Enumerative Coding for Line Polar Grassmannians
Ilaria Cardinali and Luca Giuzzi

Abstract—Codes arising from Grassmannians have been widely investigated, both as generalization of Reed–Muller codes and for applications to network coding. Recently we introduced some new codes, arising from Polar Grassmannians, namely the set of all subspaces of a vector space $\mathbb{F}_q^{2n+1}$ which are totally singular with respect to a given non-degenerate quadratic form. The aim of the present paper is to present an efficient enumerative coding and decoding strategy for line polar Grassmann codes.

Index Terms—Enumerative Coding, Polar Grassmannian, Linear Code

I. INTRODUCTION

Let $\mathbb{F}_q$ be a finite field with $q$ elements. If $k \in \{1, \ldots, n-1\}$, the $k$-Grassmannian $G_{n,k}$ of the $n$-dimensional vector space $U := \mathbb{F}_q^n$ is the geometry whose points are the $k$-dimensional subspaces of $U$ and whose lines are the pencils $\ell_{X,Y}$ of $k$-subspaces though a given $(k-1)$-subspace $X$ and all contained in a given $(k+1)$-subspace $Y$. It is well known that $G_{n,k}$ can be embedded into the projective space $\text{PG}(W)$ with $W = \mathbb{F}_q^k \cdot U$ as an algebraic variety, by means of the Plücker embedding $\varepsilon_k$; see [1, Chapter VII] and [2, Chapter XVI] for details. Grassmannians over finite fields have been extensively used both in order to construct network codes, see [3], and to obtain projective linear codes. In both cases, efficient ways to represent the points of a Grassmannian are needed; see, in particular, [4], [5].

We now recall some basics on projective codes. Given any set of points $\Omega = \{P_1, \ldots, P_N\}$ in $\text{PG}(W)$, it is always possible to construct a projective linear code $C(\Omega)$ of length $N$ by taking the coordinates of the points $P_i$ as columns of a generator matrix. We observe that $C(\Omega)$ is defined just up to code equivalence; for more details, see [6]. It is a basic result in the theory of projective codes that there is a correspondence between hyperplanes of $\text{PG}(W)$ having maximal intersection with $\Omega$ and codewords of minimum weight for $C(\Omega)$.

The projective codes $C_{n,k}$ (called the Grassmann codes) arising from the projective system $\varepsilon_k(G_{n,k}) \subseteq \text{PG}(W)$ have been first introduced in [7], [8] as a generalization of Reed–Muller codes of the first order and widely investigated ever since; their monomial automorphism groups and minimum weights are well understood, see [9]–[14]. These codes have a very low data rate; as such it is paramount to be able to describe efficient encoding and decoding algorithms acting locally on the components. To this aim, in [1], enumerative coding for Grassmannians is considered and some efficient algorithms are presented; see also [5].

Recently, in [15], we introduced a new family of linear codes, say $P_{n,k}$, arising from the Plücker embedding of orthogonal Grassmannians. Recall that a $k$-orthogonal Grassmannian is a geometry $\Delta_{n,k}$ whose points are all the totally singular $k$-subspaces of a vector space $V := \mathbb{F}_q^{2n+1}$ with respect to a given non-degenerate parabolic quadratic form $q_n : V \to \mathbb{F}_q$ and whose lines are either the lines $\ell_{X,Y}$ of $G_{2n+1,k}$ with $Y$ totally singular with respect to $q_n$ for $k < n$ or sets of the form $\ell_X := \{Y : X < Y < X^\perp, \dim Y = n, Y$ totally singular\}, with $\dim X = n-1$ for $k = n$. Here, $X^\perp$ denotes the orthogonal space to $X$ with respect to $q_n$.

We remark that the Grassmannian graph arising from a Polar Grassmannian $\Delta_{n,k}$ has diameter strictly larger than that of the corresponding $k$-Grassmannian $G_{n,k}$; thus, Polar Grassmannians might provide for some extra correction capabilities in the case of random network coding.

Observe that, as any point of $\Delta_{n,k}$ is also a point of $G_{2n+1,k}$, an orthogonal Grassmann code $P_{n,k}$ can be obtained from the ordinary Grassmann code $C_{2n+1,k}$ by just deleting all the columns corresponding to $k$-spaces which are non-singular with respect to $q_n$; it can thus be considered as a punctured version of $C_{2n+1,k}$. In general, the parameters of $P_{n,k}$ are

$$N = \#\Delta_{n,k} = \prod_{i=1}^k (q^{n-i+1}+1)(q^{n-i+1}-1) \over (q^i-1),$$

$$K = \begin{cases} \binom{2n+1}{k} + \binom{2n+1}{k-1} - \binom{2n+1}{k-2} & \text{for } q \text{ odd,} \\ \binom{2n+1}{k} & \text{for } q \text{ even,} \end{cases} \quad d \geq 2q^{k(n-k)} - 1.$$ 

For $q$ odd, the dimension of $P_{n,k}$ is the same as that of $G_{2n+1,k}$, while for $q$ even it is lower; see [15]. More recently in [16] the minimum distance for Line Polar Grassmann codes $P_{n,2}$ in odd characteristic has been fully determined to be $d = q^{4n-4} - q^{3n-4}$. This proves that our codes somehow compare favorably with the corresponding ordinary Grassmann codes.

In the present paper we introduce an enumerative coding scheme for Line Polar Grassmannians $\Delta_{n,2}$, following the approach of [17]. It will be apparent that the analysis required is more involved than that of [4] for Grassmann Codes. The main reason is that we need unavoidably to keep track of the behavior of the quadratic form defining the polar space. Still, it appears that an actual implementation of the codes might be possible using some local properties.

A. Organization of the paper

In Section II, we enumerate the points of $\Delta_{n,2}$, i.e. the totally singular lines of $V$ for $q_n$, spanned by vectors with a prescribed prefix. This is used in Section III to define an enumerative coding scheme on the pointset $\Delta_{n,2}$ according to the approach of [17]. We fully prove its behavior and, in Subsection II-A we analyze the overall complexity of
our encoding scheme. Section IV is dedicated to applications of the scheme here introduced to the projective codes \( P_{n,k} \) arising from Line Polar Grassmannians; there we suggest some encoding and error correcting schemes which can act locally on the components of the codewords.

## II. Enumerative Coding

### A. Preliminaries

As in Section I, let \( V \) denote a vector space \( \mathbb{F}_q^{2n+1} \) of odd dimension \( 2n+1 \) over \( \mathbb{F}_q \). It is well known that, up to projectivities, there is only one class of non-degenerate quadratic forms on \( V \). Thus, it is not restrictive to preliminary fix a non-degenerate quadratic form \( q_n \), determining a quadric \( Q(2n,q) \) of parabolic type in the projective space \( \text{PG}(2n,q) : = \text{PG}(V) \). The form \( q_n \) will be chosen in such a way as to minimize the complexity of the algorithm described in Section I. In case encoding with respect to a different quadratic form were to be required, a change of reference will be needed.

Throughout this paper we shall say that a line \( \ell \) of \( \text{PG}(V) \) is totally singular if \( \ell \subseteq Q(2n,q) \). It is straightforward to see that \( \ell \) is totally singular if and only if given any two points \( A,B \in \ell \) with \( A \neq B \) we have \( q_n(A) = q_n(B) = 0 \) and \( b_n(A, B) := q_n(A + B) - q_n(A) - q_n(B) = 0 \), where \( b_n \) is the symmetric bilinear form associated to \( q_n \). Observe that for \( q \) even, the form \( b_n \) is also alternating (and degenerate), while for \( q \) odd \( b_n \) is non-degenerate symmetric.

Consider \( x, y \in V \) having coordinates \( (x_i)_{i=1}^{2n+1} \) and \( (y_i)_{i=1}^{2n+1} \) with respect to a fixed reference system and take

\[
q_n(x) = x_1^2 + \sum_{i=1}^{n} x_{2i+1}x_{2i+1}.
\]

Then

- For \( q \) an odd prime power,
  \[
b_n(x, y) := x_1y_1 + \frac{1}{2} \sum_{i=1}^{n} (x_{2i}y_{2i+1} + y_{2i}x_{2i+1});
\]

- for \( q \) a power of two,
  \[
b_n(x, y) := \sum_{i=1}^{n} (x_{2i}y_{2i+1} + y_{2i}x_{2i+1}).
\]

We shall represent any totally singular line \( \ell \) of \( \text{PG}(V) \) by providing a \( 2 \times (2n+1) \) matrix \( G \) in Row Reduced Echelon Form (RREF, in brief) whose rows span \( \ell \). Recall that a matrix \( G \) is in RREF, also called Hermite normal form, when the leading non-zero entry in each row of \( G \) is 1 and all entries above a leading non-zero entry are zero. Clearly, there is a bijection between \( 2 \times (2n+1) \) matrices in RREF form in which neither row is null and 2-dimensional vector subspaces of \( V \).

Denote by \( \mathcal{R}_{2,t} \) the set of all \( 2 \times t \) matrices in RREF over \( \mathbb{F}_q \) and also let

\[
\mathcal{R}_2 := \bigcup_{i=0}^{2n+1} \mathcal{R}_{2,i}.
\]

In this section we shall discuss the function

\[
n_q : \mathcal{R}_2 \times \mathbb{N} \to \mathbb{N}
\]

mapping \( (S,n) \in \mathcal{R}_2 \times \mathbb{N} \) to the number of totally singular lines contained in the quadric \( Q(2n,q) \) whose RREF representation begins with \( S \).

In other words, for any given \( n \) and any given

\[
S = \begin{pmatrix} \alpha_1 & \ldots & \alpha_t \\ \beta_1 & \ldots & \beta_t \end{pmatrix} \in \mathcal{R}_2,
\]

\[
n_q(S,n) \text{ is the number of totally singular lines } \ell = \langle \hat{A}, \hat{B} \rangle \text{ spanned by the two vectors } \hat{A} \text{ and } \hat{B} \text{ having coordinates}
\]

\[
\hat{A} := (\alpha_1, \alpha_2, \ldots, \alpha_t, x_{t+1}, x_{t+2}, \ldots, x_{2n+1})
\]

\[
\hat{B} := (\beta_1, \beta_2, \ldots, \beta_t, y_{t+1}, y_{t+2}, \ldots, y_{2n+1}).
\]

When

\[
G := \begin{pmatrix} \hat{A} - \lambda \hat{B} \\ \hat{B} \end{pmatrix},
\]

we shall say that \( S \) is the prefix of \( G \) or that the leading part of \( G \) is \( S \).

The first step in our algorithm is to normalize the datum \( S \) so that either \( \alpha_t = 0 \) or \( \beta_t = 0 \). This can always be done and does not alter the value of \( n_q(S,n) \). Observe however that this yields a matrix which is not in RREF; we shall call this form close to RREF (in brief CRREF). Actually, if either \( \alpha_t = 0 \) or \( \beta_t = 0 \) in \( S \), then there is nothing to do; otherwise, given a matrix \( G \) with prefix \( S \), if \( \beta_t \neq 0 \), we can always subtract from the first row \( \lambda = \alpha_t \beta_t^{-1} (\neq 0) \) times the second row and get

\[
G' := \begin{pmatrix} \hat{A} - \lambda \hat{B} \\ \hat{B} \end{pmatrix}.
\]

The matrix \( G' \) has prefix

\[
S' = \begin{pmatrix} \alpha_1 - \lambda \beta_1 & \ldots & 0 \\ \beta_1 & \ldots & \beta_t \end{pmatrix}.
\]

and represents the same line as \( G \). As stated above, \( n_q(S,n) = n_q(S',n) \) since \( \langle \hat{A}, \hat{B} \rangle = \langle \hat{A} - \lambda \hat{B}, \hat{B} \rangle \).

In the remainder of this section we shall denote by

\[
A := (\alpha_1, \alpha_2, \ldots, \alpha_t, 0, 0, \ldots, 0),
\]

\[
B := (\beta_1, \beta_2, \ldots, \beta_t, 0, 0, \ldots, 0)
\]

two vectors of length \( 2n+1 \) whose first \( t \) components correspond to the rows of \( S \). The \( x_i \)'s and the \( y_i \)'s are always indeterminates.

### TABLE I

Useful Numbers

<table>
<thead>
<tr>
<th>( \Xi )</th>
<th># of points of ( \Xi )</th>
<th># of lines of ( \Xi )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{PG}(v,q) )</td>
<td>( \frac{(q^v-1)(q^{v-1}+1)}{q-1} )</td>
<td>( \frac{(q^v-1)(q^{v-1}+1)}{q-1} )</td>
</tr>
<tr>
<td>( Q(2n,q) )</td>
<td>( \frac{(q^{2n}-1)(q^{2n-1}+1)}{q-1} )</td>
<td>( \frac{(q^{2n}-1)(q^{2n-1}+1)}{q-1} )</td>
</tr>
<tr>
<td>( Q^+(2n-1,q) )</td>
<td>( \frac{(q^{2n-1}-1)(q^{2n-2}+1)}{q-1} )</td>
<td>( \frac{(q^{2n-1}-1)(q^{2n-2}+1)}{q-1} )</td>
</tr>
</tbody>
</table>
Observe that when $S \in \mathcal{R}_2$, the value $n_q(S, n)$ is the number of solutions in the unknowns $x_i$’s and $y_i$’s, $i = t + 1, \ldots, 2n + 1$, to the following system of equations:

\[
\begin{align*}
q_n(A) &= 0 \\
q_n(B) &= 0 \\
b_n(A, B) &= 0.
\end{align*}
\]  

(2)

Henceforth, it will be convenient to distinguish two cases, depending on the parity of the length $t$ of the prefix $S$. They will not be fully independent: as it will be seen, our algorithm for $t$ even requires some computations with some auxiliary prefixes $S’$ of odd length and, likewise, some cases with a prefix of odd length can be dealt by reducing to different cases where the prefix contains an even number of components. In any case, as the detailed analysis shall show, this will not lead to an infinite recursion and will ultimately provide the correct value.

B. Even $t$

If $t = 0$, then we just have to count all the (totally singular) lines of $Q(2n, q)$; thus, $n_q(0, n) = N$ and we are done. When $t \neq 0$, System (2) becomes

\[
\begin{align*}
\alpha_1^2 + \sum_{i=1}^{t/2-1} \alpha_2 \alpha_{2i+1} &+ \alpha_t x_{t+1} + \sum_{i=t/2+1}^{n} x_{2i} x_{2i+1} = 0 \\
\beta_1^2 + \sum_{i=1}^{t/2-1} \beta_2 \beta_{2i+1} &+ \beta_t y_{t+1} + \sum_{i=t/2+1}^{n} y_{2i} y_{2i+1} = 0 \\
2\alpha_1 \beta_1 &+ \alpha_t y_{t+1} + \beta_t x_{t+1} + \sum_{i=1}^{t/2-1} (\alpha_2 \beta_{2i+1} + \alpha_{2i+1} \beta_{2i}) + \\
&+ \sum_{i=t/2+1}^{n} (x_{2i} y_{2i+1} - x_{2i+1} y_{2i}) = 0.
\end{align*}
\]  

(3)

We now distinguish some subcases. Recall that, as $S$ is assumed to be already in CRREF form, both $\alpha_t \neq 0$ and $\beta_t \neq 0$ cannot occur.

- **$\alpha_t = 0$ and $\beta_t \neq 0$.** The last equation of (3) yields $x_{t+1}$ as a function of the remaining variables; the second equation determines $y_{t+1}$ as a function of $y_{t+2}, \ldots, y_{2n+1}$ and the first equation provides constraints on $x_{t+2}, \ldots, x_{2n+1}$. So the overall number of solutions is

\[
\left[ \text{# \{solutions of the first equation\}} \times \left( q^{2n-t} \right) \right] \quad \text{possibilities for } x_{t+2}, \ldots, x_{2n+1}.
\]

Observe now that as $S$ is in CRREF, we always have $A \neq 0$. Thus, each vector solution of the first equation corresponds to a distinct point; furthermore, as also $B \neq 0$ (this is because $\beta_t \neq 0$), different choices of $y_{t+1}, \ldots, y_{2n+1}$ give different lines. Thus, we have $n_q(S, n) = \eta_0(c) \cdot q^{2n-t}$, where $c = q_n(A)$ and $\eta_0(c)$ is the number of solutions of the first equation of System (3), that is

\[
\eta_0(c) := \begin{cases} 
(q - 1)(\#Q^+(2n - t - 1, q)) + 1 
& \text{if } c = 0 \\
\#PG(2n - t - 1, q) - \#Q^+(2n - t - 1, q) 
& \text{if } c \neq 0.
\end{cases}
\]

(4)

- **$\alpha_t \neq 0$ and $\beta_t = 0$.** This case is analogous to the previous one with the roles of the first and the second equation reversed. The only difference is that we also have to take into account the possibility $B = 0$. Indeed,

1) for $B \neq 0$ we argue as above and we get $n_q(S, n) = \eta_0(q_n(B)) \cdot q^{2n-t}$.

2) for $B = 0$ we need to count the number of points of the hyperbolic quadric having equation $y_{t+2} y_{t+3} + \ldots + y_{2n} y_{2n+1} = 0$; thus we get

\[
n_q(S, n) := \frac{q^{2n-t-1}}{q - 1} \left( q^{n-t/2} - 1 \right) (q^{n-t/2} + 1).
\]

- **$\alpha_t = \beta_t = 0$.** In this case the matrix $G$ has the form

\[
G = \begin{pmatrix} \alpha_1 & \cdots & \alpha_{t-1} & 0 & x_{t+1} & \cdots & x_{2n+1} \\
\beta_1 & \cdots & \beta_{t-1} & 0 & y_{t+1} & \cdots & y_{2n+1} \\
\end{pmatrix}.
\]

As the coefficients of $x_{t+1}$ and $y_{t+1}$ are both zero System (2) is formally the same as that defined by

\[
G’ = \begin{pmatrix} \alpha_1 & \cdots & \alpha_{t-1} & x_{t+2} & \cdots & x_{2n+1} \\
\beta_1 & \cdots & \beta_{t-1} & y_{t+2} & \cdots & y_{2n+1} \\
\end{pmatrix}.
\]

This is to say that formally System (2) corresponds to what arises from $n_q(S’, n-1)$ where

\[
S’ = \begin{pmatrix} \alpha_1 & \cdots & \alpha_{t-1} \\
\beta_1 & \cdots & \beta_{t-1} \\
\end{pmatrix}.
\]

Here however there are two further free parameters, namely $x_{t+1}$ and $y_{t+1}$ which may arbitrarily vary in $F_q$. More in detail, any solution $\bar{x}_{t+2} \cdots \bar{x}_{2n+1}$ to the problem $n_q(S’, n-1)$ yields several solutions of the form

\[
\begin{pmatrix} \eta & \bar{x}_{t+2} & \cdots & \bar{x}_{2n+1} \\
\xi & \bar{y}_{t+2} & \cdots & \bar{y}_{2n+1} \\
\end{pmatrix}
\]

to the problem $n_q(S, n)$ where $(\eta, \xi) \in F_q^2$. As the vectors $\bar{x} = (\bar{x}_{t+2}, \ldots, \bar{x}_{2n+1})$ and $\bar{y} = (\bar{y}_{t+2}, \ldots, \bar{y}_{2n+1})$ are already normalized, distinct values of $(\eta, \xi)$ do actually correspond to distinct solutions of the problem. In order to fully count the number of solutions thus obtained we have to distinguish three subcases:

1) If $A = B = (0, \ldots, 0)$, then $n_q(S, n)$ is the number of lines of $Q(2n, q)$ contained in a subspace $\Pi$ of codimension $t$ described by the equations

\[
x_1 = 0, x_2 = \cdots = x_t = 0.
\]

In particular, $Q’ := \Pi \cap Q(2n, q)$ has equation

\[
\begin{align*}
x_1 &= x_2 = \cdots = x_t = 0 \\
x_{t+2} x_{t+3} + x_{t+4} x_{t+5} + \cdots + x_{2n} x_{2n+1} &= 0
\end{align*}
\]

(5)
and it is a cone of vertex \( W = (0, 0, \ldots, 0, 1, 0, \ldots, 0) \) over a hyperbolic quadric isomorphic to \( Q^+(2n - t - 1, q) \). Hence \( n_q(S, n) = \sigma q^2 + \#Q^+(2n - t - 1, q) \), where \( \sigma \) is the number of lines of \( Q' \); see Table II for the actual values.

2) If \( A \not= 0 \) and \( B \not= 0 \), observe that

a) the problem \( n_q(S', n - 1) \) provides the number of vectors \((x_{t+2}, \ldots, x_{2n+1}), (y_{t+2}, \ldots, y_{2n+1})\) “completing” the sequences \((\alpha_1, \ldots, \alpha_{t-1})\) and \((\beta_1, \ldots, \beta_{t-1})\).

b) for distinct \((\eta, \zeta) \in \mathbb{F}_q^2\) we get distinct lines (as the datum is supposed to be given in CRREF).

Thus, the overall number of solutions is \( n_q(S, n) := q^2 n_q(S', n - 1) \).

3) if \( A \not= 0 \) and \( B = 0 \) (\( A = 0 \) and \( B \not= 0 \) cannot occur), then \( n_q(S, n) := q^2 n_q(S', n - 1) + \sigma_1 \), where \( \sigma_1 \) corresponds to solutions to \( n_q(S, n) \) of the form

\[
\begin{pmatrix}
0 & x_{t+2} & \ldots & x_{2n+1} \\
1 & 0 & \ldots & 0
\end{pmatrix}
\]

consequently, \( \sigma_1 = n_0(q_n(A)) \); see [4].

The case \((\alpha_1, \beta_1) = (0, 0)\) has thus been reduced to a case of the form \( n_q(S', n - 1) \) in which \( t \) is odd and the rank of the quadric is \( n - 1 \).

C. Odd \( t \)

We have to study

\[
\begin{align*}
\alpha^2_1 + & \sum_{i=1}^{(t-1)/2} \alpha_1 \alpha_{2i+1} + \sum_{i=(t+1)/2}^n x_{2i} x_{2i+1} = 0 \\
\beta^2_1 + & \sum_{i=1}^{(t-1)/2} \beta_1 \beta_{2i+1} + \sum_{i=(t+1)/2}^n y_{2i} y_{2i+1} = 0 \\
2\alpha_1 \beta_1 + & \sum_{i=1}^{(t-1)/2} (\alpha_{2i} \beta_{2i+1} + \alpha_{2i+1} \beta_{2i}) + \\
& + \sum_{i=(t+1)/2}^n (x_{2i} y_{2i+1} + x_{2i+1} y_{2i}) = 0.
\end{align*}
\]

As before, suppose

\[ S = \begin{pmatrix} \alpha_1 & \ldots & \alpha_t \\ \beta_1 & \ldots & \beta_t \end{pmatrix} \]

and distinguish several cases.

- \( A \not= 0 \) and \( B \not= 0 \). For any \((\gamma, \delta) \in \mathbb{F}_q^2\), write

\[ S_{\gamma, \delta} = \begin{pmatrix} \alpha_1 & \ldots & \alpha_t \\ \beta_1 & \ldots & \beta_t \\ \gamma & \delta \end{pmatrix} \]

Clearly,

\[ n_q(S, n) = \sum_{(\gamma, \delta) \in \mathbb{F}_q^2} n(S_{\gamma, \delta}, n). \]

In other words the sets of lines determined by the various \( S_{\gamma, \delta} \) as \( \gamma, \delta \) vary are all disjoint (and all need to be computed).

### Table II

<table>
<thead>
<tr>
<th>( q_n(A) )</th>
<th>( b_n(A, B) )</th>
<th>( q_n(B) )</th>
<th>( \Delta )</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>( q - 1 )</td>
</tr>
<tr>
<td>0</td>
<td>( \neq 0 )</td>
<td>( \neq 0 )</td>
<td>( \ast )</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>( \neq 0 )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \neq 0 )</td>
<td>( \neq 0 )</td>
<td>0</td>
<td>( \ast )</td>
<td>1</td>
</tr>
<tr>
<td>( \neq 0 )</td>
<td>( \neq 0 )</td>
<td>0</td>
<td>( \square )</td>
<td>2</td>
</tr>
<tr>
<td>( \neq 0 )</td>
<td>( \neq 0 )</td>
<td>( \neq 0 )</td>
<td>( \square )</td>
<td>0</td>
</tr>
</tbody>
</table>

\( \Delta := b_n(A, B)^2 - q_n(A) q_n(B) \).

\( \ast \) means that we do not care for the value

We decompose the problem taking into consideration the following contributions

- for each pair with \( \gamma \neq 0, \delta \neq 0 \), let \( \lambda = \delta^{-1} \gamma \). We have \( n_q(S, \gamma, \delta, n) = n_q(S', n - 1) \) where

\[ S' := \begin{pmatrix} \alpha_1 - \lambda \beta_1 & \ldots & \alpha_t - \lambda \beta_t \\ \beta_1 & \ldots & \beta_t \\ 0 \end{pmatrix} \]

Since \( S' \) has \( t + 1 \) columns, we are lead back to Subsection II-B i.e.

\[ n_q(S', n) = q^{2n-t-1} \eta_1(c), \]

where \( \eta_1(c) \) is the number of solution of the equation \( q_n(A - \lambda B) = 0 \), as \( \lambda \) varies. If \( c := q_n(A - \lambda B) \), then the contribution from a given configuration turns out to be

\[ \eta_1(c) := \begin{cases} (q - 1) \#Q^+(2n - t - 2, q) + 1 & \text{for } c = 0 \\ \#PG(2n - t - 2, q) - \#Q^+(2n - t - 2, q) & \text{for } c \neq 0. \end{cases} \]

Using the properties of the quadratic form \( q_n \) we can always write

\[ c = q_n(A) + \lambda b_n(A, B) + \lambda^2 q_n(B). \]

Let now \( \xi \) be the number of solutions (in the unknown \( \lambda \)) of (6) different from 0. The possible values assumed by \( \xi \) are outlined in Table II for \( q \) odd and in Table III for \( q \) even. Observe that for \( q \) odd, the symbols \( \square \) and \( \square \) represent respectively the set of all non-zero square elements and the set of non-square elements in \( \mathbb{F}_q \). For \( q = 2^s \) even, by \( \operatorname{Tr}_2(x) \) we denote the absolute trace of \( x \), that is to say

\[ \operatorname{Tr}_2(x) := \sum_{i=0}^{s-1} x^{2^i}. \]

The overall contribution for all cases in which \( \gamma \neq 0 \) and \( \delta \neq 0 \) turns out to be

\[ \psi_1 := (q - 1) q^{2n-t-1} \begin{pmatrix} \xi \eta_1(0) & (q - 1 - \xi) \eta_1(1) \end{pmatrix}. \]

* means that we do not care for the value
The solutions of the first equation are either \( \gamma \neq 0 \) and \( \delta = 0 \) or \( \gamma = 0 \) and \( \delta \neq 0 \): there are \( 2q - 2 \) cases which have to be investigated.  
For the case \( \gamma = 0 \) and \( \delta \neq 0 \) we have  
\[
S_{0,\delta} = \begin{pmatrix} \alpha_0 & \cdots & \alpha_t & 0 \\ \beta_1 & \cdots & \beta_t & \delta \end{pmatrix}
\]

Arguing as for \( t \) even, we see that the contribution is equal to the number of solutions of  
\[
\alpha_1^2 + \sum_{i=1}^{(t-1)/2} \alpha_{2i} \alpha_{2i+1} + \sum_{(t+3)/2}^{n} x_{2i} x_{2i+1} = 0
\]
times \( q^{2n+1-(t+3)+1} = q^{2n-t-1} \). This depends just on \( \alpha_1, \ldots, \alpha_t \) and not on the choice of \( \delta \) and can be computed as above. The case \( \delta = 0 \) and \( \gamma \neq 0 \) is entirely analogous. Denote the sum of both of these contributions as \( \psi_2 \).

- For \( (\gamma, \delta) = (0, 0) \) the contribution to consider is \( \psi_3 = q^2 n_q(S, n-1) \), where

\[
S := \begin{pmatrix} \alpha_0 & \cdots & \alpha_t \\ 0 & \cdots & \delta \end{pmatrix}
\]

We have thus to recourse and consider the corresponding case for the same value of \( t \) (which is odd) but in lower dimension.

- **\( A = B = 0 \).** (Note that \( A = 0 \) and \( B \neq 0 \) cannot happen according to the convention here followed). In close analogy to Subsection II-B for \( A = B = 0 \), we need to determine the number of lines of a hyperbolic quadric isomorphic to \( Q^+(2n-t, q) \) with equation

\[
x_{t+1} x_{t+2} + \cdots + x_{2n} x_{2n+1} = 0
\]

see Table II for the values.

- **\( A \neq 0 \) and \( B = 0 \).** The solutions of the first equation in \( \langle \gamma \rangle \) are to be counted as vectors, but those given by the second as points (i.e., they have to be normalized). In particular, we have

\[
S_{\gamma, \delta} = \begin{pmatrix} \alpha_0 & \alpha_2 & \cdots & \alpha_t & \gamma \\ 0 & \cdots & 0 & 0 & \delta \end{pmatrix}
\]
to study. There are \( q \) possibilities for \( \gamma \), but ultimately just 2 for \( \delta = y_{t+1} \), namely either \( \delta = 0 \) or \( \delta = 1 \).

For \( \delta = 1 \), then we can always assume \( \gamma = 0 \) by reducing to CRREF; thus we end up with the equivalent problem induced by

\[
S' = \begin{pmatrix} \alpha_0 & \alpha_2 & \cdots & \alpha_t & 0 \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix}
\]

and we have an overall contribution which amounts to \( n_q(S', n) \). Here the matrix \( S' \) has an even number of columns (we are in the \( t \) even case) but \( \beta_{t+1}' \neq 0 \). In particular, this even case does not recourse into an odd one.

- For \( (\delta, \gamma) = (0, 0) \) we consider

\[
S' = \begin{pmatrix} \alpha_0 & \alpha_2 & \cdots & \alpha_t & \gamma \end{pmatrix}
\]

Here we need to determine the number of vectors \( (0, \ldots, 0, x_{t+2}, \ldots, x_{2n+1}) \) times the number of (projective) points \( \langle (0, \ldots, 0, y_{t+2}, \ldots, y_{2n+1}) \rangle \) solution of System \( \langle 5 \rangle \) as induced by \( S' \). Indeed, we end up with having to consider

\[
\begin{aligned}
q_n(A) + \gamma x_{t+2} + x_{t+3} x_{t+4} + \cdots + x_{2n} x_{2n+1} = 0 \\
y_{t+3} y_{t+4} + \cdots + y_{2n} y_{2n+1} = 0 \\
\gamma y_{t+2} + x_{t+3} y_{t+4} + x_{t+4} y_{t+3} + \cdots + x_{2n+1} y_{2n} = 0.
\end{aligned}
\]

Here

* there are \( q - 1 \) distinct possibilities for \( \gamma \);
* \( x_{t+2} \) is recovered from the first equation where \( x_{t+3}, \ldots, x_{2n+1} \) are arbitrary; in particular there are overall \( q^{2n+1-(t+3)+1} = q^{2n-t-1} \) possible values for \( (x_{t+2}, \ldots, x_{2n+1}) \). However, as there is a least \( i \) such that \( y_i \neq 0 \), the corresponding entry \( x_i \) has to be taken to be zero in CRREF; thus the number of vectors \( (x_{t+2}, \ldots, x_{2n+1}) \) yielding distinct lines is always \( q^{2n-t-2} \).

* The overall contribution is thus

\[
(q-1)q^{2n-t-2} \# Q^+(2n-t-2, q).
\]

- For \( (\delta, \gamma) = (0, 0) \) we consider

\[
S' = \begin{pmatrix} \alpha_0 & \alpha_2 & \cdots & \alpha_t & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}
\]

<table>
<thead>
<tr>
<th>( q_n(A) )</th>
<th>( b_n(A, B) )</th>
<th>( q_n(B) )</th>
<th>( \Theta )</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>*</td>
<td>( q - 1 )</td>
</tr>
<tr>
<td>0</td>
<td>#0</td>
<td>#0</td>
<td>*</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>#0</td>
<td>*</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>#0</td>
<td>0</td>
<td>*</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>#0</td>
<td>0</td>
<td>*</td>
<td>0</td>
</tr>
<tr>
<td>#0</td>
<td>0</td>
<td>#0</td>
<td>*</td>
<td>1</td>
</tr>
<tr>
<td>#0</td>
<td>#0</td>
<td>#0</td>
<td>*</td>
<td>1</td>
</tr>
<tr>
<td>#0</td>
<td>#0</td>
<td>#0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>#0</td>
<td>#0</td>
<td>#0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\( \Theta := \text{Tr}_2 \left( \frac{q_n(A) q_n(B)}{b_n(A, B)^2} \right) \)

* means that we do not care for the value or the function is undefined.
The equations become
\[
\begin{align*}
q_n(A) + x_{t+3}x_{t+4} + \ldots + x_{2n}x_{2n+1} &= 0 \\
0y_{t+2} + y_{t+3}y_{t+4} + \ldots + y_{2n}y_{2n+1} &= 0 \\
x_{t+3}y_{t+4} + x_{t+4}y_{t+3} + \ldots + x_{2n+1}y_{2n+1} &= 0.
\end{align*}
\]

In particular, we can study the problem
\[
G'' = \begin{pmatrix}
\alpha_1 & \alpha_2 & \ldots & \alpha_t & x_{t+3} & \ldots & x_{2n+1} \\
0 & 0 & \ldots & 0 & y_{t+3} & \ldots & y_{2n+1}
\end{pmatrix},
\]
induced by \(n_q(S'', n - 1)\) where
\[
S'' = \begin{pmatrix}
\alpha_1 & \alpha_2 & \ldots & \alpha_t \\
0 & 0 & \ldots & 0
\end{pmatrix}
\]
and observe that any solution of the system defined by \(G''\) in \((x_{t+3}, \ldots, x_{2n+1})\) and \((y_{t+3}, \ldots, y_{2n+1})\) determines some solutions of the system defined by \(G'\) having as leading part \(S''\). Notice that different values of \(x_{t+2}\) and \(y_{t+2}\) yield different points for the first row. Also, different values of \(y_{t+2}\) might yield different lines, as the second row is normalized in the first nonzero coordinate of a solution of \(G''\).

Thus, for any solution to \(S''\) we have \(q^2\) possible solutions to \(S''\). Furthermore, when \(y_{t+2} = 1\) and, consequently, \(x_{t+2} = 0\) we can also have some extra solutions to \(G''\), namely those corresponding to \(y := (y_{t+3}, \ldots, y_{2n+1}) = 0\). These amount to \(\eta_1(q_n(A))\), as each solution of
\[
\begin{align*}
q_n(A) + x_{t+3}x_{t+4} + \ldots + x_{2n}x_{2n+1} &= 0 \\
x_{t+2} &= 0
\end{align*}
\]
is a solution of \(G''\). Thus,
\[
n_q(S', n) = q^2 \times n_q(S'', n - 1) + \eta_1(q_n(A)).
\]

**D. Full description of the algorithm**

We outline the full algorithm. We remark that here by \(S[i, j]\) we mean the entry in the \(i\)-th row and \(j\)-th column of the array \(S\), while by \(S[i]\) we just mean the \(i\)-th row of \(S\).

Let \(j \in \{0, 1\}\) and \(j + t\) even. Define
\[
\eta_j(c) = \begin{cases}
\frac{(q^{n-(t+j)}/2 - 1)(q^{n-(t+j)}/2 - 1) + 1}{q^{2n-(t+j)} - 1}(q^{n-(t+j)/2 - 1} - 1)(q^{n-(t+j)/2 - 1} + 1) & \text{for } c = 0 \\
\frac{q^{2n-(t+j)} - 1}{q^{n-(t+j)/2 - 1} - 1} & \text{for } c \neq 0.
\end{cases}
\]

**function** \(n_q(S, n)\)

- if \(n = 1\) then
  - return 0
- end if
- if \(S = 0\) then
  - return \(N\)
- end if

\(t \leftarrow \# \text{ of columns of } S\)

- if \(t = 2n + 1\) then
  - if \(q_n(S[1]) = q_n(S[2]) = b_n(S[1], S[2]) = 0\) then
    - return \(R := 1\)
  - else
    - return \(R := 0\)
  - end if
- end if

- if \(2|t\) then
  - return \(n_q^G(S, n)\)
- else
  - return \(n_q^{\mathcal{E}}(S, n)\)
- end if

**function** \(\Xi_q(S)\)

- \(a \leftarrow q_n(S[1])\)
- \(b \leftarrow q_n(S[2])\)
- \(c \leftarrow b_n(S[1], S[2])\)

- if \(a = 0\) and \(b = 0\) and \(c = 0\) then
  - return \(q - 1\)
- end if

- if \(\Delta \leftarrow c^2 - ab\)
  - if \(b \neq 0\) then
    - if \(\Delta = 0\) then
      - \(r \leftarrow 1\)
    - else if \(\Delta \in \square\) then
      - \(r \leftarrow 2\)
    - else if \(\Delta \notin \square\) then
      - \(r \leftarrow 0\)
    - end if
  - end if

  - if \(a = 0\) then
    - \(r \leftarrow \max(0, r - 1)\)
  - end if

- return \(r\)

- else
  - if \(a \neq 0\) and \(c \neq 0\) then
    - return 1
  - else
    - return 0
  - end if

- end if

- if \(2|q\) then
  - if \(c = 0\) then
    - if \(a = 0\) or \(b = 0\) then
      - return 0
    - else
      - return 1
    - end if
  - else
    - \(\triangleright c \neq 0\)
  - end if

- end if

- if \(a = 0\) then
  - if \(b = 0\) then
    - return 0
  - else
    - return 1
  - end if

- end if
III. Enumeration

It is possible to provide an enumerator for the lines contained in $Q(2n, q)$ using the function $n_q$ introduced in the previous section. We follow the approach of [17]. For the convenience of the reader, we here provide an explicit proof of the behavior of this enumerative coding scheme in the non-binary case. Our arguments are indeed a consequence of [17], where, however, the detail is illustrated in detail only for the case of binary encoding.

Fix arbitrarily a total order $≤$ on the vectors of $𝔽_2^n$ and, as usual, write $A < B$ if and only if $A ≤ B$ and $A ≠ B$.

Let $I = \{0, \ldots, N-1\}$ where $N = \frac{(q^{2n-1}-1)(q^{2n-1})}{(q^2-1)(q-1)}$. Put

$$t : \begin{cases} \Delta_{n,2} \rightarrow I \\ G \rightarrow \iota(G) := \sum_{j=1}^{2n+1} \sum_{X \prec G_j} n_q((G_1, \ldots, G_{j-1}, X, n) \ (7)$$

where $G$ is a matrix in RREF whose rows span a totally singular line of $Q(2n, q)$ and $G_j$ denotes the $j$-th column of $G$. Extend now the order $\prec$ defined on the vectors of $𝔽_q^n$ to matrices of order $2 \times (2n+1)$ lexicographically; that is to say $G \prec H$ if and only if $G < H$ and there is an $i$ such that $\forall j < i$ $G_j = H_j$ and $G_i < H_i$.

We say that $G_k \in 𝕪_2^n$ is allowable if the matrix $(G_1, \ldots, G_k, X_{k+1}, \ldots, X_{2n+1})$ represents a totally singular line for some values of $X_{k+1}, \ldots, X_{2n+1}$; in other words, $G_k$ is allowable for $(G_1, \ldots, G_k-1)$ if and only if $n_q((G_1, \ldots, G_{k-1}, G_k, n) > 0$.

Theorem 1. The index function $\iota$ defined in (7) is a bijection.

Proof: We prove that $\iota$ is injective. Let

$$G = (H_1, H_2, \ldots, H_{i-1}, G_i, \ldots, G_{2n+1}); \quad H = (H_1, H_2, \ldots, H_{i-1}, H_i, \ldots, H_{2n+1}).$$

We will show that if $G_i \ll H_i$ then $\iota(G) < \iota(H)$. Suppose
$G_i < H_i$ and define

$$\iota^\prec(G) := \{(X_1, \ldots, X_{2n+1}) : X_1 < G_1\} \cup \{(X_1, \ldots, X_{2n+1}) : X_2 < G_2\} \cup \ldots \cup \{(G_1, G_2, \ldots, G_n, X_{2n+1}) : X_{2n+1} < G_{2n+1}\},$$

where the elements of the sets are assumed all to be totally singular lines. Observe that if $G_1 = H_1, \ldots, G_{i-1} = H_{i-1}$ and $G_i < H_i$, then

$$G \in \{(G_1, \ldots, G_{i-1}, X_i, X_{i+1}, \ldots, X_{2n+1}) : X_i < H_i\}.$$  

In particular, under our assumptions, $G \in \iota^\prec(H)$. Furthermore, if $G \in \iota^\prec(H)$, then $\iota^\prec(G) \subseteq \iota^\prec(H)$. Indeed, if $Y := (Y_1, \ldots, Y_{2n+1}) \in \iota^\prec(G)$, then there exists $j$ such that $Y_i = (G_1, \ldots, Y_{i-1}, G_i, Y_{i+1}, \ldots, Y_{2n+1}) : X_i < H_i$;

- If $j < i$, then we also have $Y_1 = H_1, \ldots, Y_{j-1} = H_{j-1}$ and $Y_j < H_j$, thus $Y \in \iota^\prec(H)$.
- If $j = i$, then $Y_i < G_i < H_i$ and $Y \in \{(G_1, \ldots, G_{i-1}, X_i, X_{i+1}, \ldots, X_{2n+1}) : X_i < H_i\}$; thus, $Y \in \iota^\prec(H)$.
- If $j > i$, then $Y_i = G_i < H_i$; thus, as

$$G_1, \ldots, G_{i-1}, X_i, X_{i+1}, \ldots, X_{2n+1} \in \{(G_1, \ldots, G_{i-1}, X_i, X_{i+1}, \ldots, X_{2n+1}) \in (X_1, \ldots, X_{2n+1}) : X_i < H_i\}$$

for any allowable vector $X_1, \ldots, X_{2n+1}$ we have also

$$Y \in \{(G_1, \ldots, G_{i-1}, X_i, X_{i+1}, \ldots, X_{2n+1}) : X_i < H_i\}$$

and, consequently, $Y \in \iota^\prec(H)$.

As $G \in \iota^\prec(H)$ but $G \not\in \iota^\prec(G)$, the above inclusion is proper. We now claim that $\iota(G) = \#\iota^\prec(G)$. Note that

$$\#\{(G_1, G_2, \ldots, G_{i-1}, X_i, X_{i+1}, \ldots, X_{2n+1}) : X_i < G_i\} = \sum_{X_i < G_i} \#\{(G_1, G_2, \ldots, G_{i-1}, X_i, X_{i+1}, \ldots, X_{2n+1}) = \sum_{X_i < G_i} n_q((G_1, \ldots, G_{i-1}, X_i, n)).$$

Furthermore, as the sets in (8) are disjoint,

$$\#\iota^\prec(G) = \#\{(X_1, \ldots, X_{2n+1}) : X_1 < G_1\} + \#\{(G_1, X_2, \ldots, X_{2n+1}) : X_2 < G_2\} + \ldots + \#\{(G_1, G_2, \ldots, G_{2n+1}) : X_{2n+1} < G_{2n+1}\} = \sum_{X_1 < G_1} n_q((X_1, n)) + \sum_{X_2 < G_2} n_q((G_1 X_2, n)) + \ldots + \sum_{X_{2n+1} < G_{2n+1}} n_q((G_1 G_2 \ldots G_{2n} X_{2n+1}, n)) = \sum_{i=1}^{2n+1} \sum_{X_i < G_i} n_q((G_1, \ldots, G_{i-1}, X_i, n).$$

To conclude observe that for any two distinct lines represented by matrices $G$ and $H$ we have either $G \in \iota^\prec(H)$ or $H \in \iota^\prec(G)$. The former yields $\iota^\prec(G) \subseteq \iota^\prec(H)$, whence $\iota(G) < \iota(H)$; the latter, in an entirely analogous way, $\iota(G) > \iota(H)$. In any case $G \neq H$ gives $\iota(G) \neq \iota(H)$ and $\iota$ is injective. As the sets under consideration are finite and of the same cardinality, $\iota$ is also surjective.

In particular, as $\iota(G) = \#\iota^\prec(G)$ is the number of matrices $K$ with $K \preccurlyeq G$ representing totally singular lines, we have that for any given $i \in \{0, \ldots, N - 1\}$, the matrix $G$ such that $\iota(G) = i$ is given by $G = \min\{Y : K \preccurlyeq Y \forall K \text{ such that } \iota(K) < i\}$.

Now we can invert the function $\iota$. Proceed as follows:

**Require:** $i \in \{0, \ldots, N - 1\}$

1. $i_1 \leftarrow 1$
2. for $k = 1, \ldots, 2n + 1$ do
   a. $M \leftarrow \{Y : \sum_{X \preccurlyeq Y} n_q((G_1, \ldots, G_{k-1}, X, n)) \leq i_k$ and $n_q((G_1, \ldots, G_{k-1}, X, n)) > i_k\}$
   b. $G_k \leftarrow \max M$
   c. $\theta(G_k) \leftarrow \sum_{X \preccurlyeq G_k} n_q((G_1, \ldots, G_{k-1}, X, n))$
   d. $i_{k+1} \leftarrow i_k - \theta(G_k)$
3. end for

**return** $G = (G_1, \ldots, G_{2n+1})$

Observe that the requirement that $G_k$ is allowable means that we have to actually check $n_q((G_1, \ldots, G_k, n)) > 0$. Columns which are allowable in position $k$ may not be allowable in position $k - 1$ or vice-versa.

In general, $i_{k+1} = i - \sum_{j=1}^{k} \theta(G_j)$.

On the other hand, as for any correct datum $(G_1, \ldots, G_{2n+1}, G_k)$ we have

$$\iota(G) = \sum_{k=1}^{2n+1} \sum_{X < G_k} n_q((G_1, \ldots, G_{k-1}, X, n)) = \sum_{k=1}^{2n+1} \theta(G_k),$$

we also get $i_{2n+1} = 0$; this for any $G$ representing a totally singular line of $Q(2n, q)$.

**Theorem 2.** Suppose $G = (G_1, \ldots, G_{2n+1})$ represents a totally singular line and let $\iota(G) = i$. Then $G_k$ is the maximum allowable column of $G$ with respect to the order $\preccurlyeq$ such that $\theta(G_k) \leq i_k$ where $i_k = i - \sum_{j=1}^{k-1} \theta(G_j)$.

**Proof:** Define

$$\Theta(G_k) := \{(G_1, \ldots, G_{k-1}, Y, \ldots) : X < G_k\} = \Lambda(G_k) := \{(G_1, \ldots, G_{k-1}, Y, \ldots) \in \iota^\prec(G) : Y \preccurlyeq G_k\}.$$

Then,

$$\Lambda(G_1) = \{(Y, \ldots) \in \iota^\prec(G) : Y \preccurlyeq G_1\} = \iota^\prec(G).$$

We have

$$\#\Theta(G_k) = \sum_{X < G_k} n_q((G_1, \ldots, G_{k-1}, X, n)) = \theta(G_k).$$

On the other hand, we can write

$$\Lambda(G) = \iota^\prec(G) \setminus \{((X_1, \ldots) : X_1 < G_1) \cup \{(G_1, X_2, \ldots) : X_2 < G_2\} \cup \ldots \cup \{(G_1, G_2, \ldots, X_{k-1}, \ldots) : X_{k-1} < G_{k-1}\}\} = \iota^\prec(G) \setminus \bigcup_{j=1}^{k-1} \Theta(G_j).$$
Thus,
\[ \#\Lambda(G_k) = \ell(G) - \sum_{j=1}^{k-1} \theta(G_j) = i_k. \]

We distinguish two cases:

- \( k = 1 \): suppose \( G_1 \) is not to be maximum and such that \( \theta(G_1) \leq i_1 = i \). Then, there is a different element \( G' \in \mathbb{F}_q^2 \) with \( G_1 \prec G' \) and \( \theta(G'_1) \leq i = i_1 \). By construction, \( \Lambda(G_1) \subset \Theta(G'_1) \). Observe that \( G \in \Theta(G'_1) \) but \( G \not\subset \Lambda(G_1) \). Thus, the inclusion is proper. Moving to the cardinalities we have
\[ i = \#\Lambda(G_1) < \#\Theta(G'_1) = \theta(G'_1) \leq i, \]
a contradiction.

- \( k \rightarrow k+1 \): suppose all \( G_j \)'s for \( j \leq k \) be maximum and that \( G_{k+1} \) is not the maximum element such that \( \theta(G_{k+1}) \leq i_{k+1} \). Then, as before, there is a \( G'_{k+1} \) such that \( G_{k+1} \prec G'_{k+1} \) with \( \theta(G'_{k+1}) \leq i_{k+1} \). For any \( Y \leq G_{k+1} \) we have \( Y \prec G'_{k+1} \); thus the following inclusion holds true
\[ \Lambda(G_{k+1}) = \langle \{G_1, \ldots, G_k, Y, \ldots\} : Y \leq G_{k+1} \rangle \subset \langle \{G_1, \ldots, G_k, X, \ldots\} : X \prec G'_{k+1} \rangle = \Theta(G'_{k+1}). \]
Furthermore, as \( G \in \Theta(G'_{k+1}) \) but \( G \not\subset \Lambda(G_{k+1}) \), the above inclusion is proper. Thus,
\[ i_{k+1} = \#\Lambda(G_{k+1}) < \#\Theta(G'_{k+1}) = \theta(G'_{k+1}) \leq i_{k+1}, \]
a contradiction.

### A. Complexity

We analyze the complexity of the algorithm in Section II-D by tracking the elementary operations involved. We adopt the same notation as in Section II; \( n \) is the rank of the quadric \( Q(2n, q) \), \( t \) is the length of a prefix \( S \).

We schematically trace the operations involved in computing \( n(q)(S, n) \).

1. \( t \) even \( \) (see the function \( n_E(q)(S, n) \) in the algorithm in Section II):
   - 1) first we reduce the rows of \( S \) in CRREF; this requires \( t \) products and \( t \) sums;
   - 2) if \( A = 0 \) and \( B = 0 \), then \( n_E(q)(S, n) \) can be obtained by a fixed formula, with complexity \( O(1) \);
   - 3) if \( A \neq 0 \) and \( \alpha_t \neq 0 \) or \( \beta_t \neq 0 \) we can determine the value of \( n_q(q)(S, n) \) by computing at most one of the forms \( q(A) \) or \( q(B) \). Each of these requires \( t \) products and \( t \) sums. Thus, the overall complexity is \( O(t) \).
   - 4) if \( A \neq 0 \) and both \( \alpha_t = \beta_t = 0 \) we need to call the function \( n_O(q)(S', n-1) \) with a prefix of length \( t-1 \). In the non recursive case, say EvenCase', the complexity of this computation is \( O(t) \).
2. \( t \) odd \( \) (see the function \( n_O(q)(S, n) \) in the algorithm in Section II):
   - 1) we first compute \( q(A) \), \( q(B) \) and \( b_n(A, B) \), each of these requires \( t \) products and \( t \) sums; thus the complexity of this step is \( O(t) \).
   - 2) if \( t = 2n+1 \) we just check whether \( q(A) = q(B) = b_n(A, B) = 0 \). This is, clearly \( O(1) \).
   - 3) for \( t < 2n+1 \) and both \( A \neq 0 \) and \( B \neq 0 \), need to call two instances of the function with a prefix of even length \( t+1 \) and one further instance of the function with prefix of odd length \( t \) but rank \( n-1 \); observe that the calls of the function with prefix of length \( t+1 \) do not follow the recursive branch; thus their cost is \( O(t+1) = O(t) \).
   - 4) for \( A \neq 0 \) and \( B = 0 \), call one instance of the function with prefix of even length \( t+1 \) and rank \( n \) and one instance of the function with odd length prefix \( t-1 \) and rank \( n-1 \).
   - 5) if \( A = 0 \) and \( B = 0 \), then \( n_O(q)(S, n) \) can be obtained by a fixed formula with complexity \( O(1) \).

In particular, a run with prefix of even length \( t \) costs as much as a run with odd prefix length \( t+1 \) (and some row/column products); that is \( O(t) + \) value for odd case prefix \( t+1 \).

### Scalar products

A run with an odd prefix costs as much as 3 row/column products plus 2 runs with even length prefix (but no recursive call to the odd length case) and a run with an odd length prefix and lower rank.

We now estimate the complexity of the function \( n(q)(S, n) \) for odd prefix length. Observe that in the worst case we need to determine all of the values \( n_q(q)(G_t, n-1), \ldots, n_q(q)(G_t, n-i) \) with \( i = 1, \ldots, \frac{n-1}{2} \). Each call to \( n_q(q)(G_t, n-i) \) requires also two calls to \( n_E(q)(G_{t+1}, n) \); however these calls do not recur to a further call to \( n_q(q) \). Thus the total cost, say \( \kappa(n(q)(G_t, n)) \) is
\[ \kappa(n(q)(G_t, n)) = \sum_{i=1}^{n-(t-1)/2} (\kappa(n_q(q)(G_t, n-i)) + 2\kappa(n_q(q)(G_{t+1}, n-i))). \]

Observe now that

- \( n_q(q)(G_t, (t-1)/2) \) costs just \( O(t) \); the next step \( n_q(q)(G_t, (t-1)/2) + \) costs \( O(t) \) plus a call to \( n_q(q)(G_t, (t-1)/2) \), and so on. Thus, the overall cost is \( O(t^2) \approx O(n^2) \).
- the cost of \( n_E(q) \) is maximum when \( n(q) \) is called; in this case the complexity is bound by \( O(n^2) \).

In summary, the complexity of \( n(q) \) is \( O(n^2) \).

In order to estimate the cost of the actual enumerative encoding, observe that we would require at most \( q^2 - 1 \) calls of the function \( n(q) \) per column of \( G \). As there are \( 2n+1 \) possible columns, the overall complexity turns out to be \( O(q^2 n^3) \).

Conversely, to decode a line given an index requires to test at most \( q^2 - 1 \) candidate columns for each position; thus the cost is once more \( O(q^2 n^3) \).

### IV. Application to Polar Grassmann codes

We now apply the enumeration techniques discussed in the previous sections to efficiently implement the linear codes.
\( \mathcal{P}_{n,2} \).

A. Encoding

As before, let \( V \) be a vector space of dimension \( 2n + 1 \) over \( \mathbb{F}_q \) and take \( (e_1, \ldots, e_{2n+1}) \) to be one of its bases.

It is well known that the dual \((\Lambda^2 V)^*\) of the vector space \( \Lambda^k V \) is isomorphic to \( \Lambda^{2n+1-k} V \). We recall the following universal property of the \( k \)th exterior power of a vector space.

**Theorem 3** ([1] Theorem 14.23). Let \( V, U \) be two vector spaces over the same field. A map \( f : V^k \to U \) is alternating \( k \)-linear if, and only if, there is a linear map \( \overline{f} : \Lambda^k V \to U \) with \( \overline{f}(v_1 \wedge v_2 \wedge \cdots \wedge v_k) = f(v_1, v_2, \ldots, v_k) \). The map \( \overline{f} \) is uniquely determined.

In particular, for \( k = 2 \) and \( U = \mathbb{F}_q \) we see that any linear functional on \( \Lambda^2 V \) corresponds to a bilinear alternating form on \( V \).

In general, \( \mathcal{P}_{n,2} \) turns out to be isomorphic, as a vector space, to a quotient \((\Lambda^2 V)^*/W\) where \( W \) consists of all elements \( f : \Lambda^2 V \to \mathbb{F}_q \) which are identically zero on \( \varepsilon_2(\Lambda^2 V) \).

For \( q \) odd, \( \dim \mathcal{P}_{n,2} = \dim(\varepsilon_2(\Delta_{n,2})) = \binom{2n+1}{2} = \dim \Lambda^2 V \); thus, in this case, \( W = \{0\} \) and \( \mathcal{P}_{n,2} \) is indeed isomorphic to the vector space \( (\Lambda^2 V)^* \). We shall now restrict ourselves to this case.

Recall that each element \( \zeta \in (\Lambda^2 V)^* \) can be represented by the \((2n+1) \times (2n+1)\) antisymmetric matrix \( M \) whose entries are \( m_{ij} = \zeta(e_i \wedge e_j) \). In particular, given a message \( m = (m_1, \ldots, m_{2n+1}) \), consider the matrix \( M = M_0 - M_0^T \) where

\[
M_0 = \begin{pmatrix}
0 & m_1 & m_2 & \cdots & m_{2n} \\
0 & 0 & m_{2n+1} & \cdots & m_{4n-1} \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
0 & \cdots & 0 & m_n & m_{n+1} \\
0 & \cdots & \cdots & 0 & 0
\end{pmatrix}.
\]

This matrix defines an alternating form \( \zeta_M : V \times V \to \mathbb{F}_q \). Let the corresponding codeword \( c = (c_1, \ldots, c_N) \) be

\[
c_i := \zeta_M(G^{(i)}_1, G^{(i)}_2) = G^{(i)}_1 M G^{(i)}_2^T
\]

where \( \iota(G^{(i)}) = i \) and \( G^{(i)}_1, G^{(i)}_2 \) are the two rows of \( G^{(i)} \). It is thus possible to encode \( m \) without having to resort to the whole generator matrix of the code.

B. Decoding

Following the same approach as in the previous subsection, in order to extract a message \( m \) from a codeword \( c \), it is enough to determine a set \( I \) of information positions in \( c \). That is to say, we need a set of \( K := n(2n + 1) \) indexes corresponding to independent points in \( \Delta_{n,2} \). Then, the alternating form \( \zeta_M \) whose associated matrix is \( M \) can be recovered by interpolation, by imposing

\[
\zeta(G^{(i)}_1, G^{(i)}_2) = G^{(i)}_1 M G^{(i)}_2^T = c_i
\]

as \( i \) varies in \( I \) and solving the corresponding linear system.

C. Error correction

The area of locally decodable codes has received much attention in recent years; we refer to [20], [21] for surveys on the topic. In general, a code is locally decodable if it is able to recover a given component of a message \( m \) with probability larger than \( 1/2 \) querying just a fixed number of components of a received vector \( r \), provided that not too many errors occurred; see [22]. Clearly, a form of local decodability is essential if codes with low data rate have to be practical.

Our setting here shall be slightly different, as we will work directly on the codeword; in other words, suppose \( r \in \mathbb{F}_q^N \) and take \( 0 \leq i < N \). If there exists \( c \in \mathcal{P}_{n,2} \) with \( d(r, c) < \delta \), where \( \delta < d \) is a fixed constant, we want an algorithm to extract the value \( c_i \), with a bounded number of queries.

We can proceed as follows: fix an index \( i \) and let \( \ell \) be the line contained in \( Q_{2n}(q) \) such that \( \iota(\ell) = i \). Write

\[
\Sigma_i := \{ \pi : \ell \subseteq \pi \subseteq Q_{2n}(q), \dim \pi = 3 \}
\]

for the set of all totally singular planes containing \( \ell \) (we remark that we used vector dimension throughout the paper).

It is easy to see that there is a bijection between the elements of \( \Sigma_i \) and the points of a \( Q_{2n-4}(q) \). In particular, \#\( \Sigma_i \) is alternating \( \frac{(q^{2n-4} - 1)}{(q-1)} \). Observe also that if \( r \in \pi_1 \) and \( s \in \pi_2 \), with \( \pi_1, \pi_2 \in \Sigma_i \), \( \pi_1 \neq \pi_2 \) and \( r, s \neq \ell \), then \( r \) and \( s \) determine different conditions on the matrix \( M \), that is to say the alternating form \( \zeta_M \).

To recover the value of \( c_i \), we use a majority algorithm on the values of \( \zeta_M|_{\pi} \) with \( \pi \in \Sigma_i \).

More in detail, proceed as follows:

1) Choose a plane \( \pi \in \Sigma_i \);

2) for any two lines \( r, s \in \pi \), with \( r \neq s \) and \( r, s \neq \ell \), determine the alternating form \( \phi_{r,s} \) agreeing with \( c_i \) on \( r \) and \( s \); this corresponds to a \( 3 \times 3 \) antisymmetric matrix. With a slight abuse of notation we shall write \( \phi_{r,s}(\ell) \) for the value of the form \( \phi_{r,s}(\ell_1, \ell_2) \), where \( \ell_1 \) and \( \ell_2 \) are the two rows of the matrix representing the line \( \ell \) in RREF form.

3) If \( \phi_{r,s}(\ell) = c_i \), then accept the value \( c_i \); otherwise, try to determine a new form \( \phi_{r',s'} \) using different lines \( r' \) and \( s' \); iterate, if needed, over all of the possible \( (q^2 + q) \) pairs of lines of \( \pi \) different from \( \ell \). If there is a clear majority, assign to \( c_i \) the value determined by these computations.

4) Observe that the above procedure can correct up to \( \left\lfloor \frac{(q^2 + q - 2)}{4} \right\rfloor \) errors. If more are required, then choose different planes in \( \Sigma_i \) and repeat the algorithm, ultimately assigning to \( c_i \) the value most planes agree on.

**References**


