Math.Comput.Sci. (2023) 17:20
https://doi.org/10.1007/s11786-023-00566-7

Mathematics in Computer Science

Calculating the Minimum Distance of a Toric Code via Algebraic Algorithms

Fadime Baldemir · Mesut Şahin

Received: 16 October 2022 / Accepted: 10 May 2023 / Published online: 29 July 2023

Abstract Toric codes are examples of evaluation codes. They are produced by evaluating homogeous polynomials of a fixed degree at the $\mathbb{F}_q$-rational points of a subset Y of a toric variety X. These codes reveal how algebraic geometry and coding theory are interrelated. The minimum distance of a code is the minimum number of nonzero entries in the codewords of the code. Let $I(Y)$ be the ideal generated by all homogeneous polynomials vanishing at all the points of Y, which is also known as the vanishing ideal of Y. We give three algebraic algorithms computing the minimum distance by using commutative algebraic tools such as the multigraded Hilbert polynomials of ideals obtained from $I(Y)$ and zero divisors f of $I(Y)$, and primary decomposition of $I(Y)$, for finding a homogeneous polynomial f among all homogeneous polynomials of the same degree which has the maximum number of roots on Y.

Keywords Minimum distance · Toric code · Hilbert function

Mathematics Subject Classification 14G50 · 14M25 · 13D40 · 11T71 · 94B27

1 Introduction

A toric variety can be constructed from a nice combinatorial object called fan. Every cone σ in a fan corresponds to an affine toric variety U_σ and intersection of two cones σ_1, σ_2 corresponds to the affine toric variety $U_{\sigma_1 \cap \sigma_2}$ lying in both U_{σ_1} and U_{σ_2}. All the affine toric varieties U_{σ_1} and U_{σ_2} are glued together along the affine toric variety $U_{\sigma_1 \cap \sigma_2}$ to obtain the abstract normal toric variety of the fan. If all the cones in the fan are generated by linearly independent vectors, the fan is called simplicial.

F. Baldemir (✉)
Department of Mathematics, Çankırı Karatekin University, 18100 Çankırı, Turkey
e-mail: fadimeozkan@karatekin.edu.tr

F. Baldemir
Department of Mathematics, Middle East Technical University, 06800 Ankara, Turkey

M. Şahin
Department of Mathematics, Hacettepe University, 06800 Ankara, Turkey
e-mail: mesut.sahin@hacettepe.edu.tr

Let $\Sigma \subseteq \mathbb{R}^n$ be a complete simplicial fan with rays $\rho_1, \ldots, \rho_r$ and $X := X_\Sigma$ be the corresponding n dimensional smooth projective toric variety with Picard group isomorphic to $\mathbb{Z}^d$ over the algebraic closure $\mathbb{K} = \overline{\mathbb{F}}_q$ of a finite field $\mathbb{F}_q$, where the rank $d = r - n > 0$.

Let $v_1, \ldots, v_r$ be the generators of the rays of the cones in the fan Σ and ϕ be the matrix whose rows are these generators of the rays. We get the matrix β in the following exact sequence by applying the Smith Normal Form Algorithm (see [2] and references therein) to ϕ:

$$\mathfrak{B} : 0 \longrightarrow \mathbb{Z}^n \xrightarrow{\phi} \mathbb{Z}^r \xrightarrow{\beta} \mathcal{A} \longrightarrow 0.$$

Here the group $\mathcal{A} \cong \mathbb{Z}^r/\phi(\mathbb{Z}^n)$ is isomorphic to the Picard group of X. Applying $Hom(-, \mathbb{K}^*)$ functor to the sequence $\mathfrak{B}$ gives the following short exact sequence:

$$\mathfrak{B}^* : 1 \longrightarrow G \xrightarrow{i} (\mathbb{K}^*)^r \xrightarrow{\pi} T_X \longrightarrow 1$$

with $\pi : (\xi_1, ..., \xi_r) \to (\xi^{u_1}, ..., \xi^{u_n})$ where $u_1, ..., u_n$ are the columns of ϕ.

Let $S = \mathbb{F}_q[x_1, ..., x_r] = \bigoplus_{\alpha \in \mathcal{A}} S_\alpha$ be the multigraded polynomial ring which is also known as the Cox ring of X graded using the columns of the matrix β, i.e. $deg_{\mathcal{A}}(x_j) = \beta_j$ is the j-th column of β, for $j = 1, ..., r$. The irrelevant ideal of S is the monomial ideal

$$B = \langle x^{\hat{\sigma}} : \quad \sigma \in \Sigma \rangle, \quad \text{where } x^{\hat{\sigma}} = \prod_{\rho_i \notin \sigma} x_i.$$

The main feature of X we use here is that we can represent points on our toric variety X using homogeneous coordinates thanks to the Geometric Invariant Theory quotient representation

$$X \cong \mathbb{K}^r \setminus V(B))/G, \text{ where } G = \mathrm{Ker}(\pi) \text{ and } \mathbb{K} = \overline{\mathbb{F}}_q,$$

due to Cox, see [6]. Therefore, every point of X is identified with an orbit of the following type:

$$[P] = G \cdot P = [p_1 : \cdots : p_r] \text{ for } P \in \mathbb{K}^r \setminus V(B).$$

When the point is an $\mathbb{F}_q$-rational point, one can choose a representative $P \in \mathbb{F}_q^r$. Traditionally, the set of $\mathbb{F}_q$-rational points of X is denoted by $X(\mathbb{F}_q)$. Since we focus only on a subset Y of the $\mathbb{F}_q$-rational points, we abuse the notation and simply use $Y \subseteq X$ instead of $Y \subseteq X(\mathbb{F}_q)$ for the sake of the notation, throughout the paper.

For a fixed degree α in the semigroup $\mathbb{N}\beta$ which is generated by $\beta_1, ..., \beta_r$, and a subset $Y = \{[P_1], ..., [P_N]\} \subseteq X$ with cardinality N, the evaluation map is defined as follows:

$$ev_Y : S_\alpha \mapsto \mathbb{F}_q^N, \quad F \mapsto (F(P_1), \ldots, F(P_N)).$$

The image of S_α under ev_Y is called a (generalized) toric code which is denoted by $\mathcal{C}_{\alpha,Y}$, see [19] for a survey. By definition, the size of Y is the *length* of the code and the *dimension* $\dim_{\mathbb{F}_q} \mathcal{C}_{\alpha,Y}$ of the code is the dimension of $\mathcal{C}_{\alpha,Y}$ as a vector space over $\mathbb{F}_q$. The number of nonzero entries in a codeword is called its *weight* and the *minimum distance* δ is obtained by calculating the smallest weight among all nonzero codewords.

It is clear that the kernel of the linear map ev_Y is the degree α part $I_\alpha(Y)$ of the β-graded or homogeneous *vanishing ideal* $I(Y)$ of Y, which is defined to be the ideal generated by homogeneous polynomials vanishing at all the points of Y. Due to this relation, the dimension $dim_{\mathbb{F}_q}\mathcal{C}_{\alpha,Y}$ equals the value $H_{I(Y)}(\alpha) := \dim S_\alpha - \dim I_\alpha(Y)$ of the multigraded *Hilbert function* $H_{I(Y)}$ of $I(Y)$. It is known that for sufficiently large values of α, the function $H_{I(Y)}$ agrees with a polynomial $P_{I(Y)}$ known as the multigraded *Hilbert polynomial* of $I(Y)$. If I is the B-saturated ideal corresponding to the set of ℓ points on a smooth projective toric variety, then the Hilbert polynomial of I is just the constant $P_I(t) = \ell$ (see Example 4.12 in [12]). Therefore, the length of the code $\mathcal{C}_{\alpha,Y}$ is $P_{I(Y)}(t) = N$.

Toric codes attracted attentions of mathematicians nearly about two decades due to their rich combinatorial nature showcasing some linear codes with best parameters, see [14] for a very recent paper and the references therein for more details. They have also applications in information theory, for example Hansen [8] created a secret sharing scheme with strong multiplication by using toric codes on toric surfaces and used toric codes which are obtained from a special toric variety called Hirzebruch surface to construct new quantum codes [9]. There are lots of

champion toric codes having the largest minimum distance among all known linear codes with equal block length and dimension, see [4,5,11]. Toric codes are indeed generalizations of (generalized) Reed-Solomon codes which can be obtained by choosing $X = \mathbb{P}^1$, $Y \subseteq X(\mathbb{F}_q)$ and $\alpha < q$, which are MDS codes [9,10].

An algebraic algorithm using the vanishing ideal $I(Y)$ has been given by Martinez-Bernal, Pitones and Villarreal in [13] to compute the minimum distance of a toric code defined on a projective space $X = \mathbb{P}^n$. They pointed out that the algorithm is more interesting theoretically rather than computationally as other existing algorithms relying on the generating matrix of the code perform better. The motivation for the present article is to discuss possible generalizations of their algorithm to a more general smooth projective toric variety. We offer 3 algorithms in Sect. 2 for this purpose and share Macaulay 2 procedures to implement them in Sect. 3. In Sect. 4, we present experimental results obtained using these algorithms and discuss their performance.

2 Theoretical Results and Algorithms

In this section we give our main results leading to algebraic algorithms for computing the minimum distance of the code.

Let X be a toric variety over the finite field $\mathbb{F}_q$ and $Y \subseteq X$ be a subset of the $\mathbb{F}_q$-rational points. For a homogeneous polynomial $f \in S_\alpha$, $V_{X,Y}(f)$ denotes the subset of Y which consists of the roots of f, i.e.

$$V_{X,Y}(f) = \{[P] \in Y : f(P) = 0\}.$$

In the first step, we make the following simple observation whose proof is included for the sake of the reader, and was already given in [13] for the case where X is a projective space:

Lemma 1 *Let $Y \subseteq X$. Then, the minimum distance of the corresponding code is*

$$\delta(\mathcal{C}_{\alpha,Y}) = N - max\{\left|V_{X,Y}(f)\right| : f \in S_\alpha \setminus I_\alpha(Y)\}.$$

Proof By definition, the weight of the codeword $ev_Y(f)$ is the number of non-zero components, which is nothing but $N - \left|V_{X,Y}(f)\right|$. A codeword $ev_Y(f)$ is zero if and only if $f \in I_\alpha(Y)$. Thus, a non-zero codeword with the minimum weight corresponds to a polynomial in $S_\alpha \setminus I_\alpha(Y)$ with the maximum number of zeroes. □

We recall basic notions and definitions of commutative algebra such as minimal prime, associated prime, primary decomposition etc.

Definition 1 (Definition 5 in [3]) *A primary decomposition* of an ideal $\mathfrak{a} \subset S$ is a decomposition

$$\mathfrak{a} = \bigcap_{i=1}^{k} \mathfrak{q}_i$$

into primary ideals $\mathfrak{q}_i \subset S$. The decomposition is called *minimal* if the corresponding prime ideals $\mathfrak{p}_i = rad(\mathfrak{q}_i)$ are pairwise different and the decomposition is unshortenable; the latter means that $\cap_{i \neq j} \mathfrak{q}_i \not\subset \mathfrak{q}_j$ for all $j = 1, \ldots, k$.

By Theorem 8 in [3], the prime ideals $\mathfrak{p}_i = rad(\mathfrak{q}_i)$ are uniquely determined when the primary decomposition above is minimal. These are of the form $rad((\mathfrak{a} : f))$ for some $f \in S$ and deserve a special name as we see below:

Definition 2 (*Definition and Proposition 9 in* [3]) Let $\mathfrak{a} \subset S$ be an ideal. The set of all prime ideals in S that are of type $rad((\mathfrak{a} : f))$ for some $f \in S$ is denoted by $Ass(\mathfrak{a})$; its members are called the *prime ideals associated to* $\mathfrak{a}$.

Definition 3 (*Definition 11 in* [3]) Given any ideal $\mathfrak{a} \subset S$, the subset of all prime ideals that are minimal in $Ass(\mathfrak{a})$ is denoted by $Ass'(\mathfrak{a})$ and its members are called the *isolated* prime ideals associated to $\mathfrak{a}$. All other elements of $Ass(\mathfrak{a})$ are said to be *embedded* prime ideals.

Definition 4 (*Page 63 in* [3]) For an arbitrary ideal $\mathfrak{a} \subset S$,

$$\mathcal{Z}(\mathfrak{a}) = \bigcup_{f \in S \setminus \mathfrak{a}} (\mathfrak{a} : f) = \{z \in S : zf \in \mathfrak{a} \text{ for some } f \in S \setminus \mathfrak{a}\}$$

is called the set of *zero divisors modulo* $\mathfrak{a}$ in S. Indeed, an element $z \in S$ belongs to $\mathcal{Z}(\mathfrak{a})$ if and only if its residue class $\bar{z} \in S/\mathfrak{a}$ is a zero divisor. Furthermore, if a power z^n of some element $z \in S$ belongs to $\mathcal{Z}(\mathfrak{a})$, then z itself must belong to $\mathcal{Z}(\mathfrak{a})$ and therefore

$$\mathcal{Z}(\mathfrak{a}) = \bigcup_{f \in S \setminus \mathfrak{a}} rad((\mathfrak{a} : f)).$$

With these in mind it is now easy to prove the following useful fact as in [18, Theorem 5.1(iii)].

Lemma 2 *Let $Y \subseteq X$ be a finite subset. Then, the minimal primary decomposition of $I(Y)$ is*

$$I(Y) = \bigcap_{[P] \in Y} I([P]).$$

Furthermore, we have

$$\mathcal{Z}(I(Y)) = \bigcup_{f \in S \setminus I(Y)} rad((I(Y) : f)) = \bigcup_{[P] \in Y} I([P]).$$

By using Lemmas 1 and 2 above we obtain the following result.

Theorem 1 *Let $Y \subseteq X$. Then, the minimum distance of the corresponding code is*

$$\begin{aligned}\delta(\mathcal{C}_{\alpha,Y}) &= N - max\{|\{\mathfrak{p} \in Ass(I(Y)) : f \in \mathfrak{p}\}| : f \in \mathcal{Z}(I(Y))\},\\ &= P_{I(Y)} - max\{|\{\mathfrak{p} \in Ass(I(Y)) : f \in \mathfrak{p}\}| : f \in \mathcal{Z}(I(Y))\},\end{aligned}$$

where $P_{I(Y)}$ is the Hilbert Polynomial of the ideal $I(Y)$.

Proof By Lemma 1, we have the formula:

$$\delta(\mathcal{C}_{\alpha,Y}) = N - max\{\left|V_{X,Y}(f)\right| : f \in S_\alpha \setminus I_\alpha(Y)\}.$$

$[P] \in V_{X,Y}(f)$ if and only if $f(P) = 0$ and $[P] \in Y$ if and only if $f \in I(P)$ and $[P] \in Y$. Thus, the number of zeroes $[P]$ of f is exactly the number of associated primes $I(P)$ in the minimal primary decomposition of $I(Y)$ containing f. It follows from Lemma 2 that $Ass(I(Y)) = Ass'(I(Y)) = \{I(P) : [P] \in Y\}$. Hence, we have

$$\left|V_{X,Y}(f)\right| = |\{\mathfrak{p} \in Ass(I(Y)) : f \in \mathfrak{p}\}|.$$

In particular, $\left|V_{X,Y}(f)\right| > 0$ implies that $f \in \mathcal{Z}(I(Y))$, completing the proof. □

As a consequence, we obtain the following algorithm which calculates the minimum distance of a code obtained from a smooth projective toric variety:

Algorithm 1 Calculating the minimum distance of a toric code.

Input A prime power q, a toric variety X over the field $\mathbb{F}_q$, a degree α together with the vanishing ideal $I(Y)$ of Y.
Output The minimum distance $\delta(\mathcal{C}_{\alpha,Y})$.

1: Find a basis B_α of the vector space $M_\alpha = S_\alpha / I_\alpha(Y)$.
2: Form the set M_α by taking $\mathbb{F}_q$-linear combinations of the elements of B_α.
3: Determine zero-divisors $f \in M_\alpha$ by checking if $I(Y) : f \neq I(Y)$.
4: Find the primary decomposition of $I(Y)$.
5: Return $\delta(\mathcal{C}_{\alpha,Y}) = P_{I(Y)} - max\{|\{\mathfrak{p} \in Ass(I(Y)) : f \in \mathfrak{p}\}| : f \in \mathcal{Z}(I(Y))\}$.

Next, we give the second result leading to an alternative algorithm for computing the minimum distance.

Theorem 2 *Let $Y \subseteq X$. Then, the minimum distance of the corresponding code is*

$$\delta(\mathcal{C}_{\alpha,Y}) = N - max\{P_{I(V_{X,Y}(f))} : f \in S_\alpha \setminus I_\alpha(Y) \text{ is a zero-divisor}\},$$

where $P_{I(V_{X,Y}(f))}$ is the Hilbert Polynomial of the ideal $I(V_{X,Y}(f))$, which can be obtained as follows:

$$I(V_{X,Y}(f)) = \bigcap_{\mathfrak{p} \in Ass(I(Y)) \text{ and } f \in \mathfrak{p}} \mathfrak{p}.$$

Proof By Lemma 1, we have the formula:

$$\delta(\mathcal{C}_{\alpha,Y}) = N - max\{\left|V_{X,Y}(f)\right| : f \in S_\alpha \setminus I_\alpha(Y)\}.$$

Since $I(V_{X,Y}(f))$ is the B-saturated ideal corresponding to the set of $\left|V_{X,Y}(f)\right|$ points on a smooth projective toric variety X, it follows that the Hilbert polynomial of $I(V_{X,Y}(f))$ is just the constant $P_{I(V_{X,Y}(f))}(t) = \left|V_{X,Y}(f)\right|$ as indicated in Example 4.12 of [12]. When f is not a zero-divisor, then $f \notin I(P)$ by Lemma 2. So it has no zeroes and thus the weight of the corresponding codeword will be N, the maximum possible.

Finally, by Lemma 2, we have

$$I(V_{X,Y}(f)) = \bigcap_{[P] \in V_{X,Y}(f)} I(P) \text{ and } I(Y) = \bigcap_{[P] \in Y} I(P) = \bigcap_{\mathfrak{p} \in Ass(I(Y))} \mathfrak{p}.$$

As $[P] \in V_{X,Y}(f)$ if and only if $\mathfrak{p} \in Ass(I(Y))$ and $f \in \mathfrak{p}$, the proof follows. □

Algorithm 2 Calculating the minimum distance of a toric code.

Input A prime power q, a toric variety X over the field $\mathbb{F}_q$, a degree α together with the vanishing ideal $I(Y)$ of Y.
Output The minimum distance $\delta(\mathcal{C}_{\alpha,Y})$.

1: Find a basis B_α of the vector space $M_\alpha = S_\alpha / I_\alpha(Y)$.
2: Form the set M_α by taking $\mathbb{F}_q$-linear combinations of the elements in B_α.
3: Determine zero-divisors $f \in M_\alpha$ by checking if $I(Y) : f \neq I(Y)$.
4: Find the primary decomposition of $I(Y)$.
5: Find the ideals $I(V_{X,Y}(f))$ for zero-divisors $f \in M_\alpha$.
6: Return $\delta(\mathcal{C}_{\alpha,Y}) = P_{I(Y)} - max\{P_{I(V_{X,Y}(f))} : f \text{ is a zero-divisor }\}$.

There is a third algorithm suggested by the paper [13] which was the starting point of this research. The toric variety in that paper is the special case of the projective space $X = \mathbb{P}^n$ and our aim was to understand if we can generalize it. The connection with the second and third algorithm to be given below is the difference in their last steps. One of them uses the Hilbert polynomial $P_{I(V_{X,Y}(f))}$ of the ideal $I(V_{X,Y}(f))$ whereas the other uses the Hilbert polynomial $P_{I(Y)+(f)}$ of the ideal $I(Y) + (f)$. Notice that the notion of the degree of the ideal $I(Y) + (f)$ in their paper coincides with the Hilbert polynomial $P_{I(Y)+(f)}$ by [13, Corollary 4.3] and [12, Example 4.12]. Until now, we were not able to get a generalization of [13, Corollary 4.3] valid for any toric variety X. Therefore, the correctness of the third algorithm is guaranteed only for the case where X is a projective space.

Algorithm 3 Calculating the minimum distance of a toric code.

Input A prime power q, a toric variety X over the field $\mathbb{F}_q$, a degree α together with the vanishing ideal $I(Y)$ of Y.
Output The minimum distance $\delta(\mathcal{C}_{\alpha,Y})$.

1: Find a basis of the vector space $M_\alpha = S_\alpha / I_\alpha$ for $M = S/I(Y)$.
2: Form the set M_α by taking $\mathbb{F}_q$-linear combinations of the basis elements of M_α.
3: Determine zero-divisors $f \in M_\alpha$ by checking if $I(Y) : f \neq I(Y)$.
4: Return $\delta_Y(\alpha) = P_{I(Y)} - max\{P_{I(Y)+(f)} : f \text{ is a zero-divisor }\}$.

Remark 1 It is worth briefly describing an idea used in [14,15] adapted to our notation which is relevant to the ideals considered in Algorithm 2 and 3, see the proof of [14, Lemma 5.5]. Recall that the set $reg(Y)$ consists of the elements α for which $H_{I(Y)}(\alpha) = N$. Thus, for an element $\tilde{\alpha} \in reg(Y)$, the linear map $ev_{\tilde{\alpha},Y} : S_{\tilde{\alpha}} \mapsto \mathbb{F}_q^N$ is surjective. We also have the natural projection $\mathbb{F}_q^N \mapsto \mathbb{F}_q^{N_f}$ onto $Y_f := V_{X,Y}(f)$, for any $f \in S_\alpha \backslash I_\alpha(Y)$, thus we have the following surjective map by composing them:

$$ev_{\tilde{\alpha},Y_f} : S_{\tilde{\alpha}} \mapsto \mathbb{F}_q^{N_f}$$

where $N_f = \left|Y_f\right|$. Therefore, $N_f = H_{I(Y_f)}(\tilde{\alpha})$ and $\tilde{\alpha} \in reg(Y_f)$. This means that the Hilbert polynomial N_f of $I(Y_f)$ is attained at $\tilde{\alpha}$.
Clearly, the ideal $I(Y, f) := I(Y) + (f) \subseteq I(Y_f)$ and hence we have

$$I_{\tilde{\alpha}}(Y, f) = I_{\tilde{\alpha}}(Y) + S_{\tilde{\alpha}-\alpha} \cdot (f) \subseteq I_{\tilde{\alpha}}(Y_f).$$

Thus, an upper bound for N_f is given by

$$\widetilde{N_f}(\tilde{\alpha}) = H_{I(Y)+(f)}(\tilde{\alpha}) \geq N_f = H_{I(Y_f)}(\tilde{\alpha}).$$

Letting $\widetilde{N_f}$ be the Hilbert polynomial of the ideal $I(Y, f)$, we see that $\widetilde{N_f} \geq N_f$ and hence, it follows that we have

$$N_f = \widetilde{N_f} \iff I(Y_f) = (I(Y, f) : B^\infty),$$

since Hilbert polynomial of an ideal and that of its saturation with respect to B are the same. This leads to the following claim which is proved in [13, Corollary 4.3] when X is a projective space.

Conjecture 1 *For a smooth projective toric variety, we have* $I(Y_f) = (I(Y, f) : B^\infty)$.

Remark 2 Available algorithms in computer algebra programs such as GAP, Magma or Sage require a generating matrix for the code and in order to determine that matrix one needs to give the members of the set Y. But the elements are not always given explicitly. Sometimes they are given implicitly as in

$$Y_Q = \{[\mathbf{t}^{\mathbf{q}_1} : \cdots : \mathbf{t}^{\mathbf{q}_r}] | \mathbf{t} \in (\mathbb{F}_q^*)^s\}$$

which is parameterized by the columns of a matrix $Q = [\mathbf{q}_1 \mathbf{q}_2 \cdots \mathbf{q}_r]$. In this case our algorithms can be used alternatively as the vanishing ideal needed for them can be found as described in [1].

In some cases it is even possible to expedite our first two algorithms by omitting the 4−th step by the virtue of the minimal primary decomposition in Lemma 2. For instance, in the proof of [17, Theorem 4.1], where $X = \mathbb{P}(1, w_1, \ldots, w_n)$ is a weighted projective space over the field $\overline{\mathbb{F}}_q$, the elements of the subgroup

$$Y_Q = \{[t_0 : t_1^{w_1} : \ldots : t_n^{w_n}] \mid t_i \in \mathbb{F}_q^*, \text{ for all } i = 0, \ldots, n\}$$

is given explicitly so the minimal primary decomposition of $I(Y_Q)$ is the intersection of $I(P)$ for all $P \in Y_Q$. More precisely, the set Y_Q is found to be the union of the points $[1 : \eta_1^{i_1} : \cdots : \eta_n^{i_n}]$ for $1 \leq i_1 \leq d_1, \ldots, 1 \leq i_n \leq d_n$, where $\mathbb{F}_q^* = \langle \eta \rangle$ so that the order of the generator η is $q - 1$ and the order of $\eta_i := \eta^{w_i}$ is

$$d_i = \frac{q-1}{\gcd(q-1, w_i)} \quad i = 1, \ldots, n.$$

Since the generators of the vanishing ideal is given by [17, Proposition 3.3], and the ideals $I([1 : \eta_1^{i_1} : \cdots : \eta_n^{i_n}]) = \langle x_2 - \eta_1^{i_1} x_1^{w_1}, \ldots, x_n - \eta_n^{i_n} x_n^{w_n} \rangle$ are found using [16], the Algorithms 1 and 2 can be applied to compute the minimum distance more quickly.

3 `Macaulay2` Procedures

In this section, we share some procedures needed to implement the algorithms in the previous section in `Macaulay2`. One may reach all of these codes from the link https://github.com/fdmozkan/Macaulay2-Codes/releases/tag/v1.0.0

Procedure 1 *Calculating the minimum distance of a toric code with Macaulay2 [7].*

```
i4: Balpha=basis(alpha,S/IY);
i5: N=flatten applyTable({apply(toList (set(0..q-1))^**
      (hilbertFunction(alpha,S/IY))- (set{0})^**
      (hilbertFunction(alpha,S/IY)),i-> toList i)},
                                  i-> deepSplice i);
    P= apply(#N, j-> vector flatten N_{j});
    D= for i from 0 to #P-1 list Balpha*  flatten P_{i};
    A= flatten for i from 0 to #D-1 list entries
       (flatten D_{i})#0;
    Malpha= apply(A, i->substitute(i,S));
 i6: Z=select(Malpha, f-> not quotient(IY,ideal f)==IY);
```

Let us now explain each step of the codes in `i5-i6` in detail. In step `N`, we obtain a list of K-tuples by choosing entries from $\mathbb{F}_q$ and removing the origin, where K is the dimension of $S_\alpha/I_\alpha(Y)$ or the cardinality of the set `Balpha`. In step `P`, we transform each of these K-tuples into a vector in $\mathbb{F}_q^K$. In step `D`, by using the basis of $S_\alpha/I_\alpha(Y)$, which are given as vectors in `Balpha` we make a list of polynomials representing elements in $S_\alpha/I_\alpha(Y)$ by taking their dot products with the coefficients stored as vectors in step `P`. In step `A`, we remove all parenthesis from each member of the list `D`. In step `Malpha`, we substitute each polynomial obtained in the previous step to the ring S. In step `Z`, we determine all the zero-divisors of $S_\alpha/I_\alpha(Y)$. One may possibly expedite this process by finding a direct way to form elements of `Malpha`.

For Algorithm 1 we continue with the following:

```
i7: PrIY=primaryDecomposition IY;

i8: delta=hilbertPolynomial(X,IY)
           -max apply(Z, f->#select(PrIY, i-> f%i==0))
```

The command `primaryDecomposition` used in step `i7` is part of a package designed for providing components of ideals and modules, including associated primes and primary decompositions, see [21].

As pointed out in Remark 2, one may accelerate Algorithm 1 by replacing the usage of the general purpose `primaryDecomposition` package by using Lemma 2 and by finding a set of representatives for the set Y in question.

The `hilbertPolynomial` command used in step `i8` is part of another package which computes with normal toric varieties, see [20]. But, we need to modify the content of that package as we work with a coordinate ring S over a finite field $\mathbb{F}_q$. Therefore, one has to run the command in `i2` after loading the package as in the command in `i1`.

```
i1: needsPackage ``NormalToricVarieties'';
i2: ring NormalToricVariety := PolynomialRing =>
    (cacheValue symbol ring)
    ( X -> (
        if isDegenerate X then
```

```
            error ''-- not yet implemented
                  for degenerate varieties'';
        if not isFreeModule classGroup X then
            error ''-- gradings by torsion groups
                   not yet implemented'';
        -- constructing ring
        K := X.cache.CoefficientRing;
        x := X.cache.Variable;
        r := #rays X;
        deg := entries transpose fromWDivToCl X;
        S := K (monoid [x_1..x_r, Degrees => deg]);
        S.variety = X;
        S  ) );
```

For Algorithm 2 we replace the previous commands with the following:

```
i7: PrIY=primaryDecomposition IY;
i8: IVXYf=(PrIY,f,S) -> (int := ideal (1_S) ;
           scan(PrIY, i -> if  f%i==0 then
           int=intersect(int,i)) ; int);
    Ideals=apply(Z,f->IVXYf(PrIY,f,S));
i9: delta = hilbertPolynomial(X,IY)
           - max apply(Ideals, I-> hilbertPolynomial (X,I))
```

For Algorithm 3 we use the following command instead:

```
i7: deltaTilde=hilbertPolynomial(X,IY)
         -max apply(Z, f-> hilbertPolynomial(X,IY+ideal (f)))
```

4 Experimental Results

In this section, we present an example illustrating the speed of our algorithms implemented and run in Macaulay 2.

Example 1 Let X be the Hirzebruch Surface H_2 over $K = \mathbb{F}_7$. The first exact sequence above becomes

$$\mathfrak{P} : 0 \longrightarrow \mathbb{Z}^2 \xrightarrow{\phi} \mathbb{Z}^4 \xrightarrow{\beta} \mathbb{Z}^2 \longrightarrow 0$$

where

$$\phi = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 2 & -1 \end{bmatrix}^T \quad \text{and} \quad \beta = \begin{bmatrix} 1 & -2 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}.$$

Thus, the Cox ring $S = K[x_1, x_2, x_3, x_4]$ of X is graded via

$$\deg_{\mathcal{A}}(x_1) = \deg_{\mathcal{A}}(x_3) = (1, 0), \deg_{\mathcal{A}}(x_2) = (-2, 1), \deg_{\mathcal{A}}(x_4) = (0, 1).$$

Consider the subset $Y \subset X$ with $I(Y) = (x_3^2 - x_1^2, x_4^3 - x_3^6 x_2^3)$.

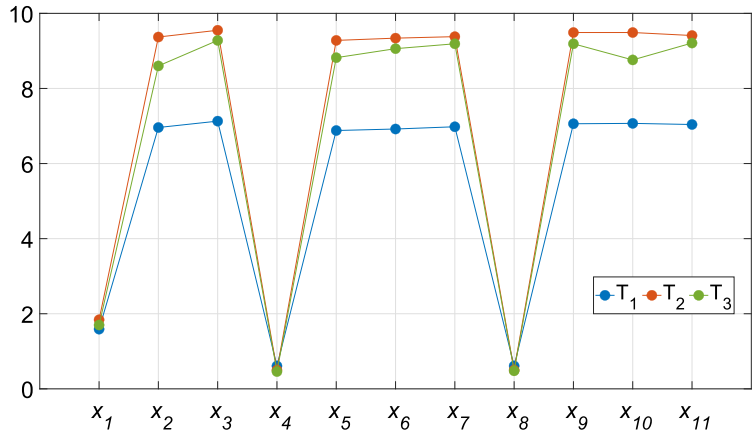
Our inputs for the example are as follows:

Table 1 Comparison of 3 algorithms.

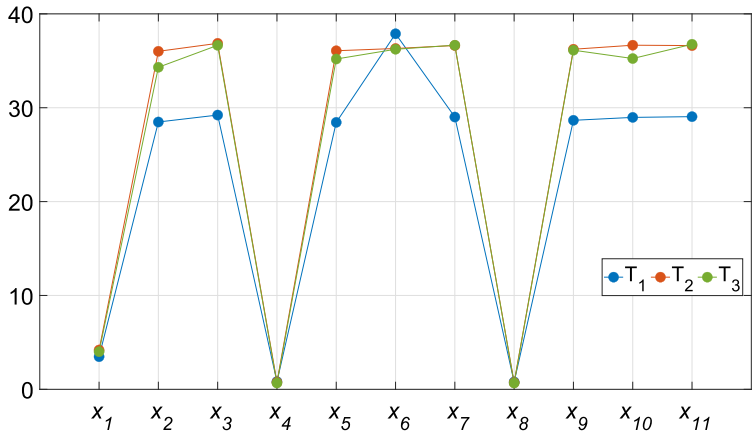
q	α	δ	$\widetilde{\delta}$	T_1	T_2	T_3
5	(0, 1)	2	2	**1.59**	1.84	1.7
	(0, 2)	1	1	**6.96**	9.37	8.6
	(0, 3)	1	1	**7.13**	9.55	9.28
	(1, 0)	2	2	0.61	0.50	**0.46**
	(1, 1)	1	1	**6.88**	9.28	8.82
	(1, 2)	1	1	**6.92**	9.34	9.06
	(1, 3)	1	1	**6.98**	9.38	9.19
	(2, 0)	2	2	0.61	0.50	**0.48**
	(2, 1)	1	1	**7.06**	9.49	9.19
	(2, 2)	1	1	**7.07**	9.49	8.76
	(2, 3)	1	1	**7.04**	9.41	9.21
7	(0, 1)	2	2	**3.48**	4.19	4.01
	(0, 2)	1	1	**28.48**	36.01	34.31
	(0, 3)	1	1	**29.22**	36.86	36.65
	(1, 0)	2	2	0.82	0.75	**0.67**
	(1, 1)	1	1	**28.44**	36.07	35.19
	(1, 2)	1	1	37.88	36.31	**36.23**
	(1, 3)	1	1	**29.01**	36.63	36.66
	(2, 0)	2	2	0.79	0.72	**0.67**
	(2, 1)	1	1	**28.66**	36.23	36.13
	(2, 2)	1	1	**28.97**	36.66	35.24
	(2, 3)	1	1	**29.05**	36.62	36.76
11	(0, 1)	2	2	**13.07**	15.72	15.17
	(0, 2)	1	1	**169.36**	200.94	199.00
	(0, 3)	1	1	**169.90**	202.16	205.76
	(1, 0)	2	2	1.35	1.34	**1.22**
	(1, 1)	1	1	**165.72**	198.50	198.90
	(1, 2)	1	1	**168.26**	200.98	205.41
	(1, 3)	1	1	**170.23**	203.26	208.31
	(2, 0)	2	2	1.37	1.36	**1.27**
	(2, 1)	1	1	**166.75**	199.71	204.3
	(2, 2)	1	1	**168.38**	201.17	198.45
	(2, 3)	1	1	**169.33**	202.11	206.96

```
i3: q=7; alpha={0,1};
    X=hirzebruchSurface(2, CoefficientRing =>
                        GF(q,Variable=>t) );
    S=ring X;
    IY=ideal ((x_3)^(2)-(x_1)^(2),
              (x_4)^(3)-(x_3)^(2*(3))*x_2^(3));
```

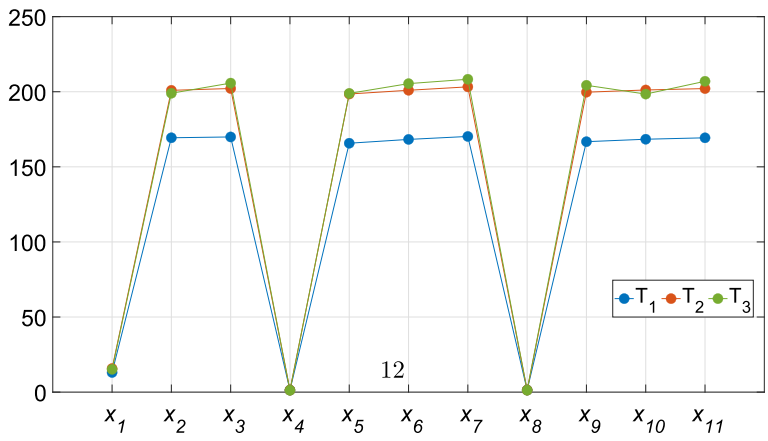
Using Procedure 1, we compute the minimum distance of $C_{\alpha,Y}$ *to be* 3 *for all the three algorithms.*



(a) $q = 5$



(b) $q = 7$



(c) $q = 11$

Fig. 1 Comparison of the algorithms

Before we conclude the paper, we will give a table and a figure to compare the performance of the three algorithms on this example for various values of α.

In Table 1, δ is the minimum distance found using the first two algorithms and $\widetilde{\delta}$ is the minimum distance found using the third algorithm. Moreover, T_i stands for the time used by the Algorithm i, for each $i = 1, 2, 3$. The smallest CPU time is given in boldface type.

See Fig. 1 for the graphical comparison of the three algorithms according to their CPU time. In the graph x_i corresponds to the i-th value of α in Table 1 and y_i corresponds to the CPU time used by the relevant algorithm. It seems that the first algorithm is almost always faster than the others. This is not surprising when it comes to a comparison with the second algorithm, because of the differences of the algorithms in Procedure 1. The first one avoids computations of the ideals $I(V_{X,Y}(f))$ in step `i8` of Procedure 1 for Algorithm 2, for all f, and replaces the computations of the `hilbertPolynomials` for all these ideals in step `i9` by a rather basic command in `i8` of Procedure 1 for Algorithm 1.

5 Conclusion

In the literature, there is an algorithm for computing the minimum distance of an evaluation code obtained from a projective space which heavily makes use of computational commutative algebra. Until now, there is no direct generalization to a more general smooth toric variety. We offered two alternatives that work correctly for a general smooth toric variety and illustrate it via an example. We also share a table showing the computation time for these three algorithms in order to give a clue as to their complexity.

All the algorithms suffer from listing all the zero divisors f and computing with them even if they have just 1 root, which is clearly unnecessary. This is overcome in certain cases by providing an upper bound on the number of roots and demonstrating a concrete polynomial meeting that upper bound, see e.g. [1,15]. One may possibly improve these algorithms by proving a result which helps eliminate zero divisors with less zeroes.

Acknowledgements Fadime Baldemir is supported by TÜBİTAK (TÜBİTAK-2211-A National PhD Scholarship Program). Mesut Şahin is supported by TÜBİTAK Project No:119F177. This article is part of Fadime Baldemir's PhD thesis.

References

1. Baran, E., Sahin, M.: On parameterized Toric codes. Appl. Algebra Eng. Commun. Comput. (2023). https://doi.org/10.1007/s00200-021-00513-8
2. Birmpilis, S., Labahn, G., Storjohann, A.: A fast algorithm for computing the Smith normal form with multipliers for a nonsingular integer matrix. J. Symbolic Comput. **116**, 146–182 (2023). https://doi.org/10.1016/j.jsc.2022.09.002
3. Bosch, S.: Algebraic Geometry and Commutative Algebra, Universitext, p. 504. Springer, London (2013). https://doi.org/10.1007/978-1-4471-4829-6
4. Brown, G., Kasprzyk, A.M.: Seven new champion linear codes. LMS J. Comput. Math. **16**, 109–117 (2013). https://doi.org/10.1112/S1461157013000041
5. Brown, G., Kasprzyk, A.M.: Small polygons and Toric codes. J. Symb. Comput. **51**, 55–62 (2013). https://doi.org/10.1016/j.jsc.2012.07.001
6. Cox, D.A., Little, J.B., Schenck, H.K.: Toric Varieties. Graduate Studies in Mathematics, vol. 124, p. 841. American Mathematical Society, Providence, RI (2011). https://doi.org/10.1090/gsm/124
7. Grayson, D.R., Stillman, M.E.: Macaulay2, a software system for research in algebraic geometry. Available at http://www.math.uiuc.edu/Macaulay2/
8. Hansen, J.P.: Secret sharing schemes with strong multiplication and a large number of players from toric varieties. In: Arithmetic, Geometry, Cryptography and Coding Theory. Contemporary Mathematics, vol. 686, pp. 171–185 (2017). https://doi.org/10.1090/conm/686/13783
9. Hansen, J.P.: Toric surfaces, linear and quantum codes secret sharing and decoding. Facets of Algebraic Geometry: Volume 1: A Collection in Honor of William Fulton's 80th Birthday **472**, 371 (2022)
10. Lachaud, G.: The parameters of projective Reed-Muller codes. Discrete Math. **81**(2), 217–221 (1990). https://doi.org/10.1016/0012-365X(90)90155-B
11. Little, J.B.: Remarks on generalized Toric codes. Finite Fields Appl. **24**, 1–14 (2013). https://doi.org/10.1016/j.ffa.2013.05.004

12. Maclagan, D., Smith, G.: Uniform bounds on multigraded regularity. J. Algebr. Geom **14**(1), 137–164 (2005). https://doi.org/10.1090/S1056-3911-04-00385-6
13. Martinez-Bernal, J., Pitones, Y., Villarreal, R.H.: Minimum distance functions of graded ideals and reed-muller-type codes. J. Pure Appl. Algebra **221**(2), 251–275 (2017). https://doi.org/10.1016/j.jpaa.2016.06.006
14. Nardi, J.: Projective Toric codes. Int. J. Number Theory **18**(01), 179–204 (2022). https://doi.org/10.1142/S1793042122500142
15. Nardi, J.: Algebraic geometric codes on minimal Hirzebruch surfaces. J. Algebra **535**, 556–597 (2019). https://doi.org/10.1016/j.jalgebra.2019.06.022
16. Şahin, M.: Computing vanishing ideals for Toric codes. arXiv preprint arXiv:2207.01061 (2022)
17. Şahin, M., Yayla, O.: Codes on subgroups of weighted projective tori. arXiv preprint arXiv:2210.07550 (2022)
18. Şahin, M.: Toric codes and lattice ideals. Finite Fields Appl. **52**, 243–260 (2018). https://doi.org/10.1016/j.ffa.2018.04.007
19. Şahin, M.: In: Barucci, V., Chapman, S., D'Anna, M., Fröberg, R. (eds.) Lattice Ideals, Semigroups and Toric Codes, pp. 285–302. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-40822-0_16
20. Smith, G.G.: NormalToricVarieties: A *Macaulay2* package. Version 1.9. A *Macaulay2* package available at https://github.com/Macaulay2/M2/blob/master/M2/Macaulay2/packages/NormalToricVarieties.m2
21. Stillman, M., Yackel, C., Chen, J., Sayrafi, M.: PrimaryDecomposition: A *Macaulay2* package. Version 2.0. A *Macaulay2* package available at https://faculty.math.illinois.edu/Macaulay2/doc/Macaulay2-1.21/share/doc/Macaulay2/PrimaryDecomposition/html/index.html