omega^\ell & 0 & 0 & \alpha \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & I_{n-4} & 0 & 0 \\ -\alpha^{-q}\omega^{ql} & 0 & 0 & 1 & 0 \\ \alpha^{-q} & 0 & 0 & 0 & 0 \end{pmatrix}.$$

$s^{-1}(T_{1,2}(\omega^\ell)s)$ induces a row operation on $T_{1,2}(\omega^\ell)s$ where we replace the first row by α^q times the last row and the last row by α^{-1} times the first row.
Hence $s^{-1}T_{1,2}(\omega^\ell)s = I_n - \alpha^{-q}\omega^{q\ell}E_{n-1,1} + \alpha^{-1}\omega^\ell E_{n,2}$. Now $\alpha^{-1}\omega^\ell = \omega^k$ if and only if $\ell = k + \frac{q+1}{2}$. Moreover $T_{n,i}(a) = T_{1,n-i+1}(-\overline{a})$ and we have

$$e_1 u v^{-1} T_{1,n-i+1}(-\overline{a})vu = e_1 T_{1,n-i+1}(-\overline{a})vu = (e_1 - \overline{a}e_{n-i+1})vu = e_1 - \overline{a}e_{n-i}.$$

Since $e_k uv^{-1} \neq e_1$ for all $k \in \{2, \dots, m\}$ it follows that $uv^{-1}T_{n,i-1}(a)vu = uv^{-1}T_{1,n-i+1}(-\overline{a})vu = T_{1,n-i}(-\overline{a}) = T_{n,i+1}(a)$. From the block in the top left follows directly that $T_{j,i}(\omega^\ell) = u_{n-j+1}T_{j-1,i}(\omega^\ell)u_{n-j+1}$. □

With this we can estimate the length and the amount of memory slots for an MSLP, which computes the Bruhat decomposition for an element in the special unitary group of even dimension.

Theorem 4.34: Let $q = p^f$ with p prime, $f \geq 1$, $n = 2m$ an even integer with $n \geq 3$ and $a \in \mathrm{SU}(n, q)$. Set $b = 18 + 5f + m$ and

$$\begin{aligned}\lambda = &45f + 2\log_2(\frac{q+1}{2}) + n - 6 + n\log_2(m) + m + mf + n\log_2(q) + \\ &(n^2 - 2n)(f + 2\log_2(q) + \log_2(m-2) + m).\end{aligned}$$

There exists a b-MSLP, S, of length at most λ such that if S is evaluated with memory containing the list $\{s, t, \delta, v, u, x, y\}$ then S outputs memory containing (u_1, u_2) with $a = u_1 w u_2$ the Bruhat decomposition of a.

Proof. In order to avoid the logarithm oracle we use the following method to calculate the transvections.
Since $\mathbb{F}_{q^2}$ is an $\mathbb{F}_p$ vector space with basis $\{\omega^0, \omega^1, \dots, \omega^{2f-1}\}$, where ω is a primitive element of $\mathbb{F}_{q^2}$, we only need to save the transvections $T_{2,1}(\omega^0), T_{2,1}(\omega^1), \dots, T_{2,1}(\omega^{2f-1})$ and obtain $T_{2,1}(\beta)$ for $\beta \in \mathbb{F}_{q^2}$ through

$$T_{2,1}(\beta) = T_{2,1}(\sum_{i=0}^{2f-1} k_i\omega^i) = \prod_{i=0}^{2f-1} T_{2,1}(\omega^i)^{k_i}.$$

Hence, we just calculate the transvections $T_{2,1}(\omega^0), T_{2,1}(\omega^1), \dots, T_{2,1}(\omega^{2f-1})$ and store them. We use the same method for the transvections with entries in the lower left block and entries on the anti diagonal.
Therefore, we start to calculate these transvections. First we look at the transvections $T_{2,1}(\omega^i)$ for $i \in \{0, \dots, 2f-1\}$ and use the formula of Theorem 4.33. We need 4 operations for $(y^{-\ell}v^{-1}y^{-\ell}v)$ and thus 8 operations in total for

$$(y^{-\ell}v^{-1}y^{-\ell}v)u^{-1}xu(y^{-\ell}v^{-1}y^{-\ell}v)^{-1}.$$

Since we do this $2f$ times, we need $8 \cdot 2f = 16f$ operations. Analogously follows that we need at most $24f + 2\log_2(\frac{q+1}{2})$ operations for the

$$T_{n,2}(\omega^\ell) = s^{-1}u^{-1}(y^{-(\ell+\frac{q+1}{2})}v^{-1}y^{-(\ell+\frac{q+1}{2})}v)u^{-1}xu(y^{-(\ell+\frac{q+1}{2})}v^{-1}y^{-(\ell+\frac{q+1}{2})}v)^{-1}us$$

for $i \in \{0, \dots, 2f-1\}$ and $5f$ operations for the

$$T_{n,1}(\alpha^{-q}(\omega^{q+1})^{\ell}) = y^{-\ell}s^{-1}tsy^{\ell}$$

for $i \in \{0, \ldots, f-1\}$. Moreover we store the u_i for $i \in \{1, \ldots, m-2\}$ of Theorem 4.33 and need $2(m-3) = n-6$ operations to calculate them. In total we need $45f + 2\log_2(\frac{q+1}{2}) + n - 6$ operations for the initialization.
Now starts the Algorithm 3. We consider a matrix with a non-zero entry in every position. We need at most $2f - 1 + 4f\log_2(p) = 2f - 1 + 4\log_2(q)$ operations to calculate a transvection $T_{2,1}(\beta)$ or $T_{n,2}(\beta)$ and at most $f - 1 + 2f\log_2(p) = f - 1 + 2\log_2(q)$ operations to calculate a transvection $T_{n,1}(\beta)$. To shift $T_{n,1}(\beta)$ to $T_{m+1,m}(\beta)$ we need with repeated squaring at most $2\log_2(m) + 2$ operations. Therefore, we need at most

$$m(2\log_2(m) + 2 + f - 1 + 2\log_2(q)) = n\log_2(m) + m + mf + n\log_2(q)$$

operations for the anti diagonal.
To shift $T_{n,2}(\beta)$ by the index i we need at most $3 + 2\log_2(m-2)$ operations and by the index j we need at most $(m-1)2$ operations. At most we need to do this $m^2 - m = \frac{n^2-2n}{4}$ times. If we sum up this results, we get

$$\tfrac{n^2-2n}{2}(f + 2\log_2(q) + \log_2(m-2) + m)$$

Analogously follows that we need at most

$$\tfrac{n^2-2n}{4}(2f - 1 + 4\log_2(q) + 2 + 2\log_2(m-2) + (m-1)2) \leq \tfrac{n^2-2n}{2}(f + 2\log_2(q) + \log_2(m-2) + m)$$

for all the transvections in the top left or lower right block, yielding the claimed maximum total length of the MSLP:

$$45f + 2\log_2(\tfrac{q+1}{2}) + n - 6 + n\log_2(m) + m + mf + n\log_2(q) + (n^2 - 2n)(f + 2\log_2(q) + \log_2(m-2) + m)$$

We need 14 memory slots for the LGO standard generator and their inverse elements, 2 memory slots for repeated squaring, 2 memory slots for the results u_1 and u_2, 1 memory slot for the current transvection and 1 additional memory slot to save some calculations.
Moreover we need $2f$ memory slots for the matrices $T_{2,1}(\omega^i)$ for $i \in \{0, \ldots, 2f-1\}$, $2f$ memory slots for the matrices $T_{n,2}(\omega^i)$ for $i \in \{0, \ldots, 2f-1\}$, f memory slots for the matrices $T_{n,1}(\omega^{2i})$ for $i \in \{0, \ldots, f-1\}$ and $m-2$ memory slots for the remaining u_i for $i \in \{1, \ldots, m-2\}$. Thus we need

$$14 + 2 + 2 + 1 + 1 + 2f + 2f + f + m - 2 = 18 + 5f + m$$

memory slots. □

Now we describe the matrices for odd dimension.

Theorem 4.35: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m+1$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element of $\mathbb{F}_{q^2}$. Let $(s, t, \delta, v, u, x, y)$ be the LGO standard generators as in Definition 4.30. We set

$$u_1 \coloneqq u,$$
$$u_i \coloneqq v^{-1} u_{i-1} v.$$

Then all matrices of the Algorithm UnitriangularDecompositionSUOdd (Algorithm 4) can be written as a word of the LGO standard generators as shown in the following tables:

Entry in the bottom left block.

Matrix	**Word in terms of the LGO generators**
$T_{n,2}(\omega^\ell)$	$(y^\ell v^{-2} y^{-\ell} v^2)(v^{-1}xvx^{-1}v^{-1}x^{-1}vx)^{-1}(y^\ell v^{-2} y^{-\ell} v^2)^{-1}$
$T_{n,i}(\omega^\ell)$	$u^{-1}v^{-1}T_{n,i-1}(\omega^\ell)vu$
$T_{j,i}(\omega^\ell)$	$u_{n-j+1}T_{j-1,i}(\omega^\ell)u_{n-j+1}$

Entry on the anti diagonal.

Matrix	**Word in terms of the LGO generators**
$T_{n,1}(\alpha^{-q}(\omega^{q+1})^\ell)$	$y^\ell s^{-1} t s y^{-\ell}$
$T_{i-1,j+1}(\alpha^{-q}(\omega^{q+1})^\ell)$	$v^{-1}T_{i,j}((\omega^{q+1})^\ell)v$

Entry on the center row and column.

Matrix	**Word in terms of the LGO generators**
$T_{n,\frac{n+1}{2}}(\omega^\ell)$	$(y^\ell v^{-2} y^{-\ell} v^2)x(y^\ell v^{-2} y^{-\ell} v^2)^{-1}$
$T_{i,\frac{n+1}{2}}(\omega^\ell)$	$v^{-1}T_{i-1,\frac{n+1}{2}}(\omega^\ell)v$

Entry in the top left or bottom right block.

Matrix	**Word in terms of the LGO generators**
$T_{2,1}(\omega^\ell)$	$u^{-1}s^{-1}(y^{\ell-\frac{q^2+q}{2}}v^{-2}y^{-(\ell-\frac{q^2+q}{2})}v^2)\cdot(v^{-1}xvx^{-1}v^{-1}x^{-1}vx)^{-1}\cdot$ $(y^{\ell-\frac{q^2+q}{2}}v^{-2}y^{-(\ell-\frac{q^2+q}{2})}v^2)^{-1}su$
$T_{i+1,i}(\omega^\ell)$	$v^{-1}T_{i,i-1}(\omega^\ell)v$
$T_{i,j}(\omega^\ell)$	$u_j T_{i,j+1}(\omega^\ell)u_j$

Notice that $\alpha^{-q}(\omega^{q+1})^\ell$ generates all solutions of $\imath + \bar{\imath} = 0$ for $\imath \in \mathbb{F}_{q^2} - \{0\}$.

Proof. Let $h \coloneqq \mathrm{diag}(b_1, \ldots, b_n)$ be a diagonal matrix. We have seen in Theorem 4.33 that $v^{-1}hv = \mathrm{diag}(b_m, b_1, b_2, \ldots, b_{m-1}, b_{m+1}, b_{m+3}, b_{m+4}, \ldots, b_n, b_{m+2})$. And therefore $v^{-2}y^{-\ell}v^2 = \mathrm{diag}(1, 1, (\omega^{-q})^{-\ell}, 1, \ldots 1, (\omega^{q-1})^{-\ell}, 1, \ldots, 1, \omega^{-\ell}, 1, 1)$. Thus

$$y^\ell v^{-2} y^{-\ell} v^2 = \mathrm{diag}((\omega^{-q})^\ell, 1, (\omega^{-q})^{-\ell}, \ldots, 1, \omega^{-\ell}, 1, \omega^\ell).$$

Now $v^{-1}xv = I_n + (-1) \cdot E_{\frac{n+1}{2},2} + (-2^{-1}) \cdot E_{n-1,2} + E_{n-1,\frac{n+1}{2}}$ and $v^{-1}x^{-1}v = I_n + E_{\frac{n+1}{2},2} + (-2^{-1}) \cdot E_{n-1,2} + (-1) \cdot E_{n-1,\frac{n+1}{2}}$ and therefore

$$x^{-1}v^{-1}x^{-1}vx = I_n + E_{\frac{n+1}{2},2} + (-2^{-1}) \cdot E_{n-1,2} + (-1) \cdot E_{n-1,\frac{n+1}{2}} + E_{n-1,1} + (-1) \cdot E_{n,2}$$

and so it follows that

$$\begin{aligned} v^{-1}xvx^{-1}v^{-1}x^{-1}vx &= I_n + E_{n-1,1} + (-1) \cdot E_{n,2}, \\ (v^{-1}xvx^{-1}v^{-1}x^{-1}vx)^{-1} &= I_n + (-1) \cdot E_{n-1,1} + E_{n,2}. \end{aligned}$$

Hence

$$(y^\ell v^{-2} y^{-\ell} v^2)(v^{-1}xvx^{-1}v^{-1}x^{-1}vx)^{-1}(y^\ell v^{-2}y^{-\ell}v^2)^{-1} = I_n + (-1)w^{q\ell}E_{n-1,1} + w^\ell \cdot E_{n,2}.$$

This shows the first claim. We have already shown in Theorem 4.33 that

$$\begin{aligned} T_{n,i}(\omega^\ell) &= u^{-1}v^{-1}T_{n,i-1}(\omega^\ell)vu, \\ T_{j,i}(\omega^\ell) &= u_{n-j+1}T_{j-1,i}(\omega^\ell)u_{n-j+1}. \end{aligned}$$

We continue with an entry on the anti diagonal. We also have seen that $s^{-1}ts = I_n + \alpha^{-q}E_{n,1}$ and in SU odd we have that

$$y^\ell s^{-1}tsy^{-\ell} = I_n + \alpha^{-q}\omega^{w^{\ell+q\ell}}E_{n,1}.$$

The rest for an entry on the anti diagonal has already been shown in Theorem 4.33.
We continue with the matrices for the center row. We already know that

$$y^\ell v^{-2}y^{-\ell}v^2 = \mathrm{diag}((\omega^{-q})^\ell, 1, (\omega^{-q})^{-\ell}, \ldots, 1, \omega^{-\ell}, 1, \omega^\ell).$$

Therefore

$$(y^\ell v^{-2}y^{-\ell}v^2)x(y^\ell v^{-2}y^{-\ell}v^2)^{-1} = I_n + (-1)\omega^{q\ell} \cdot E_{\frac{n+1}{2},1} + (-2^{-1})\omega^{q+q\ell} \cdot E_{n,1} + \omega^\ell E_{n,\frac{n+1}{2}}.$$

We show that $T_{i,\frac{n+1}{2}}(\omega^\ell) = v^{-1}T_{i-1,\frac{n+1}{2}}(\omega^\ell)v$. We know that $T_{i,\frac{n+1}{2}}(a) = T_{\frac{n+1}{2},n-i+1}(-\overline{a})$. Since

$$e_{\frac{n+1}{2}}v^{-1}T_{\frac{n+1}{2},n-i+1}(-\overline{a})v = e_{\frac{n+1}{2}}T_{\frac{n+1}{2},n-i+1}(-\overline{a})v = (e_{\frac{n+1}{2}} - \overline{a}e_{n-i+1})v = e_{\frac{n+1}{2}} - \overline{a}e_{n-i}$$

and $e_kv^{-1} \neq e_{\frac{n+1}{2}}$ for $k \in \{1, \ldots, n\} - \{\frac{n+1}{2}\}$ it follows that $v^{-1}T_{\frac{n+1}{2},n-i+1}(-\overline{a})v = T_{\frac{n+1}{2},n-i}(-\overline{a})$. Hence $v^{-1}T_{i,\frac{n+1}{2}}(a)v = v^{-1}T_{\frac{n+1}{2},n-i+1}(-\overline{a})v = T_{\frac{n+1}{2},n-i}(-\overline{a}) = T_{i-1,\frac{n+1}{2}}$.
The block in the top left remains. We prove this in analogy to Theorem 4.33. First we compute $T_{n,2}(\omega^\ell)s$. Now $T_{n,2}(\omega^\ell)s$ induces a column operation on $T_{n,2}(\omega^\ell)$ where we replace the last column by α times the first column and the first column by α^{-q} times the last column. Hence

$$T_{n,2}(\omega^\ell)s = \begin{pmatrix} 0 & 0 & 0 & 0 & \alpha \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & I_{n-4} & 0 & 0 \\ 0 & 0 & 0 & 1 & -\alpha\omega^{q\ell} \\ \alpha^{-q} & \omega^\ell & 0 & 0 & 0 \end{pmatrix}.$$

Now $s^{-1}(T_{n,2}(\omega^\ell)s)$ induces a row operation on $T_{n,2}(\omega^\ell)s$ where we replace the first row by α^q times the last row and the last row by α^{-1} times the first row. Hence $s^{-1}(T_{n,2}(\omega^\ell)s) = I_n + \alpha^q\omega^\ell E_{1,2} + -\alpha\omega^{q\ell}E_{n-1,n}$ and thus

$$u^{-1}s^{-1}T_{n,2}(\omega^\ell)su = T_{2,1}(\alpha^q\omega^\ell).$$

Since $\alpha^q\omega^\ell = w^k$ if and only if $\ell = k - \frac{q^2+q}{2}$ the last claim follows. The remaining part is obtained by Theorem 4.33. □

With this we can estimate the length and the amount of memory slots for an MSLP, which computes the Bruhat decomposition for an element in the special unitary group of odd dimension.

Theorem 4.36: Let $q = p^f$ with p prime, $f \geq 1$, $n = 2m+1$ an odd integer with $n \geq 3$ and $a \in \mathrm{SU}(n,q)$. Set $b = 18 + 7f + m$ and

$$\begin{aligned}\lambda =& 61f + n + 1 + 2\log_2(\frac{q^2+q}{2}) + (n-1)\log_2(m) + m + mf + (n-1)\log_2(q) + \\ & (n-1)(2\log_2(m) + 1 + 2f + 4\log_2(q)) + (n^2 - 4n + 3)(f + 2\log_2(q) + \log_2(m-2) + m).\end{aligned}$$

There exists a b-MSLP, S, of length at most λ such that if S is evaluated with memory containing the list $\{s, t, \delta, v, u, x, y\}$ then S outputs memory containing (u_1, u_2) with $a = u_1 w u_2$ the Bruhat decomposition of a.

Proof. With the same argument as in Theorem 4.34 we just calculate for example the transvections $T_{2,1}(\omega^0), T_{2,1}(\omega^1), \ldots, T_{2,1}(\omega^{2f-1})$ and store them. We use the same method for the transvections with entries in the lower left block, entries on the anti diagonal and entries on the centre row and column.
Therefore, we start to calculate these transvections. First we look at the transvections $T_{2,1}(\omega^i)$ for $i \in \{0, \ldots, 2f-1\}$ and use the formula of Theorem 4.35. We need 7 operations for $v^{-1}xvx^{-1}v^{-1}x^{-1}vx$ and $2\log_2(\frac{q^2+q}{2})$ operations for $y^{-\frac{q^2+q}{2}}$. Moreover we need at most 6 operations for $y^{\ell-\frac{q^2+q}{2}}v^{-2}y^{-(\ell-\frac{q^2+q}{2})}v^2$ and thus at most $12 + 7 + 2\log_2(\frac{q^2+q}{2})$ operations in total for

$$\begin{aligned}T_{2,1}(\omega^\ell) = u^{-1}s^{-1}(y^{\ell-\frac{q^2+q}{2}}v^{-2}y^{-(\ell-\frac{q^2+q}{2})}v^2) \cdot (v^{-1}xvx^{-1}v^{-1}x^{-1}vx)^{-1} \cdot \\ (y^{\ell-\frac{q^2+q}{2}}v^{-2}y^{-(\ell-\frac{q^2+q}{2})}v^2)^{-1}su.\end{aligned}$$

Since we do this $2f$ times, we need $24f + 7 + 2\log_2(\frac{q^2+q}{2})$ operations. Analogously follows that we need at most $16f$ operations for the

$$T_{n,2}(\omega^\ell) = (y^\ell v^{-2} y^{-\ell} v^2)(v^{-1}xvx^{-1}v^{-1}x^{-1}vx)^{-1}(y^\ell v^{-2} y^{-\ell} v^2)^{-1}$$

for $i \in \{0, \ldots, 2f-1\}$, $5f$ operations for the

$$T_{n,1}(\alpha^{-q}(\omega^{q+1})^\ell) = y^{-\ell} s^{-1} t s y^\ell$$

for $i \in \{0, \ldots, f-1\}$ and $16f$ operations for the

$$T_{n,\frac{n+1}{2}}(\omega^\ell) = (y^\ell v^{-2} y^{-\ell} v^2) x (y^\ell v^{-2} y^{-\ell} v^2)^{-1}$$

for $i \in \{0, \ldots, 2f-1\}$. Moreover we store the u_i for $i \in \{1, \ldots, m-2\}$ of Theorem 4.35 and need $2(m-3) = n-6$ operations to calculate them. In total we need $61f + n + 1 + 2\log_2(\frac{q^2+q}{2})$ operations for the initialization.
Now starts the Algorithm 4. We consider a matrix with a non-zero entry in every position. We need at most $2f-1+4f\log_2(p) = 2f-1+4\log_2(q)$ operations to calculate a transvection $T_{2,1}(\beta)$,$T_{n,\frac{n+1}{2}}(\beta)$ or $T_{n,2}(\beta)$ and at most $f - 1 + 2f\log_2(p) = f - 1 + 2\log_2(q)$ operations to calculate a transvection $T_{n,1}(\beta)$. To shift $T_{n,1}(\beta)$ to $T_{m,m-1}(\beta)$ we need with repeated squaring at most $2\log_2(m) + 2$ operations. Therefore, we need at most

$$m(2\log_2(m) + 2 + f - 1 + 2\log_2(q)) = (n-1)\log_2(m) + m + mf + (n-1)\log_2(q)$$

operations for the anti diagonal and

$$(n-1)(2\log_2(m) + 2 + 2f - 1 + 4\log_2(q))$$

operations for the centre row and column. To shift $T_{n,2}(\beta)$ by the index i we need at most $3 + 2\log_2(m-2)$ operations and by the index j we need at most $(m-1)2$ operations. At most we need to do this $m^2 - m = \frac{n^2-4n+3}{4}$ times. If we sum up this results, we get

$$\tfrac{n^2-4n+3}{2}(f + 2\log_2(q) + \log_2(m-2) + m)$$

Analogously follows that we need at most

$$\begin{aligned} &\frac{n^2-4n+3}{4}(2f - 1 + 4\log_2(q) + 2 + 2\log_2(m-2) + (m-1)2) \\ \leq &\frac{n^2-4n+3}{2}(f + 2\log_2(q) + \log_2(m-2) + m) \end{aligned}$$

for all the transvections in the top left or lower right block, yielding the claimed maximum total length of the MSLP:

$$\begin{gathered} 61f + n + 1 + 2\log_2(\tfrac{q^2+q}{2}) + (n-1)\log_2(m) + m + mf + (n-1)\log_2(q) + \\ (n-1)(2\log_2(m) + 1 + 2f + 4\log_2(q)) + (n^2-4n+3)(f + 2\log_2(q) + \log_2(m-2) + m) \end{gathered}$$

We need 14 memory slots for the LGO standard generator and their inverse elements, 2 memory slots for repeated squaring, 2 memory slots for the results u_1 and u_2, 1 memory slot for the current transvection and 1 additional memory slot to save some calculations.
Moreover we need $2f$ memory slots for the matrices $T_{2,1}(\omega^i)$ for $i \in \{0, \dots, 2f-1\}$, $2f$ memory slots for the matrices $T_{n,2}(\omega^i)$ for $i \in \{0, \dots, 2f-1\}$, $2f$ memory slots for the matrices $T_{n,\frac{n+1}{2}}(\omega^i)$ for $i \in \{0, \dots, 2f-1\}$, f memory slots for the matrices $T_{n,1}(\omega^{2i})$ for $i \in \{0, \dots, f-1\}$ and $m-2$ memory slots for the remaining u_i for $i \in \{1, \dots, m-2\}$. Thus we need

$$14 + 2 + 2 + 1 + 1 + 2f + 2f + 2f + f + m - 2 = 18 + 7f + m$$

memory slots. □

Remark 4.37: Let $q = p^f$ with p prime, $f \geq 1$ and $n = 2m+1$ an odd integer. In Algorithm 4, Lemma 4.15 and Example 4.14 we have considered the equation

$$\imath\bar{\imath} + \jmath + \bar{\jmath} = 0$$

for $\imath, \jmath \in \mathbb{F}_{q^2}$. In particular, in Algorithm 4 we have given $\alpha \in \mathbb{F}_{q^2}$ and are looking for a $x \in \mathbb{F}_{q^2}$ such that

$$\alpha\overline{\alpha} + x + \overline{x} = 0.$$

In Theorem 4.35 we have seen that $T_{n,\frac{n+1}{2}}(\alpha) \in \mathrm{SU}(n, q)$ and therefore

$$x \coloneqq (T_{n,\frac{n+1}{2}}(\alpha))_{n,1}$$

is a solution for the equation $\alpha\overline{\alpha} + x + \overline{x} = 0$. So we get a solution if we compute the entry in position $(n, 1)$ of

$$T_{n,\frac{n+1}{2}}(\alpha) = T_{n,\frac{n+1}{2}}(\sum_{i=0}^{2f-1} k_i\omega^i) = \prod_{i=0}^{2f-1} T_{n,\frac{n+1}{2}}(\omega^i)^{k_i}, \qquad \text{where } \alpha = \sum_{i=0}^{2f-1} k_i\omega^i.$$

We perform the calculation iteratively. We select $j \in \{0, \dots, 2f-1\}$ minimal with $k_j \neq 0$ and set $x = -2^{-1}(\omega^{q+1})^j, c = \omega^j$. Now we execute the matrix multiplication only at the entries in position $(n, 1)$ and $(n, \frac{n+1}{2})$. Let

$$\begin{aligned} ((T_{n,\frac{n+1}{2}}(\omega^j))^{i-1})_{n,1} &\to x \\ ((T_{n,\frac{n+1}{2}}(\omega^j))^{i-1})_{n,\frac{n+1}{2}} &\to c \end{aligned}$$

for $i \in \{1, \dots, k_j\}$.

We have

$$\begin{aligned}((T_{n,\frac{n+1}{2}}(\omega^j))^i)_{n,1} &= (T_{n,\frac{n+1}{2}}(\omega^j)(T_{n,\frac{n+1}{2}}(\omega^j))^{i-1})_{n,1} \\ &= ((T_{n,\frac{n+1}{2}}(\omega^j))^{i-1})_{n,1} + -2^{-1}(\omega^{q+1})^j + ((T_{n,\frac{n+1}{2}}(\omega^j))^{i-1})_{n,\frac{n+1}{2}} \cdot (\overline{-\omega^j}) \\ &= x + -2^{-1}(\omega^{q+1})^j + c \cdot (\overline{-\omega^j})\end{aligned}$$

and

$$((T_{n,\frac{n+1}{2}}(\omega^j))^i)_{n,\frac{n+1}{2}} = (T_{n,\frac{n+1}{2}}(\omega^j)(T_{n,\frac{n+1}{2}}(\omega^j))^{i-1})_{n,\frac{n+1}{2}} = c + \omega^j.$$

Thus, we have calculated $(T_{n,\frac{n+1}{2}}(\omega^j))^{k_j}$ at the position $(n,1)$ and $(n,\frac{n+1}{2})$. We now assume there exists a $\zeta \in \{j,\ldots,2f-1\}$ and $i \in \{0,\ldots,k_\zeta\}$ such that

$$\begin{aligned}(\prod_{i=0}^{\zeta-1} T_{n,\frac{n+1}{2}}(\omega^i)^{k_i}(T_{n,\frac{n+1}{2}}(\omega^\zeta))^i)_{n,1} &\to x, \\ (\prod_{i=0}^{\zeta-1} T_{n,\frac{n+1}{2}}(\omega^i)^{k_i}(T_{n,\frac{n+1}{2}}(\omega^\zeta))^i)_{n,\frac{n+1}{2}} &\to c.\end{aligned}$$

If $i < k_\zeta$, then we have

$$\begin{aligned}(T_{n,\frac{n+1}{2}}(\omega^\zeta)\prod_{i=0}^{\zeta-1} T_{n,\frac{n+1}{2}}(\omega^i)^{k_i}(T_{n,\frac{n+1}{2}}(\omega^\zeta))^i)_{n,1} &= x + -2^{-1}(\omega^{q+1})^\zeta + c \cdot (\overline{-\omega^\zeta}), \\ (T_{n,\frac{n+1}{2}}(\omega^\zeta)\prod_{i=0}^{\zeta-1} T_{n,\frac{n+1}{2}}(\omega^i)^{k_i}(T_{n,\frac{n+1}{2}}(\omega^\zeta))^i)_{n,\frac{n+1}{2}} &= c + \omega^\zeta.\end{aligned}$$

Otherwise, if $\zeta < 2f-1$ we have

$$\begin{aligned}(T_{n,\frac{n+1}{2}}(\omega^{\zeta+1})\prod_{i=0}^{\zeta} T_{n,\frac{n+1}{2}}(\omega^i)^{k_i})_{n,1} &= x + -2^{-1}(\omega^{q+1})^{\zeta+1} + c \cdot (\overline{-\omega^{\zeta+1}}), \\ (T_{n,\frac{n+1}{2}}(\omega^{\zeta+1})\prod_{i=0}^{\zeta} T_{n,\frac{n+1}{2}}(\omega^i)^{k_i}(T_{n,\frac{n+1}{2}}(\omega^\zeta))^i)_{n,\frac{n+1}{2}} &= c + \omega^{\zeta+1}.\end{aligned}$$

Finally $x = (T_{n,\frac{n+1}{2}}(\alpha))_{n,1}$ is a solution for $\alpha\overline{\alpha} + x + \overline{x} = 0$. Therefore, we need at most $4fp - 4f - 2$ field multiplications and $8fp - 8f - 4$ field additions to compute a solution.

Now we transform a monomial matrix in $\mathrm{SU}(n,q)$ into a diagonal matrix in $\mathrm{SU}(n,q)$ and write this with the LGO standard generators.

Let $M_{\mathrm{SU}} \leq \mathrm{SU}(n,q)$ be the subgroup of monomial matrices of $\mathrm{SU}(n,q)$. Let $\{e_1,\dots,e_n\}$ be the standard basis of $\mathbb{F}_{q^2}^n$. We define

$$\varphi_{\mathrm{SU}}\colon M_{\mathrm{SU}} \to S_n$$

such that $g \in M_{\mathrm{SU}}$ permutes the subspaces $\langle e_1\rangle,\dots,\langle e_n\rangle$ in the same way that $\varphi_{\mathrm{SU}}(g)$ permutes $1,\dots,n$.
Clearly $\varphi_{\mathrm{SU}}(M_{\mathrm{SU}})$ is generated by the images of the standard generators s,v,u from Definition 4.27 and Definition 4.30. For even n we set $m = \frac{n}{2}$ and get

$$\varphi_{\mathrm{SU}}(M_{\mathrm{SU}}) = \langle (1,n), (1,2)(n-1,n), (1,2,\dots,m)(m+1,n,n-1,\dots,m+2)\rangle =: G_{SUE}.$$

For odd n we set $m = \frac{n+1}{2}$ and get

$$\varphi_{\mathrm{SU}}(M_{\mathrm{SU}}) = \langle (1,n), (1,2)(n-1,n), (1,2,\dots,m-1)(m+1,n,n-1,\dots,m+2)\rangle =: G_{SUO}.$$

Let n be even, $2m = n$ and $\pi \in G_{SUE}$. We want to find an MSLP that outputs π as a word in these generators when evaluated with the input $(1,n),(1,2)(n-1,n)$ and $(1,2,\dots,m)(m+1,n,n-1,\dots,m+2)$. Let us assume that the inverse of $(1,2,\dots,m)(m+1,n,n-1,\dots,m+2)$ is also given as input.
First we divide $\Omega = \{1,\dots,n\}$ into two sets $\Omega_1 = \{1,\dots,m\}$ and $\Omega_2 = \{m+1,\dots,n\}$. We know that $S_m = S_{\Omega_1}$ is generated by $(1,2)$ and $(1,2,\dots,m)$ and, therefore, we can also write every element in S_m by $(1,2)(n-1,n)$ and $(1,2,\dots,m)(m+1,n,n-1,\dots,m+2)$ if we ignore the back part. We describe how we can do that later. Our first aim is to divide π into $\pi = g_1^{-1}\pi_1\pi_2 g_2^{-1}$ where $g_1,g_2 \in G_{SUE}$, $\pi_1 \in S_{\Omega_1}$ and $\pi_2 \in S_{\Omega_2}$.
For this let $\pi = z_1 \dots z_k$ where $z_i \in S_n$ is a cycle. We now go through π cycle for cycle. Therefore, let $i \in \{1,\dots,k\}$ and $g_1, g_2 = \mathrm{Id}_{S_n}$.
Case 1: The largest moved point of z_i is smaller than or equal to m or the smallest moved point of z_i is greater than m:
In this case we have nothing to do, since z_i has the required form. We can continue with the next cycle.
Case 2: Let $\imath$ be the largest moved point of z_i and $(n-\imath+1)^{z_i} = (n-\imath+1)$:
Since we are not in the first case, we know that z_i moves a point in Ω_1 and in Ω_2. We know that z_i stabilises $(n-\imath+1)$ and, therefore, $(n-\imath+1)$ is not in z_i. We use $(n-\imath+1,\imath) = (1,n)^{(1,2,\dots,m)(m+1,n,n-1,\dots,m+2)^{n-\imath+1}}$ by conjugation on z_i. Moreover, we save this action by $g_1 \leftarrow (n-\imath+1,\imath)g_1$ and $g_2 \leftarrow g_2(n-\imath+1,\imath)$. Again we go through the hole procedure for the cycle.
Case 3: Let $\imath$ be the largest moved point of z_i. Then z_i moves $\imath$ and $n-\imath+1$:
In this case we split z_i in two cycles.

Let $z_i = (\imath_1, \ldots, \imath_{\ell_1}, \imath, \imath_{\ell_1+1}, \ldots, \imath_{\ell_1+\ell_2}, n-\imath+1, \imath_{\ell_1+\ell_2+1}, \ldots, \imath_{\ell_1+\ell_2+\ell_3})$. Again we use $(n-\imath+1, \imath) = (1,n)^{(1,2,\ldots,m)(m+1,n,n-1,\ldots,m+2)^{n-\imath+1}}$, but this time by left multiplication. We have

$$\begin{aligned}
&(n-\imath+1, \imath) \cdot z_i \\
=&(n-\imath+1, \imath) \cdot (\imath_1, \ldots, \imath_{\ell_1}, \imath, \imath_{\ell_1+1}, \ldots, \imath_{\ell_1+\ell_2}, n-\imath+1, \imath_{\ell_1+\ell_2+1}, \ldots, \imath_{\ell_1+\ell_2+\ell_3}) \\
=&(n-\imath+1, \imath_{\ell_1+1}, \ldots, \imath_{\ell_1+\ell_2})(\imath, \imath_{\ell_1+\ell_2+1}, \ldots, \imath_{\ell_1+\ell_2+\ell_3}, \imath_1, \ldots, \imath_{\ell_1}).
\end{aligned}$$

We save this by $g_1 \leftarrow (n-\imath+1, \imath)g_1$. Again we go through the same procedure for both cycles.

In the end we have that $g_1 \pi g_2 = \pi_1 \pi_2$, where $g_1, g_2 \in G_{SUE}$, $\pi_1 \in S_{\Omega_1}$ and $\pi_2 \in S_{\Omega_2}$. Since $\pi_1\pi_2 \in G_{SUE}$ we have that $\pi_1\pi_2 \in \langle (1,2)(n-1,n), (1,2,\ldots,m)(m+1,n,n-1,\ldots,m+2)\rangle$. Now we can write π_1 as a word of $(1,2)$ and $(1,2,\ldots,m)$. Clearly we also get π_2, otherwise we would have to multiply $(1,2)(n-1,n)$ or $(1,2,\ldots,m)(m+1,n,n-1,\ldots,m+2)$ from left or right and, therefore, destroy π_1. That is a contradiction to

$$\pi_1\pi_2 \in \langle (1,2)(n-1,n), (1,2,\ldots,m)(m+1,n,n-1,\ldots,m+2)\rangle.$$

So it remains to write π_1 as a word of $(1,2)$ and $(1,2,\ldots,m)$. How to do this is described in chapter 3.3 [NPP]. Therefore, we only give the result again. Let $\pi \in S_m$ be a permutation. Moreover, we set $s_1 \coloneqq (1,2)$ and $v_1 \coloneqq (1,m,m-1,\ldots,2) = (1,2,\ldots,m)^{-1}$. We have

$$\pi = (\prod_{i=1}^{m} v_i^{\pi_{i-1}(i)-i})^{-1}$$

where

$$\begin{aligned}
&v_i \coloneqq (i,m,m-1,\ldots,i+1), \\
&\pi_0 \coloneqq \pi, \\
&\pi_i \coloneqq \pi_{i-1} \cdot v_i^{\pi_{i-1}(i)-i} \quad 1 \leq i.
\end{aligned}$$

The v_i are computed recursively from s_1 and v_1 according to

$$v_i = s_{i-1}v_{i-1}, \text{ where } s_i \coloneqq (i, i+1),$$

with the s_i computed recursively via

$$s_i = v_1 s_{i-1} v_1^{-1}.$$

Hence, we can write every element of G_{SUE} with certain generators.
In G_{SUO} we can use the same algorithm. We only have to ignore the point in the middle.

We also want to know the length and memory quota of an MSLP for this algorithm.

Theorem 4.38: Let $n \in \mathbb{N}$, $\lambda = (\sum_{k=1}^{\lceil \log_2(n) \rceil} 2^k)(4\log_2(n)+7) + 2n\log_2(n) + 4n$ and $b = 11$. There exists a b-MSLP, S, of length at most λ such that if S is evaluated with memory containing $\{s, u, u^{-1}, v\}$ then S outputs memory containing the permutation $\pi \in G_{SUE}$.

Proof. In the worst case we have to split the permutation into a product of 2-cycles. Therefore, we are in case 3 at most $\sum_{k=1}^{\lceil \log_2(n) \rceil} 2^k$ times. For this we use

$$(1, 2, \ldots, m)(m+1, n, n-1, \ldots, m+2)^{n-\imath+1}.$$

Calculating this via repeated squaring needs (see Section 2.2(ii) in [NPP]) at most

$$2\log_2(n)$$

MSLP instructions. We conjugate $(1, n)$ with $(1, 2, \ldots, m)(m+1, n, n-1, \ldots, m+2)^{n-\imath+1}$ and multiply the result from left. This costs 3 MSLP instructions. Therefore, we need at most

$$(\sum_{k=1}^{\lceil \log_2(n) \rceil} 2^k)(2\log_2(n)+3)$$

MSLP instructions. In the worst case all 2-cycles are in case 2. Therefore, we have to perform case 2 at most $\sum_{k=1}^{\lceil \log_2(n) \rceil} 2^k$ times. Analogous to case 3 we calculate

$$(1, n)^{(1,2,\ldots,m)(m+1,n,n-1,\ldots,m+2)^{n-\imath+1}}$$

and this needs at most $2\log_2(n)+2$ MSLP instructions. This time we conjugate our permutation and this needs 2 more instructions. Hence, we need at most

$$(\sum_{k=1}^{\lceil \log_2(n) \rceil} 2^k)(2\log_2(n)+4)$$

MSLP instructions. Finally we use the algorithm from Section 3.3 [NPP]. This needs at most $2n\log_2(n) + 4n$ MSLP instructions, yielding the claimed maximum total length of the MSLP:

$$(\sum_{k=1}^{\lceil \log_2(n) \rceil} 2^k)(4\log_2(n)+7) + 2n\log_2(n) + 4n$$

In addition to storing the input X, the memory quota for the MSLP is as follows. Two memory slots ared needed to store the permutations we multiply from the left and right. Additionally we need 3 memory slots for the algorithm from Section 3.3 [NPP]. Two memory slots are needed for repeated squaring (see Section 2.2(ii) in [NPP]). Thus we need

$$4 + 2 + 3 + 2 = 11$$

memory slots. □

The same proof can be done analogously with G_{SUO}.

Finally, we have to write a diagonal matrix in SU in terms of the LGO standard generators. We observe the following property of diagonal matrices in SU which helps us solve the problem at hand.

Theorem 4.39: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power and $a = \mathrm{diag}(a_1, \ldots, a_n) \in \mathrm{SU}(n, q)$ a diagonal matrix. If n is even, then $a_i = (\overline{a_{n-i+1}})^{-1}$ for $i \in \{1, \ldots, n\}$. Otherwise $a_i = (\overline{a_{n-i+1}})^{-1}$ for $i \in \{1, \ldots, n\} - \{\frac{n+1}{2}\}$ and $a_{\frac{n+1}{2}} \overline{a_{\frac{n+1}{2}}} = 1$.

Proof. We use Corollary 4.7 and know that $a \cdot J_n \cdot a^* = J_n$. We start with even n and have

$$\begin{aligned}
a \cdot J_n \cdot a^* &= a \cdot J_n \cdot \overline{a} \\
&= \mathrm{diag}(a_1, \ldots, a_n) \cdot J_n \cdot \mathrm{diag}(\overline{a_1}, \ldots, \overline{a_n}) \\
&= \begin{pmatrix} 0 & 0 & \ldots & 0 & a_1 \\ 0 & 0 & \ldots & a_2 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n-1} & \ldots & 0 & 0 \\ a_n & 0 & \ldots & 0 & 0 \end{pmatrix} \cdot \mathrm{diag}(\overline{a_1}, \ldots, \overline{a_n}) \\
&= \begin{pmatrix} 0 & 0 & \ldots & 0 & a_1\overline{a_n} \\ 0 & 0 & \ldots & a_2\overline{a_{n-1}} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n-1}\overline{a_2} & \ldots & 0 & 0 \\ a_n\overline{a_1} & 0 & \ldots & 0 & 0 \end{pmatrix} \\
&= J_n.
\end{aligned}$$

Therefore, the statement holds for even n. For odd n we have

$$\begin{aligned}
a \cdot J_n \cdot a^* &= a \cdot J_n \cdot \overline{a} \\
&= \mathrm{diag}(a_1, \ldots, a_n) \cdot J_n \cdot \mathrm{diag}(\overline{a_1}, \ldots, \overline{a_n})
\end{aligned}$$

$$
\begin{aligned}
&= \begin{pmatrix}
0 & 0 & \dots & 0 & \dots & 0 & a_1 \\
0 & 0 & \dots & 0 & \dots & a_2 & 0 \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \dots & a_{\frac{n+1}{2}} & \dots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
0 & a_{n-1} & \dots & 0 & \dots & 0 & 0 \\
a_n & 0 & \dots & 0 & \dots & 0 & 0
\end{pmatrix} \cdot \operatorname{diag}(\overline{a_1}, \dots, \overline{a_n}) \\
&= \begin{pmatrix}
0 & 0 & \dots & 0 & \dots & 0 & a_1\overline{a_n} \\
0 & 0 & \dots & 0 & \dots & a_2\overline{a_{n-1}} & 0 \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \dots & a_{\frac{n+1}{2}}\overline{a_{\frac{n+1}{2}}} & \dots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
0 & a_{n-1}\overline{a_2} & \dots & 0 & \dots & 0 & 0 \\
a_n\overline{a_1} & 0 & \dots & 0 & \dots & 0 & 0
\end{pmatrix} \\
&= J_n.
\end{aligned}
$$

Hence, the claim follows. □

We use some special matrices to write diagonal matrices. For this we also make a case distinction depending on the dimension. We start with even n and define these matrices and show that we can express them in terms of the LGO standard generators.

Definition 4.40: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element. We set

$$h_j^{\mathrm{SUE}} := \operatorname{diag}(\underbrace{1, \dots, 1}_{j-1}, \omega, \omega^{-1}, 1, \dots, 1, \omega^q, \omega^{-q}, \underbrace{1, \dots, 1}_{j-1})$$

for $j \in \{1, \dots, m-1\}$ and

$$h^{\mathrm{SUEC}} := \operatorname{diag}(\underbrace{1, \dots, 1}_{m-1}, \omega^{q+1}, \omega^{-(q+1)}, \underbrace{1, \dots, 1}_{m-1}).$$

The next theorem shows how we can write these matrices in terms of the LGO standard generators which confirms that they lie in $\mathrm{SU}(n, q)$.

Theorem 4.41: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element of $\mathbb{F}_{q^2}$. We have

$$\begin{aligned} h_1^{\mathrm{SUE}} &= y, \\ h_j^{\mathrm{SUE}} &= v^{-1} h_{j-1}^{\mathrm{SUE}} v \qquad \text{for } j \in \{2, \dots, m-1\} \text{ and} \\ h^{\mathrm{SUEC}} &= v^{-(m-1)} \delta v^{m-1}. \end{aligned}$$

Proof. We use φ_{SU} and notice that

$$\varphi_{\mathrm{SU}}(v) = (1, 2, \dots, m)(m+1, n, n-1, \dots, m+2).$$

Clearly $h_1^{\mathrm{SUE}} = y \in \mathrm{SU}(n, q)$ and $v^{-1} h_{j-1}^{\mathrm{SUE}} v$ induces the desired permutation on the diagonal entries.
Moreover, we have $\delta = \mathrm{diag}(\omega^{q+1}, 1, \dots, 1, \omega^{-(q+1)})$ and therefore $h^{\mathrm{SUEC}} = v^{-(m-1)} \delta v^{m-1}$. □

Now we can write every diagonal matrix in $\mathrm{SU}(n, q)$ for even n with the h_j^{SUE} and h^{SUEC}.

Theorem 4.42: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element of $\mathbb{F}_{q^2}$. Let $a = \mathrm{diag}(a_1, \dots, a_n) \in \mathrm{SU}(n, q)$ be a diagonal matrix. Then a is a product of h_j^{SUE} and h^{SUEC}.

Proof. We know by Theorem 4.39 that the matrix a has the form

$$a = \mathrm{diag}(a_1, \dots, a_n) = \mathrm{diag}(a_1, \dots, a_m, (\overline{a_m})^{-1}, \dots, (\overline{a_1})^{-1}).$$

For $j \in \{1, \dots, m-1\}$ we choose $\ell_j \in \{0, \dots, q^2 - 1\}$ such that $\omega^{\ell_j} = a_j$ and therefore $\mathrm{diag}(\omega^{\ell_1}, \dots, \omega^{\ell_{m-1}}) = \mathrm{diag}(a_1, \dots, a_{m-1})$. We set $\sum_{k=1}^{j} \ell_j = \lambda_j$. Then we have

$$\begin{aligned} \prod_{j=1}^{m-1} (h_j^{\mathrm{SUE}})^{\lambda_j} &= (h_1^{\mathrm{SUE}})^{\lambda_1} \dots (h_{m-1}^{\mathrm{SUE}})^{\lambda_{m-1}} \\ &= \mathrm{diag}(\omega, \omega^{-1}, 1, \dots, 1, \omega^q, \omega^{-q})^{\lambda_1} \dots \\ &\quad \dots \mathrm{diag}(1, \dots, 1, \omega, \omega^{-1}, \omega^q, \omega^{-q}, 1, \dots, 1)^{\lambda_{m-1}} \\ &= \mathrm{diag}(\omega^{\lambda_1}, (\omega^{-1})^{\lambda_1}, 1, \dots, 1, (\omega^q)^{\lambda_1}, (\omega^{-q})^{\lambda_1}) \dots \\ &\quad \dots \mathrm{diag}(1, \dots, 1, \omega^{\lambda_{m-1}}, (\omega^{-1})^{\lambda_{m-1}}, (\omega^q)^{\lambda_{m-1}}, (\omega^{-q})^{\lambda_{m-1}}, 1, \dots, 1). \end{aligned}$$

Since each h_j^{SUE} has a 1 in every diagonal entry except the jth, the $(j+1)$th, $(n-j)$th and $(n-j+1)$th, the first entry of the product is $\omega^{\lambda_1} = \omega^{\ell_1} = a_1$ and for $i = 2, \dots, m-1$ the ith entry is the product of the ith entries of h_i^{SUE} and h_{i-1}^{SUE}, namely $\omega^{\lambda_i} \omega^{-\lambda_{i-1}} = \omega^{\ell_i}$. Therefore, we know by Theorem 4.39 that

$$\prod_{j=1}^{m-1}(h_j^{\mathrm{SUE}})^{\lambda_j} = \mathrm{diag}(a_1,\ldots,a_{m-1},\omega^{-\lambda_{m-1}},(\omega^q)^{\lambda_{m-1}},(\overline{a_{m-1}})^{-1},\ldots,(\overline{a_1})^{-1}).$$

Now $\det(a) = a_1a_2\ldots a_m(\overline{a_m})^{-1}\ldots(\overline{a_1})^{-1}$. Then $\det(a)=1$ if and only if

$$\overline{a_m}a_m^{-1} = a_1\ldots a_{m-1}(\overline{a_1\ldots a_{m-1}})^{-1}.$$

Hence, all solutions are given by $\ell_C\cdot\overline{a_1\ldots a_{m-1}}$ for $\ell_C\in\mathbb{F}_0=\mathrm{Fix}_{\mathbb{F}_{q^2}}(\,\overline{\ }\,)$ since

$$\begin{aligned}\overline{a_m}a_m^{-1} &= \overline{\ell_C\overline{a_1\ldots a_{m-1}}}(\ell_C\overline{a_1\ldots a_{m-1}})^{-1}\\ &= \overline{\ell_C}\ell_C^{-1}a_1\ldots a_{m-1}(\overline{a_1\ldots a_{m-1}})^{-1}\\ &= \ell_C\ell_C^{-1}a_1\ldots a_{m-1}(\overline{a_1\ldots a_{m-1}})^{-1} = a_1\ldots a_{m-1}(\overline{a_1\ldots a_{m-1}})^{-1}.\end{aligned}$$

Since ω^{q+1} is a primitive element for $\mathbb{F}_0$ there exists a $\ell_C\in\mathbb{F}_0$ such that

$$\prod_{j=1}^{m-1}(h_j^{\mathrm{SUE}})^{\lambda_j}\cdot(h^{\mathrm{SUEC}})^{\ell_C} = a.$$

□

We calculate the maximum length and the number of memory slots of an MSLP, which computes a diagonal matrix with even dimension from the special unitary group. We proceed analogously to the proof of Proposition 3.4 in [NPP].

Theorem 4.43: Let $n,f,m\in\mathbb{N}$, $q=p^f$ a prime power, $n=2m$ and $\omega\in\mathbb{F}_{q^2}$ a primitive element. Let $\lambda = 3n-6+(n-2)\log_2(q^2)+(n-2)\log_2(m)+2\log_2(q)$ and $b=18$. Let $h\in\mathrm{SU}(n,q)$ be a diagonal matrix given in the form $h=\mathrm{diag}(\omega_1^{\ell},...,\omega_n^{\ell_n})$. There exists a b-MSLP, S, of length at most λ such that if S is evaluated with memory containing the set $X=\{s,s^{-1},t,t^{-1},\delta,\delta^{-1},v,v^{-1},u,u^{-1}x,x^{-1},y,y^{-1}\}$ then S outputs final memory containing h.

Proof. Our MSLP writes h as the product described in Theorem 4.42, with the h_j^{SUE} computed in terms of the standard generators as per Theorem 4.41. Computing the h_j^{SUE} via Theorem 4.41 costs $(\frac{n}{2}-2)2 = n-4$ MSLP instructions: h_1^{SUE} is already the standard generator y and each subsequent h_j^{SUE} is composed by conjugating with v. Computing h^{SUEC} via Theorem 4.41 costs $(\frac{n}{2}-1)2 = n-2$ MSLP instructions. Hence, we need $2n-6$ MSLP instructions.
Powering each h_j^{SUE} up to $(h_j^{\mathrm{SUE}})^{\lambda_j}$ via repeated squaring costs (see Section 2.2(ii) in [NPP]) at most

$$2\log_2(\lambda_j) = 2\log_2(\sum_{k=1}^{j}\ell_k) \le 2\log_2(j(q^2-1)) < 2\log_2(q^2)+2\log_2(j)$$

MSLP instructions, and so computing all of the $(h_j^{\mathrm{SUE}})^{\lambda_j}$ requires at most

$$(\tfrac{n}{2}-1)(2\log_2(q^2)+2\log_2(m)) = (n-2)\log_2(q^2)+(n-2)\log_2(m)$$

instructions. Computing $(h^{\mathrm{SUEC}})^{\ell_C}$ requires at most

$$2\log_2(q-1) < 2\log_2(q)$$

instructions. Moreover, $\frac{n}{2} < n$ instructions are required to multiply the $(h_j^{\mathrm{SUE}})^{\lambda_j}$ and $(h^{\mathrm{SUEC}})^{\ell_C}$ together. Overall, this results in a maximum total length of the MSLP of

$$3n-6+2\log_2(q)+(n-2)\log_2(q^2)+(n-2)\log_2(m).$$

The memory quota for the MSLP is as follows. First we store the input X. At most one memory slot is needed to store the h_j^{SUE} because the recursion for the h_j^{SUE} refers back to h_{j-1}^{SUE} and v, hence, each h_j^{SUE} can overwrite the one from which it is computed. In the end we can store h_j^{SUEC} in this memory slot. Two memory slots are needed for computing the $(h^{\mathrm{SUE}})^{\lambda_j}$ and $(h_j^{\mathrm{SUEC}})^{\ell_C}$ (see Section 2.2(ii) in [NPP]), and one final memory slot is needed to multiply these together. Thus we need

$$14+2+1+1 = 18$$

memory slots. □

Now we assume that n is odd and prove the results analogously.

Definition 4.44: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m+1$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element of $\mathbb{F}_{q^2}$. We set

$$h_j^{\mathrm{SUO}} := \mathrm{diag}(\underbrace{1,\ldots,1}_{j-1},\omega^{-q},1,\ldots,1,\omega,\underbrace{1,\ldots,1}_{j-1}) + (\omega^{q-1}-1)E_{\frac{n+1}{2},\frac{n+1}{2}}$$

for $j \in \{1,\ldots,\frac{n+1}{2}-1\}$.

The next theorem shows how we can express these matrices in terms of LGO standard generators which confirms that they lie in $\mathrm{SU}(n,q)$.

Theorem 4.45: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m+1$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element of $\mathbb{F}_{q^2}$. We have

$$\begin{aligned} h_1^{\mathrm{SUO}} &= y \text{ and} \\ h_j^{\mathrm{SUO}} &= v^{-1}h_{j-1}^{\mathrm{SUO}}v. \end{aligned}$$

Proof. We use φ_{SU} and notice that

$$\varphi_{\mathrm{SU}}(v) = (1, 2, \ldots, \frac{n+1}{2} - 1)(\frac{n+1}{2} + 1, n, n-1, \ldots, \frac{n+1}{2} + 2).$$

Clearly $h_1^{\mathrm{SUO}} = y \in \mathrm{SU}(n,q)$ and $v^{-1} h_{j-1}^{\mathrm{SUO}} v$ induces the desired permutation on the diagonal entries. □

Now we can write every diagonal matrix in $\mathrm{SU}(n,q)$ for n odd with the h_j^{SUO}.

Theorem 4.46: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m+1$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element of $\mathbb{F}_{q^2}$. Let $a = \mathrm{diag}(a_1, \ldots, a_n) \in \mathrm{SU}(n,q)$ a diagonal matrix. Then a is a product of the h_j^{SUO}.

Proof. We know by Theorem 4.39 that the matrix a has the form

$$a = \mathrm{diag}(a_1, \ldots, a_n) = \mathrm{diag}(a_1, \ldots, a_m, a_{m+1}, (\overline{a_m})^{-1}, \ldots, (\overline{a_1})^{-1}).$$

For $j \in \{1, \ldots, m\}$ we choose $k_j \in \{0, \ldots, q^2 - 1\}$ such that $(\omega^{-q})^{k_j} = a_j$. We show that

$$a = \prod_{j=1}^{m} (h_j^{\mathrm{SUO}})^{k_j}.$$

We have

$$\begin{aligned}
\prod_{j=1}^{m} (h_j^{\mathrm{SUO}})^{k_j} =& (h_1^{\mathrm{SUO}})^{k_1} \ldots (h_m^{\mathrm{SUO}})^{k_m} \\
=& (\mathrm{diag}(\omega^{-q}, 1, \ldots, 1, \omega^{q-1}, 1, \ldots, 1, \omega))^{k_1} \ldots \\
& \ldots (\mathrm{diag}(1, \ldots, 1, \omega^{-q}, \omega^{q-1}, \omega, 1, \ldots, 1))^{k_m} \\
=& \,\mathrm{diag}((\omega^{-q})^{k_1}, 1, \ldots, 1, \omega^{k_1}) \ldots \mathrm{diag}(1, \ldots, 1, (\omega^{-q})^{k_m}, 1, \omega^{k_m}, 1, \ldots, 1) \cdot \\
& \mathrm{diag}(1, \ldots, 1, \omega^{q-1}, 1, \ldots, 1)^{k_1 + \ldots + k_m} \\
=& \,\mathrm{diag}(a_1, 1, \ldots, 1, (\overline{a_1})^{-1}) \ldots \mathrm{diag}(1, \ldots, 1, a_m, 1, (\overline{a_m})^{-1}, 1, \ldots, 1) \cdot \\
& \mathrm{diag}(1, \ldots, 1, \omega^{q-1}, 1, \ldots, 1)^{k_1 + \ldots + k_m} \\
=& \,\mathrm{diag}(a_1, \ldots, a_m, 1, (\overline{a_m})^{-1}, \ldots, (\overline{a_1})^{-1}) \cdot \mathrm{diag}(1, \ldots, 1, (\omega^{q-1})^{k_1 + \ldots + k_m}, 1, \ldots, 1).
\end{aligned}$$

Now $\det(a) = a_1 a_2 \ldots a_m a_{m+1} (\overline{a_m})^{-1} \ldots (\overline{a_1})^{-1} = 1$. Since a and $\prod_{j=1}^{m} (h_j^{\mathrm{SUO}})^{k_j}$ have $2m$ identical entries and $\det(\prod_{j=1}^{m} (h_j^{\mathrm{SUO}})^{k_j}) = 1$, it follows that $a_{m+1} = (\omega^{q-1})^{k_1 + \ldots + k_m}$ and therefore $a = \prod_{j=1}^{m} (h_j^{\mathrm{SUO}})^{k_j}$. □

Finally, we calculate the maximum length and the number of memory slots of an MSLP, which computes a diagonal matrix with odd dimension from the special unitary group. Again we proceed analogously to the proof of Proposition 3.4 in [NPP].

Theorem 4.47: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m + 1$ and $\omega \in \mathbb{F}_{q^2}$ a primitive element. Let $\lambda = 2n - 3 + (n-1)\log_2(q^2)$ and $b = 18$. Let $h \in \mathrm{SU}(n,q)$ be a diagonal matrix given in the form $h = \mathrm{diag}(\omega_1^{\ell}, ..., \omega_n^{\ell_n})$. There exists a b-MSLP, S, of length at most λ such that if S is evaluated with memory containing the set $X = \{s, s^{-1}, t, t^{-1}, \delta, \delta^{-1}, v, v^{-1}, u, u^{-1}x, x^{-1}, y, y^{-1}\}$ then S outputs final memory containing h.

Proof. Our MSLP writes h as the product described in Theorem 4.46, with the h_j^{SUO} computed in terms of the standard generators as per Theorem 4.45. Computing the h_j^{SUO} via Theorem 4.45 costs $(\frac{n-1}{2} - 1)2 = n - 3$ MSLP instructions: h_1^{SUO} is already the standard generator y and each subsequent h_j^{SUO} is composed by conjugating with v.
Powering each h_j^{SUO} up to $(h_j^{\mathrm{SUO}})^{k_j}$ via repeated squaring costs (see Section 2.2(ii) in [NPP]) at most

$$2\log_2(k_j) \leq 2\log_2(q^2)$$

MSLP instructions, and so computing all of the $(h_j^{\mathrm{SUO}})^{k_j}$ requires at most

$$\tfrac{n-1}{2}(2\log_2(q^2)) = (n-1)\log_2(q^2)$$

instructions. Moreover, $\frac{n-1}{2} < n$ instructions are required to multiply the $(h_j^{\mathrm{SUO}})^{k_j}$ together. Overall, this results in a maximum total length of the MSLP of

$$2n - 3 + (n-1)\log_2(q^2).$$

The memory quota for the MSLP is as follows. First we store the input X. At most one memory slot is needed to store the h_j^{SUO} because the recursion for the h_j^{SUO} refers back to h_{j-1}^{SUO} and v, hence, each h_j^{SUO} can overwrite the one from which it is computed. Two memory slots are needed for computing the $(h_j^{\mathrm{SUO}})^{k_j}$ (see Section 2.2(ii) in [NPP]), and one final memory slot is needed to multiply these together. Thus we need

$$14 + 2 + 1 + 1 = 18$$

memory slots. □

5 Bruhat Decomposition in Sp

This chapter deals with the symplectic group.
The structure is analogous to the structure of the chapter about the special unitary group. Since the symplectic groups only exist for vector spaces with even dimensions, we do not need a case distinction this time.

5.1 Mathematical Background for the Bruhat Decomposition in Sp

We start with some mathematical background for this group, which we need in the rest of the chapter.

First we show that Sp is generated by symplectic transvections. The proof is from Yaim Cooper of May 11 2005 and available on the website of the Massachusetts Institute of Technology. [SPGT]

Theorem 5.1: Let V be an $n = 2m$-dimensional $\mathbb{F}$-vector space with a symplectic bilinear form Φ. Then $\mathrm{Sp}(V)$ is generated by symplectic transvections.

Proof. Denote by T the group generated by all symplectic transvections of (V, Φ). We show that $T = \mathrm{Sp}(v)$. This is done in 3 steps.
Step(1): T acts transitively on $V - \{0\}$.
Let $\nu, w \in V - \{0\}$.
Case(1.1): $\Phi(\nu, w) \neq 0$.
We set $\imath = \frac{1}{\Phi(\nu,w)}$ and $u = \nu - w$. Then $\tau_{u,\imath}(x) = x + \imath\Phi(x, u)u$ is a symplectic transvection [Taylor, page 71] and we have

$$\tau_{u,\imath}(\nu) = \nu + \imath\Phi(\nu, u)u = \nu + \tfrac{\Phi(\nu,\nu-w)}{\Phi(\nu,w)}(\nu - w) = \nu - (\nu - w) = w.$$

Case(1.2): $\Phi(\nu, w) = 0$.
We want a vector z such that $\Phi(\nu, z) \neq 0 \neq \Phi(w, z)$. We can find a symplectic basis $(e_1, f_1, \ldots, e_m, f_m)$ [Taylor, page 69] and, therefore, we can write

$$\nu = \sum_{j=1}^{m} (\imath_j ej + \jmath_j f_j) \text{ and}$$
$$w = \sum_{j=1}^{m} (\imath'_j ej + \jmath'_j f_j).$$

If for some $1 \leq i \leq m$, $\imath_i \neq 0$ or $\jmath_i \neq 0$ and $\imath'_i \neq 0$ or $\jmath'_i \neq 0$, then some linear combinations of e_i and f_i satisfies our requirement for z. Otherwise since neither ν nor w is 0, there is a non-zero coefficient $\imath_{\nu,i}$ of ν and non-zero coefficient $\imath_{w,j}$ of w and $i \neq j$. Then we can take z

to be an appropriate non-zero linear combination of e_i and f_i plus an appropriate non-zero linear combination of e_j and f_j.
Therefore, the desired z exists. We can now use Case 1.1 to construct τ_1 and τ_2 such that $\tau_1(\nu) = z$, $\tau_2(z) = w$ and thus $\tau_2\tau_1(\nu) = w$.
Step(2): T is transitive on hyperbolic pairs.
We want to find a product of transvections which maps (u_1, ν_1) to (u_2, ν_2) where (u_1, ν_1) and (u_2, ν_2) are hyperbolic pairs. By step(1), there is a transvection τ mapping u_1 to u_2. Thus $\tau((u_1, \nu_1)) = (u_2, \tau(\nu_1))$. We define $\nu_3 = \tau(\nu_1)$. We want $\rho \in T$ such that $\rho(u_2) = u_2$ and $\rho(\nu_3) = \nu_2$.
Case(2.1): $\Phi(\nu_3, \nu_2) \neq 0$.
We set $\nu_4 = \nu_3 - \nu_2$ and $\imath = \frac{1}{\Phi(\nu_3,\nu_2)}$. We have

$$\begin{aligned}\Phi(u_2, \nu_4) &= \Phi(u_2, \nu_3 - \nu_2)\\ &= \Phi(u_2, \nu_3) - \Phi(u_2, \nu_2)\\ &= \Phi(\tau(u_1), \tau(\nu_1)) - \Phi(u_2, \nu_2)\\ &= \Phi(u_1, \nu_1) - \Phi(u_2, \nu_2) = 1 - 1 = 0\end{aligned}$$

and therefore

$$\tau_{\nu_4,\imath}(u_2) = u_2 + \imath\Phi(u_2, \nu_4)\nu_4 = u_2 + 0 = u_2 \text{ and}$$
$$\tau_{\nu_4,\imath}(\nu_3) = \nu_3 + \imath\Phi(\nu_3, \nu_4)\nu_4 = \nu_3 + \frac{\Phi(\nu_3, \nu_3 - \nu_2)}{\Phi(\nu_3, \nu_2)}(\nu_3 - \nu_2) = \nu_3 - (\nu_3 - \nu_2) = \nu_2.$$

Case(2.2): $\Phi(\nu_3, \nu_2) = 0$.
We have $\Phi(\nu_3, u_2 + \nu_3) = -1$, $\Phi(u_2 + \nu_3, \nu_2) = \Phi(u_2, \nu_2) = 1$ and $\Phi(u_2, -u_2) = 0 = \Phi(u_2, u_2) + \Phi(u_2, \nu_3) - \Phi(u_2, \nu_2) = \Phi(u_2, u_2 + \nu_3 - \nu_2)$. Therefore

$$\begin{aligned}&\tau_{-u_2,-1}(u_2) = u_2 + (-1)\Phi(u_2, -u_2)(-u_2) = u_2 + 0u_2 = u_2,\\ &\tau_{-u_2,-1}(\nu_3) = \nu_3 + (-1)\Phi(\nu_3, -u_2)(-u_2) = \nu_3 + u_2,\\ &\tau_{u_2+\nu_3-\nu_2,1}(u_2) = u_2 + \Phi(u_2, u_2 + \nu_3 - \nu_2)(u_2 + \nu_3 - \nu_2) = u_2 + 0(u_2 + \nu_3 - \nu_2) = u_2,\\ &\tau_{u_2+\nu_3-\nu_2,1}(u_2 + \nu_3) = u_2 + \nu_3 + \Phi(u_2 + \nu_3, u_2 + \nu_3 - \nu_2)(u_2 + \nu_3 - \nu_2) = \nu_2.\end{aligned}$$

Thus $\tau_{u_2+\nu_3-\nu_2,1}\tau_{-u_2,-1}\tau \in T$ and $\tau_{u_2+\nu_3-\nu_2,1}\tau_{-u_2,-1}\tau(u_1, \nu_1) = (u_2, \nu_2)$.
Step(3): The symplectic group $\mathrm{Sp}(V)$ is generated by symplectic transvections.
We prove the result by induction on m. Base case: If $m = 1$ then $\mathrm{Sp}(V) = \mathrm{SL}(V)$ and $\mathrm{SL}(V)$ is generated by transvections (Lemma 3.2).
Induction step: Chose a hyperbolic pair (u, ν) in V and let $W = \langle u, \nu\rangle$. Then $V = W \oplus W^\perp$. Let $\sigma \in \mathrm{Sp}(V)$. Then $(\sigma(u), \sigma(\nu))$ is a also a hyperbolic pair and by step(2) there exist $\tau \in T$ such that $\tau\sigma(u) = u$ and $\tau\sigma(\nu) = \nu$, so $\tau\sigma|_W = \mathrm{Id}_W$. Moreover $\tau\sigma|_{W^\perp} \in \mathrm{Sp}(W^\perp)$. By induction, $\tau\sigma|_{W^\perp}$ is a product of symplectic transvections on $W^\perp$. Since any transvection on $W^\perp$ can be extended to a transvection V which includes W in the in the fixed hyperplane, we see that in V, $\tau\sigma \in T$ and, hence, $\sigma \in T$. □

To find a Bruhat decomposition for Sp we use the same strategy that we used for SU. We start with a monomorphism from $\mathrm{GL}(n,q)$ into $\mathrm{Sp}(2n,q)$. This gives us a good idea how components of the decomposition could look like.

Definition 5.2: Let $n, f \in \mathbb{N}$ and $q = p^f$ a prime power. We define

$$P_n \coloneqq \begin{pmatrix} 0 & J_n \\ -J_n & 0 \end{pmatrix} \in \mathbb{F}_q^{2n\times 2n}.$$

Example 5.3: For $n = 2$ we have

$$P_2 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix}.$$

Remark 5.4: Observe that $P_n^4 = I_{2n}$.

The next result shows that we may assume that $\mathrm{Sp}(n,q)$ preserves a form with Gram-matrix $P_{\frac{n}{2}}$. We need this for later proofs .

Theorem 5.5: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power and P_n as in Definition 5.2.
Then $\mathrm{Sp}(2n,q) = \{a \in \mathrm{GL}(2n,q) \mid aP_na^{\mathrm{T}} = P_n\}$.

Proof. We can find a symplectic basis $\{e_1, f_1, \ldots, e_n, f_n\}$ [Taylor, page 69]. We can reorder this basis into $(e_1, e_2, \ldots, f_2, f_1)$ such that the corresponding Gram-matrix has the form P_n. Therefore, $a \in \mathrm{Sp}(2n,q)$ if and only if $aP_na^{\mathrm{T}} = P_n$. □

Remark 5.6: We see that

$$P^{\mathrm{T}} = P^{-1} = \begin{pmatrix} 0 & -J_n \\ J_n & 0 \end{pmatrix}.$$

Moreover we notice that $aP_na^{\mathrm{T}} = P_n$ if and only if $aP_n^{\mathrm{T}}a^{\mathrm{T}} = P_n^{\mathrm{T}}$. Therefore
$\mathrm{Sp}(2n,q) = \{a \in \mathrm{GL}(2n,q) \mid aP_na^{\mathrm{T}} = P_n\} = \{a \in \mathrm{GL}(2n,q) \mid aP_n^{\mathrm{T}}a^{\mathrm{T}} = P_n^{\mathrm{T}}\}$.

Now we can define a monomorphism from $\mathrm{GL}(n,q)$ into $\mathrm{Sp}(2n,q)$.

Lemma 5.7: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power, $a \in \mathrm{GL}(n,q)$ and define $a_d = \mathrm{Diag}(a, J_n a^{-\mathrm{T}} J_n) \in \mathrm{Sp}(2n, \mathbb{F}_q)$. Then

$$\theta : \mathrm{GL}(n,q) \hookrightarrow \mathrm{Sp}(2n,q) : a \mapsto a_d$$

is a monomorphism.

Proof. First we notice that $a_d^{\mathrm{T}} = \mathrm{Diag}(a^{\mathrm{T}}, J_n a^{-1} J_n)$ since J_n is symmetric. We have

$$\begin{aligned} a_d P_n a_d^{\mathrm{T}} &= \begin{pmatrix} a & 0 \\ 0 & J_n a^{-\mathrm{T}} J_n \end{pmatrix} \cdot \begin{pmatrix} 0 & J_n \\ -J_n & 0 \end{pmatrix} \cdot \begin{pmatrix} a^{\mathrm{T}} & 0 \\ 0 & J_n a^{-1} J_n \end{pmatrix} \\ &= \begin{pmatrix} 0 & aJ_n \\ J_n a^{-\mathrm{T}}(-J_n J_n) & 0 \end{pmatrix} \cdot \begin{pmatrix} a^{\mathrm{T}} & 0 \\ 0 & J_n a^{-1} J_n \end{pmatrix} \\ &= \begin{pmatrix} 0 & aJ_n J_n a^{-1} J_n \\ -J_n a^{-\mathrm{T}} a^{\mathrm{T}} & 0 \end{pmatrix} \\ &= \begin{pmatrix} 0 & J_n \\ -J_n & 0 \end{pmatrix} = P_n. \end{aligned}$$

By Theorem 5.5 it follows that $a_d \in \mathrm{Sp}(2n,q)$.
The block in the top left corner shows that θ is injective. It remains to show that θ is a group homomorphism. Let $a, b \in \mathrm{GL}(n,q)$. We have

$$\begin{aligned} \theta(a)\theta(b) &= \mathrm{Diag}(a, J_n a^{-\mathrm{T}} J_n)\,\mathrm{Diag}(b, J_n b^{-\mathrm{T}} J_n) \\ &= \mathrm{Diag}(ab, J_n a^{-\mathrm{T}} b^{-\mathrm{T}} J_n) \\ &= \mathrm{Diag}(ab, J_n (ab)^{-\mathrm{T}} J_n) \\ &= \theta(ab). \end{aligned}$$

Hence, the claim follows. □

For the remainder of the chapter, we keep the notation from the previous lemma.

5.2 Transvections in Sp

The focus of this chapter is on symplectic transvections. We describe transvections and prove that they lie in the special unitary group. First we investigate the images of the matrices $I_{i,j}(\imath)$ from Lemma 3.2.

Lemma 5.8: Let $I_{i,j}(\imath) \in \mathrm{GL}(n,q)$ for $\imath \in \mathbb{F}_q$, $i,j \in \{1,\dots,n\}$ and $j < i$ be a lower-unitriangular matrix. Then $\theta(I_{i,j}(\imath))$ is a lower-unitriangular matrix.

Proof. It is enough to check that $J_n \cdot (I_{i,j}(\imath))^{-\mathrm{T}} \cdot J_n$ is a lower-unitriangular matrix. We see that $(I_{i,j}(\imath))^{-\mathrm{T}} = I_{j,i}(\imath^{-1})$ is an upper-unitriangular matrix and $J_n \cdot I_{j,i}(\imath^{-1}) \cdot J_n = I_{n+1-j,n+1-i}(\imath^{-1})$. Now $j < i$ if and only if $n+1-j > n+1-i$. Hence, $J_n \cdot (I_{i,j}(\imath))^{-\mathrm{T}} \cdot J_n$ is a lower-unitriangular matrix. □

By calculating a Bruhat decomposition we can use these matrices to eliminate entries where the entry in the transvection is in the upper left or lower right block. The following example shows how matrices that can eliminate entries where the corresponding entry in a transvection is in the lower left block can look.

Example 5.9: Let $f \in \mathbb{N}$ and $q = p^f$ be a prime power. We look at matrices with a field element in the lower left block. For example we could define

$$a = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ a_{3,1} & 0 & 1 & 0 \\ 0 & a_{4,2} & 0 & 1 \end{pmatrix} \in \mathbb{F}_q^{4\times 4}.$$

We notice that

$$a \cdot P_2 \cdot a^{\mathrm{T}} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & a_{3,1}-a_{4,2} \\ -1 & 0 & a_{4,2}-a_{3,1} & 0 \end{pmatrix}.$$

So $a \in \mathrm{Sp}(4,q)$ if and only if $a_{3,1} = a_{4,2}$.

We prove this in general.

Theorem 5.10: Let $n, m, f \in \mathbb{N}$, n even with $n = 2m$ and $q = p^f$ a prime power. Define

$$S_{i,j}(\imath) \coloneqq I_n + E_{i,j}(\imath) + E_{n-j+1,n-i+1}(\imath)$$

for $i \in \{\frac{n}{2}+1,\dots,n\}$, $j \in \{1,\dots,\frac{n}{2}\}$, $j < i$, $i+j \neq n+1$ and $\imath \in \mathbb{F}_q$. Then $S_{i,j}(\imath) \in \mathrm{Sp}(n,q)$.

Proof. We use Theorem 5.5 and check that $S_{i,j}(\imath) \cdot P_m \cdot S_{i,j}(\imath)^{\mathrm{T}} = P_m$. We have

$$\begin{aligned} S_{i,j}(\imath) \cdot P_m \cdot S_{i,j}(\imath)^{\mathrm{T}} &= (I_n + E_{i,j}(\imath) + E_{n-j+1,n-i+1}(\imath)) \cdot P_m \cdot S_{i,j}(\imath)^{\mathrm{T}} \\ &= (P_m + E_{i,n-j+1}(\imath) + E_{n-j+1,i}(\imath)) \cdot S_{i,j}(\imath)^{\mathrm{T}} \\ &= (P_m + E_{i,n-j+1}(\imath) + E_{n-j+1,i}(\imath)) \cdot (I_n + E_{j,i}(\imath) + E_{n-i+1,n-j+1}(\imath)) \\ &= P_m + E_{i,n-j+1}(\imath) + E_{n-j+1,i}(\imath) - E_{n-j+1,i}(\imath) - E_{i,n-j+1}(\imath) \\ &= P_m. \end{aligned}$$

Hence, $S_{i,j}(\imath) \in \mathrm{Sp}(n,q)$. □

We also need transvections $S_{i,j}(\imath)$ where $i + j = n + 1$. The next theorem shows that we can use all these matrices.

Theorem 5.11: Let $n, m, f \in \mathbb{N}$, n even with $n = 2m$, $q = p^f$ a prime power and $i \in \{\frac{n}{2} + 1, \ldots, n\}$. Then $S_{i,n-i+1}(\imath) \in \mathrm{Sp}(n,q)$ for all $\imath \in \mathbb{F}_q$.

Proof. We set $j = n - i + 1$. We use Theorem 5.5 and check that $S_{i,j}(\imath) \cdot P_m \cdot S_{i,j}(\imath)^{\mathrm{T}} = P_m$. We have

$$\begin{aligned} S_{i,j}(\imath) \cdot P_m \cdot S_{i,j}(\imath)^{\mathrm{T}} &= (I_n + E_{i,j}(\imath)) \cdot P_m \cdot S_{i,j}(\imath)^{\mathrm{T}} \\ &= (P_m + E_{i,i}(\imath)) \cdot S_{i,j}(\imath)^{\mathrm{T}} \\ &= (P_m + E_{i,i}(\imath)) \cdot (I_n + E_{j,i}(\imath)) \\ &= P_m + E_{i,i}(\imath) - E_{n-j+1,i}(\imath) \\ &= P_m + E_{i,i}(\imath) - E_{i,i}(\imath) \\ &= P_m. \end{aligned}$$

Hence, $S_{i,j}(\imath) \in \mathrm{Sp}(n,q)$. □

We use the following notation to work with transvections in the symplectic group.

Definition 5.12: Let $n, f \in \mathbb{N}$, $q = p^f$ a prime power and n even. For $\imath \in \mathbb{F}_q - \{0\}$, $i, j \in \{1, \ldots, n\}$ and $j < i$ we set

$$S_{i,j}(\imath) \coloneqq \begin{cases} I_n + E_{i,j}(\imath), & \text{if } i + j = n + 1, \\ I_n + E_{i,j}(\imath) - E_{n-j+1,n-i+1}(\imath), & \text{if } i \in \{2, \ldots, \frac{n}{2}\} \text{ or } j \in \{\frac{n}{2} + 1, \ldots, n - 1\}, \\ I_n + E_{i,j}(\imath) + E_{n-j+1,n-i+1}(\imath), & \text{otherwise.} \end{cases}$$

5.3 Algorithm for the Bruhat Decomposition in Sp

Now we have everything to describe an algorithm to compute a Bruhat decomposition for the symplectic group.

The idea is basically the same as for the unitary group with even dimension except that we use the matrices $S_{i,j}(\imath)$ in this case.
The difference is that we have to make a case distinction where the position of the entry to eliminate is. This corresponds to Example 5.9 and Definition 5.12.

We give the algorithm UnitriangularDecompositionSp in pseudo code.
Let $a \in \mathrm{Sp}(n, q)$.

Algorithm 5: UnitriangularDecompositionSp

Input: $a \in \mathrm{Sp}(n, q)$.
Output: A monomial matrix $w \in \mathrm{Sp}(n, q)$ and two lower unitriangular matrices $u_1, u_2 \in \mathrm{Sp}(n, q)$ such that $w = u_1 \cdot a \cdot u_2$.

```
1  function UnitriangularDecompositionSp(a)
2      u_1 := I_n, u_2 := I_n
3      for c ∈ [n, ..., n/2] do
4          Search first entry i ∈ [1, ..., n] with a_{i,c} ≠ 0
5          Set r := i and piv := a_{r,c}
6          for i ∈ [r + 1, ..., n] do
7              α := -a_{i,c} / a_{r,c}
8              if r + i ≠ n + 1 then
9                  if r < n/2 then
10                     if i < n/2 then
11                         a_{i,-} := a_{i,-} + α a_{r,-}; u_{1i,-} := u_{1i,-} + α u_{1r,-}
12                         a_{n-r+1,-} := a_{n-r+1,-} - α a_{n-i+1,-};
13                         u_{1n-r+1,-} := u_{1n-r+1,-} - α u_{1n-i+1,-}
14                     else
15                         a_{i,-} := a_{i,-} + α a_{r,-}; u_{1i,-} := u_{1i,-} + α u_{1r,-}
16                         a_{n-r+1,-} := a_{n-r+1,-} + α a_{n-i+1,-};
17                         u_{1n-r+1,-} := u_{1n-r+1,-} + α u_{1n-i+1,-}
18                 else
19                     a_{i,-} := a_{i,-} + α a_{r,-}; u_{1i,-} := u_{1i,-} + α u_{1r,-}
20                     a_{n-r+1,-} := a_{n-r+1,-} - α a_{n-i+1,-}; u_{1n-r+1,-} := u_{1n-r+1,-} - α u_{1n-i+1,-}
21             else
22                 a_{i,-} := a_{i,-} + α a_{r,-}; u_{1i,-} := u_{1i,-} + α u_{1r,-}
```

21 **for** $j \in [c-1, ..., 1]$ **do**
22 $\alpha := -\frac{a_{r,j}}{a_{r,c}}$;
23 **if** $c + j \neq n + 1$ **then**
24 **if** $j > \frac{n}{2}$ **then**
25 $a_{-,j} := a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} := u_{2-,j} + \alpha u_{2-,c}$;
26 $a_{-,n-c+1} := a_{-,n-c+1} - \alpha a_{-,n-j+1}$; $u_{2-,n-c+1} := u_{2-,n-c+1} - \alpha u_{2-,n-j+1}$;
27 **else**
28 $a_{-,j} := a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} := u_{2-,j} + \alpha u_{2-,c}$;
29 $a_{-,n-c+1} := a_{-,n-c+1} + \alpha a_{-,n-j+1}$; $u_{2-,n-c+1} := u_{2-,n-c+1} + \alpha u_{2-,n-j+1}$;
30 **else**
31 $a_{-,j} := a_{-,j} + \alpha a_{-,c}$; $u_{2-,j} := u_{2-,j} + \alpha u_{2-,c}$;
32 **return** $[a, u_1, u_2]$

We also give an example to see how the algorithm works.

Example 5.13: We demonstrate how Algorithm 5 works using the example of

$$a := \begin{pmatrix} 4 & 2 & 3 & 4 \\ 1 & 6 & 1 & 5 \\ 1 & 2 & 1 & 2 \\ 6 & 4 & 1 & 0 \end{pmatrix} \in \mathrm{Sp}(4, 7).$$

The algorithm first considers the 4th column. We pick the entry $a_{1,4} = 4$ and clear everything below this element in column 4.

$$\begin{pmatrix} 4 & 2 & 3 & 4 \\ 1 & 6 & 1 & 5 \\ 1 & 2 & 1 & 2 \\ 6 & 4 & 1 & 0 \end{pmatrix} \xrightarrow{\cdot^L S_{2,1}(4)} \begin{pmatrix} 4 & 2 & 3 & 4 \\ 3 & 0 & 6 & 0 \\ 1 & 2 & 1 & 2 \\ 2 & 3 & 4 & 6 \end{pmatrix} \xrightarrow{\cdot^L S_{3,1}(3)} \begin{pmatrix} 4 & 2 & 3 & 4 \\ 3 & 0 & 6 & 0 \\ 6 & 1 & 3 & 0 \\ 4 & 3 & 1 & 6 \end{pmatrix}.$$

Now we clear the entry $a_{4,4} = 6$ and column 4 is finished. We observe that as soon as we have ensured that column 4 has only one non-zero entry then row 4 also has only one non-zero element.

$$\begin{pmatrix} 4 & 2 & 3 & 4 \\ 3 & 0 & 6 & 0 \\ 6 & 1 & 3 & 0 \\ 4 & 3 & 1 & 6 \end{pmatrix} \xrightarrow{\cdot^L S_{4,1}(2)} \begin{pmatrix} 4 & 2 & 3 & 4 \\ 3 & 0 & 6 & 0 \\ 6 & 1 & 3 & 0 \\ 5 & 0 & 0 & 0 \end{pmatrix}.$$

We continue the algorithm and clear row 1 with $a_{1,4} = 4$.

$$\begin{pmatrix} 4 & 2 & 3 & 4 \\ 3 & 0 & 6 & 0 \\ 6 & 1 & 3 & 0 \\ 5 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^R S_{4,3}(1)} \begin{pmatrix} 2 & 2 & 0 & 4 \\ 3 & 0 & 6 & 0 \\ 5 & 1 & 3 & 0 \\ 5 & 0 & 0 & 0 \end{pmatrix}.$$

Again we observe that as soon as we have ensured that row 1 has only one non-zero entry then column 1 also has only one non-zero element.

$$\begin{pmatrix} 2 & 2 & 0 & 4 \\ 3 & 0 & 6 & 0 \\ 5 & 1 & 3 & 0 \\ 5 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^R S_{4,2}(3)} \begin{pmatrix} 2 & 0 & 0 & 4 \\ 0 & 0 & 6 & 0 \\ 0 & 1 & 3 & 0 \\ 5 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^R S_{4,1}(3)} \begin{pmatrix} 0 & 0 & 0 & 4 \\ 0 & 0 & 6 & 0 \\ 0 & 1 & 3 & 0 \\ 5 & 0 & 0 & 0 \end{pmatrix}.$$

We prove this result in the course of the chapter.
Now we can continue with the next column. We select $a_{2,3} = 6$ and clear column 3 and row 2.

$$\begin{pmatrix} 0 & 0 & 0 & 4 \\ 0 & 0 & 6 & 0 \\ 0 & 1 & 3 & 0 \\ 5 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\cdot^L S_{3,2}(3)} \begin{pmatrix} 0 & 0 & 0 & 4 \\ 0 & 0 & 6 & 0 \\ 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 \end{pmatrix} := h.$$

Therefore

$$h = S_{3,2}(3)S_{4,1}(2)S_{3,1}(3)S_{2,1}(4)aS_{4,3}(1)S_{4,2}(3)S_{4,1}(3)$$

Now we have the Bruhat decomposition of a.

The next theorem shows that if a matrix $a \in \mathrm{Sp}(n, q)$ has exactly one non-zero entry in row k, say in position (k, j), then it also has exactly one non-zero element in the column $n - j + 1$.

Lemma 5.14: Let $n, f, m \in \mathbb{N}$, $n = 2m$, $j \in \{m+1, \ldots, n\}$, $q = p^f$ a prime power and $a \in \mathrm{Sp}(n, q)$ such that there exists a $k \in \{1, \ldots, n\}$ with $a_{k,j} \neq 0$ and $a_{k,i} = 0$ for all $i \in \{1, \ldots, n\} - \{j\}$. Then

$$a_{i,n-j+1} = \begin{cases} a_{k,j}^{-1}, & \text{if } i = n-k+1 \text{ and } k > m, \\ -a_{k,j}^{-1}, & \text{if } i = n-k+1 \text{ and } k \leq m, \\ 0, & \text{otherwise.} \end{cases}$$

Proof. Since $a \in \mathrm{Sp}(n,q)$ we know by Theorem 5.5 that $a \cdot P_m \cdot a^{\mathrm{T}} = P_m$. We set $t = n-j+1$. Hence

$$
\begin{aligned}
& a \cdot P_m \cdot a^{\mathrm{T}} \\
&= \begin{pmatrix}
* & \dots & * & a_{1,t} & * & \dots & * & * & * & \dots & * \\
* & \dots & * & a_{2,t} & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & a_{i-1,t} & * & \dots & * & * & * & \dots & * \\
0 & \dots & 0 & 0 & 0 & \dots & 0 & a_{k,j} & 0 & \dots & 0 \\
* & \dots & * & a_{i+1,t} & * & \dots & * & * & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & a_{n,t} & * & \dots & * & * & * & \dots & *
\end{pmatrix} \cdot P_m \cdot a^{\mathrm{T}} \\
&= \begin{pmatrix}
* & \dots & * & * & * & \dots & * & a_{1,t} & * & \dots & * \\
* & \dots & * & * & * & \dots & * & a_{2,t} & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & a_{i-1,t} & * & \dots & * \\
0 & \dots & 0 & -a_{k,j} & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\
* & \dots & * & * & * & \dots & * & a_{i+1,t} & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & * & * & \dots & * & a_{n,t} & * & \dots & *
\end{pmatrix} \cdot \begin{pmatrix}
* & \dots & * & 0 & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & 0 & * & \dots & * \\
a_{1,t} & \dots & a_{k-1,t} & 0 & a_{k+1,t} & \dots & a_{n,t} \\
* & \dots & * & 0 & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & 0 & * & \dots & * \\
* & \dots & * & a_{k,j} & * & \dots & * \\
* & \dots & * & 0 & * & \dots & * \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
* & \dots & * & 0 & * & \dots & *
\end{pmatrix} \\
&= \begin{pmatrix}
0 & 0 & 0 & \dots & 0 & 1 \\
0 & 0 & 0 & \dots & 1 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & 0 & 0 \\
-a_{k,j}a_{1,t} & -a_{k,j}a_{2,t} & -a_{k,j}a_{3,t} & \dots & -a_{k,j}a_{n-1,t} & -a_{k,j}a_{n,t} \\
0 & 0 & 0 & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 1 & 0 & \dots & 0 & 0 \\
1 & 0 & 0 & \dots & 0 & 0
\end{pmatrix} \\
&= P_m.
\end{aligned}
$$

Therefore, the claim follows. □

We also have to consider the case that $j \in \{1, \dots, m\}$ since we want to show the same result for a matrix with one non-zero entry in a column and we transpose the matrix in the proof.

Lemma 5.15: Let $n, f, m \in \mathbb{N}$, $n = 2m$, $j \in \{1, \ldots, m\}$, $q = p^f$ a prime power and $a \in \mathrm{Sp}(n, q)$ such that there exists a $k \in \{1, \ldots, n\}$ with $a_{k,j} \neq 0$ and $a_{k,i} = 0$ for all $i \in \{1, \ldots, n\} - \{j\}$. Then

$$a_{i,n-j+1} = \begin{cases} a_{k,j}^{-1}, & \text{if } i = n-k+1 \text{ and } k \leq m, \\ -a_{k,j}^{-1}, & \text{if } i = n-k+1 \text{ and } k > m, \\ 0, & \text{else.} \end{cases}$$

Proof. Since $a \in \mathrm{Sp}(n, q)$ we know by Theorem 5.5 that $a \cdot P_m \cdot a^{\mathrm{T}} = P_m$. We set $t = n-j+1$. Hence

$$\begin{aligned} & a \cdot P_m \cdot a^{\mathrm{T}} \\ &= \begin{pmatrix} * & \ldots & * & * & * & \ldots & * & a_{1,t} & * & \ldots & * \\ * & \ldots & * & * & * & \ldots & * & a_{2,t} & * & \ldots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \ldots & * & * & * & \ldots & * & a_{i-1,t} & * & \ldots & * \\ 0 & \ldots & 0 & a_{k,j} & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 \\ * & \ldots & * & * & * & \ldots & * & a_{i+1,t} & * & \ldots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \ldots & * & * & * & \ldots & * & a_{n,t} & * & \ldots & * \end{pmatrix} \cdot P_m \cdot a^{\mathrm{T}} \\ &= \begin{pmatrix} * & \ldots & * & -a_{1,t} & * & \ldots & * & * & * & \ldots & * \\ * & \ldots & * & -a_{2,t} & * & \ldots & * & * & * & \ldots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \ldots & * & -a_{i-1,t} & * & \ldots & * & * & * & \ldots & * \\ 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & a_{k,j} & 0 & \ldots & 0 \\ * & \ldots & * & -a_{i+1,t} & * & \ldots & * & * & * & \ldots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \ldots & * & -a_{n,t} & * & \ldots & * & * & * & \ldots & * \end{pmatrix} \cdot \begin{pmatrix} * & \ldots & * & 0 & * & \ldots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \ldots & * & 0 & * & \ldots & * \\ * & \ldots & * & a_{k,j} & * & \ldots & * \\ * & \ldots & * & 0 & * & \ldots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \ldots & * & 0 & * & \ldots & * \\ a_{1,t} & \ldots & a_{k-1,t} & 0 & a_{k+1,t} & \ldots & a_{n,t} \\ * & \ldots & * & 0 & * & \ldots & * \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & \ldots & * & 0 & * & \ldots & * \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0 & 0 & \ldots & 0 & 1 \\ 0 & 0 & 0 & \ldots & 1 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 & 0 \\ a_{k,j}a_{1,t} & a_{k,j}a_{2,t} & a_{k,j}a_{3,t} & \ldots & a_{k,j}a_{n-1,t} & a_{k,j}a_{n,t} \\ 0 & 0 & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & 0 & \ldots & 0 & 0 \\ 1 & 0 & 0 & \ldots & 0 & 0 \end{pmatrix} = P_m. \end{aligned}$$

Therefore the claim follows. □

We can now show the same result also for a column with exactly one non-zero entry.

Lemma 5.16: Let $n, f \in \mathbb{N}$, $j \in \{1, \dots, n\}$, $q = p^f$ a prime power and $a \in \mathrm{Sp}(n, q)$ such that there exists a $k \in \{1, \dots, m\}$ with $a_{k,j} \neq 0$ and $a_{i,j} = 0$ for all $i \in \{1, \dots, n\} - \{k\}$. Then

$$a_{n-k+1,i} = \begin{cases} a_{k,j}^{-1}, & \text{if } i = n - j + 1 \text{ and } j \leq m, \\ -a_{k,j}^{-1}, & \text{if } i = n - j + 1 \text{ and } j > m, \\ 0, & \text{else.} \end{cases}$$

Proof. We know by Theorem 5.5 that $aP_n a^{\mathrm{T}} = P_n$. Since $P_n^{-1} = \begin{pmatrix} 0 & -J_n \\ J_n & 0 \end{pmatrix}$ we see that

$$(aP_n a^{\mathrm{T}})^{-1} = (P_n)^{-1} \iff (a^{\mathrm{T}})^{-1} P_n^{-1} a^{-1} = (P_n)^{-1}.$$

Hence, $(a^{\mathrm{T}})^{-1} \in \mathrm{Sp}(n, q)$ with Remark 5.6 and so $a^{\mathrm{T}} \in \mathrm{Sp}(n, q)$. Therefore, the claim follows by Lemma 5.15. □

We also prove similar results when $k > m$.

Lemma 5.17: Let $n, f \in \mathbb{N}$, $j \in \{1, \dots, n\}$, $q = p^f$ a prime power and $a \in \mathrm{Sp}(n, q)$ such that there exists a $k \in \{m+1, \dots, n\}$ with $a_{k,j} \neq 0$ and $a_{i,j} = 0$ for all $i \in \{1, \dots, n\} - \{k\}$. Then

$$a_{n-k+1,i} = \begin{cases} a_{k,j}^{-1}, & \text{if } i = n - j + 1 \text{ and } j > m, \\ -a_{k,j}^{-1}, & \text{if } i = n - j + 1 \text{ and } j \leq m, \\ 0, & \text{else.} \end{cases}$$

Proof. We know by Theorem 5.5 that $aP_n a^{\mathrm{T}} = P_n$. Since $P_n^{-1} = \begin{pmatrix} 0 & -J_n \\ J_n & 0 \end{pmatrix}$ we see that

$$(aP_n a^{\mathrm{T}})^{-1} = (P_n)^{-1} \iff (a^{\mathrm{T}})^{-1} P_n^{-1} a^{-1} = (P_n)^{-1}.$$

Hence, $(a^{\mathrm{T}})^{-1} \in \mathrm{Sp}(n, q)$ with Remark 5.6 and so $a^{\mathrm{T}} \in \mathrm{Sp}(n, q)$. Therefore, the claim follows by Lemma 5.14. □

Now we can prove the correctness of Algorithm 5.

Theorem 5.18: The algorithm UnitriangularDecompositionSp (Algorithm 5) works correctly and terminates.

Proof. Let $a \in \mathrm{Sp}(n, q)$.
We start with column n and pick the least i such that the entry $a_{i,n} \neq 0$. Such an entry must exist, since $\det(a) = 1$. By Theorem 5.11, Theorem 5.10 and Lemma 5.8 we can use the matrices $S_{i,j}$ to clear all entries $a_{k,n}$ for $k \in \{i+1, \ldots, n\}$. Therefore, everything above and below the entry $a_{i,n}$ is zero and we used only lower-unitriangular matrices. Again we use Theorem 5.11, Theorem 5.10 and Lemma 5.8 and clear every entry $a_{i,k}$ left from $a_{i,n}$ for $k \in \{1, \ldots, n-1\}$. Therefore, the row i also contains only one non-zero entry.
By Lemma 5.14, Lemma 5.17 and Lemma 5.16 it follows that row $n-i+1$ as well as column $n-n+1$ contain only one non-zero entry. Hence, we can ensure that in the course of the algorithm one non-zero entry remains in the cleared row and column.
Therefore, the claim follows iteratively. □

5.4 Implementation of the Bruhat Decomposition in Sp

In this subsection we discuss the implementation of the algorithm for the Bruhat decomposition in Sp (Algorithm 5) and how it can be described with an MSLP. In addition, we discuss how a monomial matrix can be described as the product of particular generators.

First we describe the Leedham-Green-O'Brien standard generators [GB] of $\mathrm{Sp}(n,q)$, because we want to use these generators to solve the word problem described in chapter 2.

Definition 5.19: Let q be a prime power, say $q = p^f$, n a positive even integer and $\omega \in \mathbb{F}_q$ a primitive element. The *Leedham-Green-O'Brien standard generators* [GB] of $\mathrm{Sp}(n,q)$ are the following:

A transposition :	$s_0 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ and $\widetilde{s} = \begin{pmatrix} s_0 & 0 \\ 0 & I_{n-2} \end{pmatrix}$,
A transvection :	$t_0 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ and $\widetilde{t} = \begin{pmatrix} t_0 & 0 \\ 0 & I_{n-2} \end{pmatrix}$,
A diagonalmatrix :	$\widetilde{\delta} = \mathrm{diag}(\omega, \omega^{-1}, 1, \ldots, 1)$,
Product of two cycles of length $\frac{n}{2}$:	$\widetilde{v} = \begin{pmatrix} 0 & I_{n-2} \\ I_2 & 0 \end{pmatrix}$,
Two 2 cycles :	$u_0 = \begin{pmatrix} 0 & I_2 \\ I_2 & 0 \end{pmatrix}$ and $\widetilde{u} = \begin{pmatrix} u_0 & 0 \\ 0 & I_{n-4} \end{pmatrix}$,
A transvection :	$x_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$ and $\widetilde{x} = \begin{pmatrix} x_0 & 0 \\ 0 & I_{n-4} \end{pmatrix}$.

Leedham-Green and O'Brien use a different basis in their paper. Therefore, we have to perform a basis transformation. Let $\{e_1, \ldots, e_n\}$ be the standard basis of $\mathbb{F}_q^n$. We use the following matrix

$$T_n^{\mathrm{Sp}} = \begin{pmatrix} e_1 & e_n & e_2 & e_{n-1} & \ldots & e_{\frac{n}{2}} & e_{\frac{n}{2}+1} \end{pmatrix} \in \mathrm{GL}(n,q).$$

Example 5.20: For $n = 6$ we have

$$T_6^{\mathrm{Sp}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

When we use the matrix T_n^{Sp} to change the basis , we obtain the following generating set.

Definition 5.21: Let q be a prime power, say $q = p^f$, n a positive even integer and $\omega \in \mathbb{F}_q$ a primitive element. The *Leedham-Green-O'Brien standard generators* [GB] of $\mathrm{Sp}(n, q)$ transformed with T_n^{Sp} are the following:

A transposition :	$s = \begin{pmatrix} 0 & 0 & 1 \\ 0 & I_{n-2} & 0 \\ -1 & 0 & 0 \end{pmatrix}$,
A transvection :	$t = I_n + E_{1,n}$,
A diagonalmatrix :	$\delta = \mathrm{diag}(\omega, 1, \ldots, 1, \omega^{-1})$,
Product of two cycles of length $\frac{n}{2}$:	$v = \begin{pmatrix} 0 & I_{\frac{n}{2}-1} & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & I_{\frac{n}{2}-1} & 0 \end{pmatrix}$,
Two 2 cycles :	$u = \begin{pmatrix} J_2 & 0 & 0 \\ 0 & I_{n-4} & 0 \\ 0 & 0 & J_2 \end{pmatrix}$,
A transvection :	$x = I_n + E_{n-1,1} + E_{n,2}$.

Example 5.22: We get the following generating set for $n = 6$ and $q = 7$:

$$s = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad t = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad \delta = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \end{pmatrix},$$

$$v = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad u = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad x = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Now we can continue as in the paper "Straight-line programs with memory and matrix Bruhat decomposition" from A. C. Niemeyer, T. Popiel and C. E. Praeger [NPP].

First we write the matrices from the Algorithm UnitriangularDecompositionSp as words in terms of the LGO generators.

Theorem 5.23: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_q$ a primitive element of $\mathbb{F}_q$. Let (s, t, δ, v, u, x) be the LGO standard generators as in Definition 5.21. We set

$$u_1 \coloneqq u,$$
$$u_i \coloneqq v^{-1} u_{i-1} v.$$

Then all matrices of the Algorithm UnitriangularDecompositionSp (Algorithm 5) are a product of the LGO standard generators as shown in the following tables:

Entry in the bottom left block.

Matrix	**Word in terms of the LGO generators**
$S_{n,2}(\omega^\ell)$	$(\delta^{-\ell} v^{-2} \delta^\ell v^2) x (\delta^{-\ell} v^{-2} \delta^\ell v^2)^{-1}$
$S_{n,i}(\omega^\ell)$	$u^{-1} v^{-1} S_{n,i-1}(\omega^\ell) v u$
$S_{j,i}(\omega^\ell)$	$u_{n-j+1} S_{j-1,i}(\omega^\ell) u_{n-j+1}$

Entry on the anti diagonal.

Matrix	**Word in terms of the LGO generators**
$S_{n,1}((\omega^2)^\ell)$	$s^{-1} \delta^\ell t^{-1} \delta^{-\ell} s$
$S_{i-1,j+1}((\omega^2)^\ell)$	$v^{-1} S_{i,j}((\omega^2)^\ell) v$

Entry in the top left or bottom right block.

Matrix	**Word in terms of the LGO generators**
$S_{2,1}(\omega^\ell)$	$us(\delta^{-\ell} v^{-2} \delta^\ell v^2) x (\delta^{-\ell} v^{-2} \delta^\ell v^2)^{-1} s^{-1} u^{-1}$
$S_{i+1,i}(\omega^\ell)$	$v^{-1} S_{i,i-1}(\omega^\ell) v$
$S_{i,j}(\omega^\ell)$	$u_j S_{i,j+1}(\omega^\ell) u_j$

Proof. We know by Theorem 4.35 that $(\delta^{-\ell}v^{-2}\delta^{\ell}v^2) = \mathrm{diag}(\delta^{-\ell}, 1, \delta^{\ell}, 1, \ldots, 1, \delta^{-\ell}, 1, \delta^{\ell})$ and therefore

$$(\delta^{-\ell}v^{-2}\delta^{\ell}v^2)^{-1}x(\delta^{-\ell}v^{-2}\delta^{\ell}v^2) = I_n + \omega^{\ell}E_{n-1,1} + \omega^{\ell}E_{n,2}.$$

For the matrices in the top left block we calculate $us(\delta^{-\ell}v^{-2}\delta^{\ell}v^2)x(\delta^{-\ell}v^{-2}\delta^{\ell}v^2)^{-1}s^{-1}u^{-1} = usS_{n,2}(\omega^{\ell})s^{-1}u^{-1}$. First we compute $S_{n,2}(\omega^{\ell})s^{-1}$. Now $S_{n,2}(\omega^{\ell})s^{-1}$ induces a column operation on $S_{n,2}(\omega^{\ell})$ where we replace the last column by -1 times the first column and the first column by 1 times the last column. Hence

$$S_{n,2}(\omega^{\ell})s^{-1} = \begin{pmatrix} 0 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & I_{n-4} & 0 & 0 \\ 0 & 0 & 0 & 1 & -\omega^{\ell} \\ 1 & \omega^{\ell} & 0 & 0 & 0 \end{pmatrix}.$$

$s(S_{n,2}(\omega^{\ell})s^{-1})$ induces a row operation on $S_{n,2}(\omega^{\ell})s^{-1}$ where we replace the first row by 1 times the last row and the last row by -1 times the first row. Hence $s(S_{n,2}(\omega^{\ell})s^{-1}) = I_n + \omega^{\ell}E_{1,2} - \omega^{\ell}E_{n-1,n}$ and thus

$$usS_{n,2}(\omega^{\ell})s^{-1}u^{-1} = S_{2,1}(\omega^{\ell}).$$

For an entry on the anti diagonal we have that

$$\delta^{\ell}t^{-1}\delta^{-\ell} = I_n - \omega^{2\ell}E_{1,n}$$

and thus $s^{-1}\delta^{\ell}t^{-1}\delta^{-\ell}s = I_n + \omega^{2\ell}E_{1,n}$. Since $\{w^2, w^4, w^6, \ldots, w^{q-1}\}$ is a generating set for $\mathbb{F}_q$ as a $\mathbb{F}_p$-vector space we can get every matrix $S_{n,1}(\omega^{\ell})$.
The remaining parts follow by Theorem 4.35 and Theorem 4.33. □

Finally we can estimate the length and the amount of memory slots for an MSLP, which computes the Bruhat decomposition for an element in the symplectic group.

Theorem 5.24: Let $q = p^f$ with p prime, $f \geq 1$, $n = 2m$ an even integer with $n \geq 3$ and $a \in \mathrm{Sp}(n, q)$. Set $b = 16 + 3f + m$ and

$$\begin{aligned}\lambda = {} & 25f + n - 6 + n\log_2(m) + m + mf + n\log_2(q) + \\ & \frac{n^2 - 2n}{2}(f + 2\log_2(q) + 2\log_2(m-2) + n).\end{aligned}$$

There exists a b-MSLP, S, of length at most λ such that if S is evaluated with memory containing the list $\{s, t, \delta, v, u, x\}$ then S outputs memory containing (u_1, u_2) with $a = u_1 w u_2$ the Bruhat decomposition of a.

Proof. In order to avoid the logarithm oracle we use the following method to calculate the transvections.
Since $\mathbb{F}_q$ is a $\mathbb{F}_p$ vector space with basis $\{\omega^0, \omega^1, \ldots, \omega^{f-1}\}$, where ω is a primitive element of $\mathbb{F}_q$, we only need to save the transvections $S_{2,1}(\omega^0), S_{2,1}(\omega^1), \ldots, S_{2,1}(\omega^{f-1})$ and obtain $S_{2,1}(\beta)$ for $\beta \in \mathbb{F}_q$ through

$$S_{2,1}(\beta) = S_{2,1}(\sum_{i=0}^{f-1} k_i\omega^i) = \prod_{i=0}^{f-1} S_{2,1}(\omega^i)^{k_i}.$$

Hence, we just calculate the transvections $S_{2,1}(\omega^0), S_{2,1}(\omega^1), \ldots, S_{2,1}(\omega^{f-1})$ and store them. We use the same method for the transvections with entries in the lower left block and entries on the anti diagonal.
Therefore, we start to calculate these transvections. First we look at the transvections $S_{2,1}(\omega^i)$ for $i \in \{0, \ldots, f-1\}$ and use the formula of Theorem 5.23. We need 6 operations for $\delta^{-\ell}v^{-2}\delta^{\ell}v^2$ and thus 12 operations in total for

$$us(\delta^{-\ell}v^{-2}\delta^{\ell}v^2)x(\delta^{-\ell}v^{-2}\delta^{\ell}v^2)^{-1}s^{-1}u^{-1}.$$

Since we do this f times, we need $12f$ operations. Analogously follows that we need $8f$ operations for the

$$S_{n,2}(\omega^i) = (\delta^{-\ell}v^{-2}\delta^{\ell}v^2)x(\delta^{-\ell}v^{-2}\delta^{\ell}v^2)^{-1}$$

for $i \in \{0, \ldots, f-1\}$ and $5f$ operations for the

$$S_{n,1}((\omega^2)^{\ell}) = s^{-1}\delta^{\ell}t^{-1}\delta^{-\ell}s$$

for $i \in \{0, \ldots, f-1\}$. Moreover we store the u_i for $i \in \{1, \ldots, m-2\}$ of Theorem 5.23 and need $2(m-3) = n-6$ operations to calculate them. In total we need $25f + n - 6$ operations for the initialization.
Now starts the algorithm 5. We consider a matrix with a non-zero entry in every position. We need at most $f - 1 + f2\log_2(p) = f - 1 + 2\log_2(q)$ operations to calculate a transvection $S_{2,1}(\beta), S_{n,2}(\beta)$ or $S_{n,1}(\beta)$. To shift $S_{n,1}(\beta)$ to $S_{m+1,m}(\beta)$ we need with repeated squaring (see Section 2.2(ii) in [NPP]) at most $2\log_2(m) + 2$ operations. Therefore, we need at most

$$m(f - 1 + 2\log_2(q) + 2\log_2(m) + 2) = n\log_2(m) + m + mf + n\log_2(q)$$

operations for the anti diagonal.
To shift $S_{n,2}(\beta)$ by the index j we need at most $3 + 2\log_2(m-2)$ operations and by the index i we need at most $(m-1)2$ operations. At most we need to do this $m^2 - m = \frac{n^2-2n}{4}$ times. If we sum up this results, we get

$$\tfrac{n^2-2n}{4}(f + 2\log_2(q) + 2\log_2(m-2) + n)$$

Analogously follows that we need at most

$$\tfrac{n^2-2n}{4}(f-1+2\log_2(q)+2+2\log_2(m-2)+(m-1)2) \leq \tfrac{n^2-2n}{4}(f+2\log_2(q)+2\log_2(m-2)+n)$$

for all the transvections in the top left and lower right block, yielding the claimed maximum total length of the MSLP:

$$25f + n - 6 + n\log_2(m) + m + mf + n\log_2(q) + \tfrac{n^2-2n}{2}(f + 2\log_2(q) + 2\log_2(m-2) + n)$$

We need 12 memory slots for the LGO standard generator and their inverse elements, 2 memory slots for repeated squaring, 2 memory slots for the results u_1 and u_2, 1 memory slot for the current transvection and 1 additional memory slot to save some calculations.
Moreover we need f memory slots for the matrices $S_{2,1}(\omega^i)$ for $i \in \{0, \dots, f-1\}$, f memory slots for the matrices $S_{n,2}(\omega^i)$ for $i \in \{0, \dots, f-1\}$, f memory slots for the matrices $S_{n,1}(\omega^{2i})$ for $i \in \{0, \dots, f-1\}$ and $m-2$ memory slots for the remaining u_i for $i \in \{1, \dots, m-2\}$. Thus we need

$$12 + 2 + 2 + 1 + 1 + f + f + f + m - 2 = 16 + 3f + m$$

memory slots. □

Now we transform a monomial matrix in $\mathrm{Sp}(n,q)$ into a diagonal matrix in $\mathrm{Sp}(n,q)$ and write this with the LGO standard generators.

Let $M_{\mathrm{Sp}} \leq \mathrm{Sp}(n,q)$ be the subgroup of monomial matrices of $\mathrm{Sp}(n,q)$. Let $\{e_1, \dots, e_n\}$ be the standard basis of $\mathbb{F}_q$. We define

$$\varphi_{\mathrm{Sp}} \colon M_{\mathrm{Sp}} \to S_n$$

such that $g \in M_{\mathrm{Sp}}$ permutes the subspaces $\langle e_1\rangle, \dots, \langle e_n\rangle$ in the same way that $\varphi_{\mathrm{Sp}}(g)$ permutes $1, \dots, n$.
We have that $\varphi_{\mathrm{Sp}}(M_{\mathrm{Sp}})$ is generated by the images of the LGO standard generators s, v, u from Definition 5.21 and hence

$$\varphi_{\mathrm{Sp}}(M_{\mathrm{Sp}}) = \langle (1,n), (1,2)(n-1,n), (1,2,\dots,\tfrac{n}{2})(\tfrac{n}{2}+1, n, n-1, \dots, \tfrac{n}{2}+2)\rangle =: G_{Sp}.$$

Since $G_{Sp} = G_{SUE}$, we can use the same algorithm from chapter 4 to write every element of G_{Sp} as a product of the generators and get the same result as in Theorem 4.38.

Finally, we have to write a diagonal matrix in Sp in terms of the LGO standard generators. We observe the following property of diagonal matrices in Sp which helps us solve the problem at hand.

Theorem 5.25: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $a = \mathrm{diag}(a_1, \dots, a_n) \in \mathrm{Sp}(n,q)$ a diagonal matrix. Then $a_i = a_{n-i+1}^{-1}$ for $i \in \{1, \dots, n\}$.

Proof. We use Theorem 5.5 and know that $a \cdot P_m \cdot a^{\mathrm{T}} = P_m$. We have

$$\begin{aligned}
a \cdot P_m \cdot a^{\mathrm{T}} &= a \cdot P_m \cdot a \\
&= \operatorname{diag}(a_1, \ldots, a_n) \cdot P_m \cdot \operatorname{diag}(a_1, \ldots, a_n) \\
&= \begin{pmatrix}
0 & 0 & \ldots & 0 & 0 & \ldots & 0 & a_1 \\
0 & 0 & \ldots & 0 & 0 & \ldots & a_2 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & 0 & a_m & \ldots & 0 & 0 \\
0 & 0 & \ldots & -a_{m+1} & 0 & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & -a_{n-1} & \ldots & 0 & 0 & \ldots & 0 & 0 \\
-a_n & 0 & \ldots & 0 & 0 & \ldots & 0 & 0
\end{pmatrix} \cdot \operatorname{diag}(a_1, \ldots, a_n) \\
&= \begin{pmatrix}
0 & 0 & \ldots & 0 & 0 & \ldots & 0 & a_1 a_n \\
0 & 0 & \ldots & 0 & 0 & \ldots & a_2 a_{n-1} & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & 0 & a_m a_{m+1} & \ldots & 0 & 0 \\
0 & 0 & \ldots & -a_{m+1} a_m & 0 & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & -a_{n-1} a_2 & \ldots & 0 & 0 & \ldots & 0 & 0 \\
-a_n a_1 & 0 & \ldots & 0 & 0 & \ldots & 0 & 0
\end{pmatrix} \\
&= P_m.
\end{aligned}$$

Hence, the claim follows. □

We use some special matrices to write diagonal matrices. We define these matrices and show that we can express them in terms of LGO generators.

Definition 5.26: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_q$ a primitive element. We set

$$h_j^{\mathrm{Sp}} \coloneqq \operatorname{diag}(\underbrace{1, \ldots, 1}_{j-1}, \omega, 1, \ldots, 1, \omega^{-1}, \underbrace{1, \ldots, 1}_{j-1})$$

for $j \in \{1, \ldots, \frac{n}{2}\}$.

The next theorem shows how we can write these matrices in terms of LGO standard generators and, therefore, lie in $\mathrm{Sp}(n, q)$.

Theorem 5.27: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_q$ a primitive element. We have

$$h_1^{\mathrm{Sp}} = \delta,$$
$$h_j^{\mathrm{Sp}} = v^{-1} h_{j-1}^{\mathrm{Sp}} v.$$

Proof. We use φ_{Sp} and notice that

$$\varphi_{\mathrm{Sp}}(v) = (1, 2, \ldots, m)(m+1, n, n-1, \ldots, m+2).$$

Clearly $h_1^{\mathrm{Sp}} = \delta \in \mathrm{Sp}(n, q)$ and we have $v^{-1} h_{j-1}^{\mathrm{Sp}} v$ induces the desired permutation on the diagonal entries.

□

Now we can write every diagonal matrix in $\mathrm{Sp}(n, q)$ with the h_j^{Sp}.

Theorem 5.28: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_q$ a primitive element. Let $a = \mathrm{diag}(a_1, \ldots, a_n) \in \mathrm{Sp}(n, q)$ be a diagonal matrix. Then a is a product of the h_j^{Sp}.

Proof. We know by Theorem 5.25 that the matrix a has the form

$$a = \mathrm{diag}(a_1, \ldots, a_n) = \mathrm{diag}(a_1, \ldots, a_m, a_m^{-1}, \ldots, a_1^{-1}).$$

For $j \in \{1, \ldots, m\}$ we choose $k_j \in \{1, \ldots, q\}$ such that $\omega^{k_j} = a_j$. We show that

$$a = \prod_{j=1}^{m} (h_j^{\mathrm{Sp}})^{k_j}.$$

We have

$$\begin{aligned}
\prod_{j=1}^{m} (h_j^{\mathrm{Sp}})^{k_j} &= (h_1^{\mathrm{Sp}})^{k_1} \ldots (h_m^{\mathrm{Sp}})^{k_m} \\
&= \mathrm{diag}(\omega, 1, \ldots, 1, \omega^{-1})^{k_1} \ldots \mathrm{diag}(1, \ldots, 1, \omega, \omega^{-1}, 1, \ldots, 1)^{k_m} \\
&= \mathrm{diag}(\omega^{k_1}, 1, \ldots, 1, (\omega^{-1})^{k_1}) \ldots \mathrm{diag}(1, \ldots, 1, \omega^{k_m}, (\omega^{-1})^{k_m}, 1, \ldots, 1) \\
&= \mathrm{diag}(a_1, 1, \ldots, 1, a_1^{-1}) \ldots \mathrm{diag}(1, \ldots, 1, a_m, (a_m)^{-1}, 1, \ldots, 1) \\
&= \mathrm{diag}(a_1, \ldots, a_m, a_m^{-1}, \ldots, a_1^{-1}) \\
&= a.
\end{aligned}$$

Hence, the claim follows. □

Finally, we calculate the maximum length and the number of memory slots of an MSLP, which computes a diagonal matrix from the symplectic group.
We proceed analogously to the proof of Proposition 3.4 in [NPP].

Theorem 5.29: Let $n, f, m \in \mathbb{N}$, $q = p^f$ a prime power, $n = 2m$ and $\omega \in \mathbb{F}_q$ a primitive element. Let $\lambda = m+n-2+n\log_2(q)$ and $b = 16$. Let $h \in \mathrm{Sp}(n,q)$ be a diagonal matrix given in the form $h = \mathrm{diag}(\omega^{k_1}, ..., \omega^{k_n})$. There exists a b-MSLP, S, of length at most λ such that if S is evaluated with memory containing the set $X = \{s, s^{-1}, t, t^{-1}, \delta, \delta^{-1}, v, v^{-1}, u, u^{-1}x, x^{-1}\}$ then S outputs final memory containing h.

Proof. Our MSLP writes h as described in Theorem 5.28, with the h_j^{Sp} computed in terms of the LGO standard generators as per Theorem 5.27. Computing the h_j^{Sp} via Theorem 5.27 costs $(\frac{n}{2}-1)2 = n-2$ MSLP instructions: h_1^{Sp} is already the standard generator δ and each subsequent h_j^{Sp} is composed by conjugating with v.
Powering each h_j^{Sp} up to $(h_j^{\mathrm{Sp}})^{k_j}$ via repeated squaring costs (see Section 2.2(ii) in [NPP]) at most

$$2\log_2(k_j) \leq 2\log_2(q)$$

MSLP instructions, and so computing all of the $(h_j^{\mathrm{Sp}})^{k_j}$ requires at most

$$\tfrac{n}{2}(2\log_2(q)) = n\log_2(q)$$

instructions. Moreover, $\frac{n}{2} = m$ instructions are required to multiply the $(h_j^{\mathrm{Sp}})^{k_j}$ together. Overall, this results in a maximum total length of the MSLP of

$$m + n - 2 + n\log_2(q).$$

The memory quota for the MSLP is as follows. First we store the input X. At most one memory slot is needed to store the h_j^{Sp} because the recursion for the h_j^{Sp} refers back to h_{j-1}^{Sp} and v, hence, each h_j^{Sp} can overwrite the one from which it is computed. Two memory slots are needed for computing the $(h_j^{\mathrm{Sp}})^{k_j}$ (see Section 2.2(ii) in [NPP]), and one final memory slot is needed to multiply these together. Thus we need

$$12 + 2 + 1 + 1 = 16$$

memory slots. □

References

[NPP] A. C. Niemeyer, T. Popiel and C. E. Praeger, *Straight-line programs with memory and matrix Bruhat decomposition*. CoRR abs/1305.5617 (May 2017)

[GB] C. R. Leedham-Green and E. A. O'Brien, *Constructive recognition of classical groups in odd characteristic*. Journal of Algebra 322 (2009), 833–881

[Taylor] Donald E. Taylor, *The geometry of the classical groups*, volume 9 of Sigma Series in Pure Mathematics (Heldermann Verlag, Berlin, 1992)

[MGR] Henrik Bäärnhielm, Derek Holt, C.R. Leedham-Green, and E.A. O'Brien, *A practical model for computation with matrix groups*. J. Symbolic Comput. 68 (2015), part 1, 27–60

[NTWA] W.C. Winnie Li *NUMBER THEORY WITH APPLICATIONS*. Series on University Mathematics: Volume 7 (February 1996)

[SPGT] Yaim Cooper *Generators of the Symplectic Group*. under: `http://www-math.mit.edu/~dav/sympgen.pdf` (retrieved on 01.08.2019)

Acknowledgements

My thanks go to the many people who have supported me during my bachelor thesis in numerous ways. I thank Prof. Dr. Alice Niemeyer for the opportunity to write my bachelor thesis on this interesting subject and the support. I would like to thank Priv.-Doz. Dr. Markus Kirschmer for dealing with my sometimes quite technical bachelor thesis. I also thank M. Sc. Dominik Bernhardt, who is a patient, motivated and very supportive supervisor. Furthermore I would like to thank Csaba Schneider, who has dealt with the same problem in Magma and compared his implementation with mine. In addition, I would like to thank the entire Lehrstuhl B for the very pleasant and fun climate, the countless little tips, assistance in general and the openness towards me.

Furthermore, without naming individual, I thank my private environment for the support.