Designs, Codes and Cryptography (2023) 91:3995–4017
https://doi.org/10.1007/s10623-023-01286-6

Skew differential Goppa codes and their application to Mceliece cryptosystem

José Gómez-Torrecillas[1,3] · F. J. Lobillo[1,3,4] · Gabriel Navarro[2,3,4]

Received: 26 September 2022 / Revised: 24 May 2023 / Accepted: 27 July 2023 /
Published online: 19 August 2023

Abstract
A class of linear codes that extends classical Goppa codes to a non-commutative context is defined. An efficient decoding algorithm, based on the solution of a non-commutative key equation, is designed. We show how the parameters of these codes, when the alphabet is a finite field, may be adjusted to propose a McEliece-type cryptosystem.

Keywords Skew-differential Goppa code · Decoding algorithm · McEliece-type cryptosystem

Mathematics Subject Classification 94B35 · 94A60 · 16S36

1 Introduction

Code-based cryptography proposals are still alive after the Round 4 of the NIST Post-Quantum Cryptography competition. The strength of these technologies rests upon the hardness of the decoding problem for a general linear code. Of course, an efficient decoding algorithm is required in practice. So, what is already needed is a family of codes with some conveniently masked properties that facilitate their efficient decoding. The original McEliece

Communicated by G. Korchmaros.

✉ F. J. Lobillo
jlobillo@ugr.es

José Gómez-Torrecillas
gomezj@ugr.es

Gabriel Navarro
gnavarro@ugr.es

1 Department of Algebra, University of Granada, Av. Fuentenueva s/n, 18071 Granada, Spain

2 Department of Computer Science and Artificial Intelligence, University of Granada, Periodista Daniel Saucedo Aranda s/n, 18071 Granada, Spain

3 IMAG, University of Granada, Ventanilla, 11, 18001 Granada, Spain

4 CITIC, University of Granada, Periodista Rafael Gómez Montero, 2, 18071 Granada, Spain

cryptosystem took advantage of such characteristics that the classical binary Goppa codes enjoy.

One way to introduce Goppa codes is the following. Let $F \subseteq L$ be an extension of finite fields and let $g \in L[x]$ be a polynomial which, in this introduction, we assume irreducible for sake of simplicity. A subset of the group of units of the field $L[x]/\langle g \rangle$, whose elements are represented by linear polynomials, is selected. Their inverses serve to build a parity check matrix of the Goppa code. The arithmetic in $L[x]$ is a main tool in the design of efficient decoding algorithms for Goppa codes.

From an algebraic point of view, our proposal replaces, in the simplest case, the cyclic group of units of $L[x]/\langle g \rangle$ by a linear group, whose mathematical structure is more complex. In order to design an efficient decoding algorithm, this non-commutative group is presented as the group of units of a factor ring of the ring of Ore polynomials $L[x; \sigma, \partial]$ modulo a suitable invariant polynomial g. The arithmetic of this non-commutative polynomial ring is used to design an efficient decoding algorithm. Classical Goppa codes become instances of our construction. Therefore, the security of our cryptosystem is expected to be at least as strong as the original one.

In Sect. 2 we recall some basic essentials on Ore polynomials and define skew differential Goppa codes. A non-commutative key equation is derived for these codes (Theorem 1), which turns out to be a left multiple of an equation computed with the help of the Left Extended Euclidean Algorithm in $L[x; \sigma, \partial]$.

The topic of Sect. 3 is the design of an efficient decoding algorithm for skew differential Goppa codes. To this end, the position points are assumed to be P-independent in the sense of [14]. Under this hypothesis, the non-commutative locator polynomial already finds the error positions, and a decoding algorithm, based on the solution of the key equation, is provided (Algorithm 2). This algorithm gives a solution in most cases, but it is possible that its output falls in a decoding failure. An efficient backup algorithm solves any of these failures (Algorithm 3). In resume, the combination of both algorithms correctly computes an error added to a codeword up to the correction capability.

Section 4 describes how to construct parity check matrices and position points to define skew Goppa codes suitable to be used in a code-based cryptosystem. The construction is made for $\partial = 0$, which guarantees the polynomial run-time of the algorithms, see Remark 4.

The cryptosystem is presented in Sect. 5. A discussion on the choice of the parameters of the code is included.

There is a patent pending by University of Granada in order to protect some of the results in this work, see [8].

2 Skew differential Goppa codes and their non-commutative key equation

In this section, the required algebraic framework is introduced. Let σ be an automorphism of finite order μ of a field L. An additive map $\partial : L \to L$ is called a σ-derivation if it satisfies $\partial(ab) = \sigma(a)\partial(b) + \partial(a)b$ for all $a, b \in L$. By $R = L[x; \sigma, \partial]$ we denote the ring of Ore polynomials built from (σ, ∂). This is a fundamental example of non-commutative ring, introduced in [20], whose basic properties may be found in several texts. We follow [3, Chap. 1, Sects. 3 and 4] and adopt its notation, see also [11]. In particular, R is a left and right Euclidean domain. The left division algorithm computes, given $f, d \in R$ with $d \neq 0$, two Ore polynomials $q, r \in R$ such that $f = qd + r$ with $\deg r < \deg d$, where deg denotes the degree (in x) function. Then we will write

$$\text{l-quo-rem}(f, d) = (q, r), \qquad \text{l-quo}(f, d) = q.$$

Given $f, g \in R$, the notation $f \mid_r g$ declares that f is a right divisor of g, which means $Rg \subseteq Rf$, that is, $g = uf$ for some $u \in R$. The notation $f \mid_\ell g$ is used analogously, meaning $gR \subseteq fR$. When $(x - \alpha) \mid_r f$, for $f \in R$ and $\alpha \in L$, we say that α is a *right root* of the Ore polynomial f. Greatest common left/right divisors and least common left/right multiples are well defined since all left/right ideals are principal. Concretely,

$$Rg + Rf = R\,(g, f)_r\,, \quad gR + fR = (g, f)_\ell\,R$$

and

$$Rg \cap Rf = R\,[g, f]_\ell\,, \quad gR \cap fR = [g, f]_r\,R.$$

Remark 1 Let $f, g, f', g' \in R$ nonzero such that $f'f = g'g$. Then $[f, g]_\ell = f'f$ if and only if $\left(f', g'\right)_\ell = 1$. In fact, assume $[f, g]_\ell = f'f = g'g$ and let $d = \left(f', g'\right)_\ell$. Then $f' = df''$ and $g' = dg''$. Hence $df''f = dg''g$, so $[f, g]_\ell \mid_r f''f = g''g$. Therefore $f'f \mid_r f''f$, i.e., $f' \mid_r f''$. It follows that d is a unit, so $d \in L$ and $\left(f', g'\right)_\ell = 1$. Conversely, assume $\left(f', g'\right)_\ell = 1$. If $[f, g]_\ell = f'''f = g'''g$, since $[f, g]_\ell \mid_r f'f = g'g$, there exists $c \in R$ such that $c\,[f, g]_\ell = f'f = g'g$. It follows $cf''' = f'$ and $cg''' = g'$, hence $c \mid_l \left(f', g'\right)_\ell$. Therefore $c \in L$ and $[f, g]_\ell = f'f = g'g$.

The analogous result for least common right multiples and greatest common right divisors also holds.

There exist Left and Right Extended Euclidean algorithms (LEEA and REEA, for short) that compute greatest common divisors and least common multiples on both sides. For our forthcoming reasoning, we will need a very precise statement of the LEEA that provides the Bezout coefficients in each step of the algorithm. This is given in Algorithm 1.

Algorithm 1 `Left Extended Euclidean Algorithm`

```
Require: $f, g \in L[x; \sigma, \partial]$ with $f \neq 0, g \neq 0$.
Ensure: $\{u_i, v_i, r_i\}_{i=0,\dots,h,h+1}$ such that $r_i = u_i f + v_i g$ for every $i$, $r_h = (f, g)_r$, and $u_{h+1} f = [f, g]_\ell$.
  $r_0 \leftarrow f, r_1 \leftarrow g$.
  $u_0 \leftarrow 1, u_1 \leftarrow 0$.
  $v_0 \leftarrow 0, v_1 \leftarrow 1$.
  $i \leftarrow 1$.
  while $r_i \neq 0$ do
      find $q_i, r$ such that $r_{i-1} = q_i r_i + r$ and $\deg r < \deg r_i$        ▷ Left division
      $r_{i+1} \leftarrow r$
      $u_{i+1} \leftarrow u_{i-1} - q_i u_i$
      $v_{i+1} \leftarrow v_{i-1} - q_i v_i$
      $i \leftarrow i + 1$
  end while
  return $\{u_i, v_i, r_i\}_{i=0,\dots,h,h+1}$
```

The output of Algorithm 1 enjoys some properties that will be used later. We record them in the following lemma, whose commutative version may be found in [23, Lemma 3.8].

Lemma 1 *Let $f, g \in R$ and $\{u_i, v_i, r_i\}_{i=0,\dots,h}$ be the coefficients obtained when applying the LEEA to f and g. Then, for all $i = 0, \dots, h$, we have:*

1. $u_i f + v_i g = r_i$.
2. $(u_i, v_i)_\ell = 1$.
3. $\deg f = \deg r_{i-1} + \deg v_i$.

Proof The proof given in [6, Lemma 24] works here step by step. □

Let $0 \neq g \in R$ be invariant, i.e. $Rg = gR$. Therefore, R/Rg is a ring. It is easy to check that $fh = g$ if and only if $h'f = g$, where $gh = h'g$, so $f \mid_r g$ if and only if $f \mid_l g$. In particular, since $x - \gamma$ is irreducible for all $\gamma \in L$, $(x - \gamma, g)_r = 1$ if and only if $(x - \gamma, g)_\ell = 1$.

Observe that $(x - \gamma, g)_\ell = 1$ means that $x - \gamma + Rg$ is a unit in R/Rg, so there exists a unique $h \in R$ with $\deg(h) < \deg(g)$ such that $(x - \gamma)h - 1 \in Rg$ and $h(x - \gamma) - 1 \in Rg$.

Definition 1 Let $F \subseteq L$ be a field extension. Let $g \in R = L[x; \sigma, \partial]$ be a nonzero invariant polynomial. Let $\alpha_0, \dots, \alpha_{n-1} \in L$ be different elements such that $(x - \alpha_i, g)_r = 1$ for all $0 \leq i \leq n-1$, let $h_i \in R$ such that $\deg(h_i) < \deg(g)$ and

$$(x - \alpha_i)h_i - 1 \in Rg, \tag{1}$$

and let $\eta_0, \dots, \eta_{n-1} \in L^*$. A (generalized) skew differential Goppa code $\mathcal{C} \subseteq F^n$ is the set of vectors $(c_0, \dots, c_{n-1}) \in F^n$ such that

$$\sum_{i=0}^{n-1} h_i \eta_i c_i = 0. \tag{2}$$

By a degree argument, (2) is equivalent to

$$\sum_{i=0}^{n-1} h_i \eta_i c_i \in Rg. \tag{3}$$

We say that $\{\alpha_0, \dots, \alpha_{n-1}\}$ are the position points, g is the (skew differential) Goppa polynomial and $h_0, \dots, h_{n-1}$ are the parity check polynomials. If $\partial = 0$, we just call it a (generalized) skew Goppa code.

Remark 2 A classical Goppa code is an instance of the skew differential Goppa codes when σ is the identity map, $\partial = 0$ and $\eta_i = 1$ for all $0 \leq i \leq n-1$.

Remark 3 In [25], linearized Goppa codes are introduced. Since the ring of linearized polynomials over a finite field is isomorphic to the ring of Ore polynomials built from the Frobenius automorphism with trivial skew derivation, linearized Goppa codes become instances of skew differential Goppa codes. Nevertheless, there are some mistakes in this reference. For instance, [25, Proposition 1] seems not to be correct. Indeed, following the notation in [25], let $q = 2$, $m = 3$. The field $\mathbb{F}_{q^m}$ is represented as $\mathbb{F}_{2^3} = \mathbb{F}_2[b]/\langle b^3 + b + 1\rangle$. Let $\mathbf{g} = \big\langle g_1 = b^2 + b, g_2 = b\big\rangle$. We get

$$\begin{aligned}\sigma_{\mathbf{g}}(x) &= \sigma_{\langle g_1, g_2\rangle}(x) = x(x + b^2 + b)(x + b)(x + b^2) = x^4 + x^2 + x,\\ \sigma_{\mathbf{g}_1}(x) &= \sigma_{\langle g_2\rangle}(x) = x(x + b) = x^2 + bx\end{aligned}$$

and

$$\sigma_{\mathbf{g}_1}(x) \circ (x^q - g_1^{q-1}x) = (x^2 + bx) \circ (x^2 + (b^2 + b)x) = x^4 + (b^2 + b + 1)x,$$

so it is not true that $\sigma_{\mathbf{g}}(x) = \sigma_{\mathbf{g}_i}(x) \circ (x^q - g_i^{q-1}x)$.

For the rest of this section a skew differential Goppa code $\mathcal{C}$ is fixed. Let $\{\varepsilon_i \mid 0 \leq i \leq n-1\}$ be the canonical basis of F^n. Assume $c \in \mathcal{C}$ is transmitted and $r \in F^n$ is received. Therefore

$$r = c + e$$

for some $e = \sum_{j=1}^{\nu} e_j \varepsilon_{k_j}$ with $e_j \neq 0$ for $1 \leq j \leq \nu$. The *syndrome polynomial* is defined and computed as

$$s = \sum_{i=0}^{n-1} h_i \eta_i r_i.$$

By (2), it follows that

$$s - \sum_{j=1}^{\nu} h_{k_j} \eta_{k_j} e_j = \sum_{i=0}^{n-1} h_i \eta_i c_i \in Rg = gR. \tag{4}$$

We define the (non-commutative) *error locator polynomial* as

$$\lambda = \left[\{x - \alpha_{k_j} \mid 1 \leq j \leq \nu\}\right]_{\ell} \in R.$$

Then $\deg(\lambda) \leq \nu$ and, for each $1 \leq j \leq \nu$, there exists $\rho_{k_j} \in R$ such that $\deg(\rho_{k_j}) < \nu$ and

$$\lambda = \rho_{k_j}(x - \alpha_{k_j}). \tag{5}$$

The *error evaluator polynomial* is defined as

$$\omega = \sum_{j=1}^{\nu} \rho_{k_j} \eta_{k_j} e_j.$$

It follows that $\deg(\omega) < \nu$.

Our next aim is to derive and solve a non-commutative key equation that relates syndrome, and error locator and error evaluator polynomials. The solution requires the following lemma.

Lemma 2 *Let $f, g \in R$ such that $\deg f < \deg g = \chi$. Assume that there exist $\kappa, \lambda, \omega \in R$ such that $\kappa g + \lambda f = \omega$, $\deg \lambda \leq \left\lfloor \frac{\chi}{2} \right\rfloor$ and $\deg \omega < \left\lfloor \frac{\chi}{2} \right\rfloor$. Let u_I, v_I and r_I be the (partial) Bezout coefficients returned by the LEEA with input g and f, where I is the index determined by the conditions $\deg r_{I-1} \geq \left\lfloor \frac{\chi}{2} \right\rfloor$ and $\deg r_I < \left\lfloor \frac{\chi}{2} \right\rfloor$. Then there exists $h \in R$ such that $\kappa = hu_I$, $\lambda = hv_I$ and $\omega = hr_I$.*

Proof Since $\kappa g + \lambda f = \omega$, $\deg \lambda \leq \left\lfloor \frac{\chi}{2} \right\rfloor$ and $\deg \omega < \left\lfloor \frac{\chi}{2} \right\rfloor$, it follows that $\deg \kappa < \left\lfloor \frac{\chi}{2} \right\rfloor$. By Lemma 1, $\deg v_I + \deg r_{I-1} = \chi$, so that $\deg v_I \leq \chi - \left\lfloor \frac{\chi}{2} \right\rfloor$.

Write $[\lambda, v_I]_{\ell} = a\lambda = bv_I$, where $a, b \in R$ with $\deg a \leq \deg v_I \leq \chi - \left\lfloor \frac{\chi}{2} \right\rfloor$ and $\deg b \leq \deg \lambda \leq \left\lfloor \frac{\chi}{2} \right\rfloor$. Then $(a, b)_{\ell} = 1$ by Remark 1.

From $\kappa g + \lambda f = \omega$ we get

$$a\kappa g + a\lambda s = a\omega. \tag{6}$$

By Lemma 1, we have $u_I g + v_I f = r_I$, which we multiply on the left by b to get

$$bu_I g + bv_I s = br_I. \tag{7}$$

Hence, from (6) and (7),

$$(a\kappa - bu_I)g = a\omega - br_I. \tag{8}$$

Since

$$\deg(a\omega - br_I) \leq \max\{\deg a + \deg \omega, \deg b + \deg r_I\}$$
$$< \max\left\{\chi - \left\lfloor \frac{\chi}{2} \right\rfloor + \left\lfloor \frac{\chi}{2} \right\rfloor, \left\lfloor \frac{\chi}{2} \right\rfloor + \left\lfloor \frac{\chi}{2} \right\rfloor\right\} = \chi = \deg g,$$

it follows, from (8), that $a\kappa = bu_I$ and $a\omega = br_I$. Actually, $(a, b)_\ell = 1$ yields $[\kappa, u_I]_\ell = a\kappa = bu_I$ and $[\omega, r_I]_\ell = a\omega = br_I$ by Remark 1. In particular, $\deg a \leq \deg r_I < \left\lfloor \frac{\chi}{2} \right\rfloor$.

Let $[a, b]_r = aa' = bb'$. Since $[\lambda, v_I]_\ell$ is a right multiple of a and b, there exists $m \in R$ such that $[\lambda, v_I]_\ell = [a, b]_r m$. Then $a\lambda = bv_I = aa'm = bb'm$. Thus, $\lambda = a'm$ and $v_I = b'm$ and, by minimality, $(\lambda, v_I)_r = m$. Similar arguments prove that there exist $m', m'' \in R$ such that $u_I = b'm'$ and $\kappa = a'm'$, and that $r_I = b'm''$ and $\omega = a'm''$. Nevertheless, by Lemma 1, $(u_I, v_I)_\ell = 1$, so $b' = 1$. In this way, $b = aa'$ and we get $\lambda = a'v_I$, $\omega = a'r_I$ and $\kappa = a'u_I$. This completes the proof. □

Theorem 1 *The error locator λ and the error evaluator ω polynomials satisfy the non-commutative key equation*

$$\omega = \kappa g + \lambda s, \tag{9}$$

for some $\kappa \in R$. Assume that $\nu \leq t = \left\lfloor \frac{\deg g}{2} \right\rfloor$. Let u_I, v_I and r_I be the Bezout coefficients returned by the left extended Euclidean algorithm with input g and s, where I is the index determined by the conditions $\deg r_{I-1} \geq t$ and $\deg r_I < t$. Then there exists $h \in R$ such that $\kappa = hu_I$, $\lambda = hv_I$ and $\omega = hr_I$.

Proof Since $(x - \alpha_i)h_i + Rg = 1 + Rg$ for all $0 \leq i \leq n-1$, we get from (4) the following computation in the ring R/Rg:

$$\begin{aligned}\lambda s + Rg &= \sum_{j=1}^{\nu} \lambda h_{k_j} \eta_{k_j} e_j + Rg \\ &= \sum_{j=1}^{\nu} \rho_{k_j}(x - \alpha_{k_j}) h_{k_j} \eta_{k_j} e_j + Rg \\ &= \sum_{j=1}^{\nu} \rho_{k_j} \eta_{k_j} e_j + Rg \\ &= \omega + Rg.\end{aligned}$$

This proves (9). By construction,

$$\deg s \leq \max\{\deg(h_i) \mid 0 \leq i \leq n-1\} < \deg g,$$

so the second statement of the theorem follows from Lemma 2. □

3 Decoding algorithms

From now on we assume

$$\nu \leq t = \left\lfloor \frac{\deg g}{2} \right\rfloor.$$

It follows from Theorem 1 that the condition $(\lambda, \omega)_\ell = 1$ implies that λ and ω are left associated to v_I and r_I, respectively. Hence, under this condition the LEEA computes the error locator and evaluator polynomials. In the commutative case, it is easy to check that locator and evaluator are always relatively prime. Although, in our experiments, most of examples in this non-commutative setting already satisfy the condition $(\lambda, \omega)_\ell = 1$, this is not always the case (see Example 3). For a correct decoding, we need to know when v_I, r_I are actually the error locator and the error evaluator polynomials, and if the error locator polynomial already locates the error positions.

In order to proceed, we need the notion of left P-independent set in the sense of [4, 15]. From now on, we assume the following hypothesis on the position points.

Hypothesis 1 *We assume that* $\{\alpha_0, \ldots, \alpha_{n-1}\} \subseteq L$ *is left P-independent, that is,*

$$\deg\left[\{x - \alpha_i \mid 0 \le i \le n-1\}\right]_\ell = n. \tag{10}$$

Observe that, by [4, Theorem 5.3], every subset of a P-independent set is P-independent.

As a consequence of Hypothesis 1, $\deg(\lambda) = \nu$. Let us deduce that λ already locates the error positions.

Proposition 1 $x - \alpha_k \mid_r \lambda$ *if and only if* $k \in \{k_1, \ldots, k_\nu\}$.

Proof If $x - \alpha_k \mid_r \lambda$ with $k \notin \{k_1, \ldots, k_\nu\}$ then the set $\{\alpha_k, \alpha_{k_1}, \ldots, \alpha_{k_\nu}\}$ is left P-dependent. □

3.1 Decoding algorithm with unlikely decoding failure

In this subsection we give a criterion on the partial outputs of LEEA to decide if λ is left associated to v_I (Proposition 2). This leads to a decoding algorithm (Algorithm 2) that turns out to work in most cases. In the next subsection, we discuss how to correctly decode when Algorithm 2 outputs a decoding failure. Our approach is adapted from [6, Lemma 26 and Theorem 15].

Lemma 3 *Let* $\{i_1, \ldots, i_m\} \subseteq \{0, \ldots, n-1\}$ *with* $1 < m \le n$, *and*

$$f = \left[x - \alpha_{i_1}, \ldots, x - \alpha_{i_m}\right]_\ell .$$

Let $f_1, \ldots, f_m \in R$ *such that* $f = f_j(x - \alpha_{i_j})$ *for all* $1 \le j \le m$. *Then:*

1. $[f_1, \ldots, f_m]_r = f$ *and* $(f_1, \ldots, f_m)_\ell = 1$.
2. $R/fR = \bigoplus_{j=1}^m f_j R/fR$.
3. *For any* $h \in R$ *with* $\deg h < m$ *there exist* $a_1, \ldots, a_m \in L$ *such that* $h = \sum_{j=1}^m f_j a_j$.
4. *The set* $\{f_1, \ldots, f_m\}$ *gives, modulo* fR, *a basis of* R/fR *as an* L*-vector space.*

Proof (1) By Hypothesis 1, $\{\alpha_{i_1}, \ldots, \alpha_{i_m}\}$ is left P-independent. So, by (10), $\deg f = m$ and, thus, $\deg f_j = m - 1$ for every $j = 1, \ldots, m$. Since $m > 1$, the degree of $[f_1, \ldots, f_m]_r$ must be at least $m - 1 + 1 = m$. But f is obviously a common left multiple of $f_1, \ldots, f_m$, whence $f = [f_1, \ldots, f_m]_\ell$. It is straightforward to check that $(f_1, \ldots, f_m)_\ell = 1$, otherwise there would be a left common multiple of $x - \alpha_{i_j}$ for $1 \le j \le m$ with degree smaller than $\deg f$.

(2) Since $fR \subseteq f_j R$ for all $1 \le j \le m$ and $(f_1, \ldots, f_m)_\ell = 1$, we get $R/fR = \sum_{j=1}^m f_j R/fR$. Observe that $f_j R/fR \cong R/(x - \alpha_{i_j})R$ is one-dimensional over L. Since the dimension of R/fR as an L–vector space is $\deg f = m$, we get that the sum is direct.

(3) and (4) follow from (2). □

Proposition 2 *Let $u, v, r \in R$ such that $ug + vs = r$, $hu = \kappa$, $hv = \lambda$ and $hr = \omega$ for some $h \in R$. Let $T = \{l_1, l_2, \dots, l_m\} = \{0 \leq l \leq n-1 \mid (x - \alpha_l) \mid_r v\}$. Then $m = \deg v$ if and only if $\deg h = 0$.*

Proof Since $v \mid_r \lambda$, every right root of v is a right root of λ, hence $\{l_1, \dots, l_m\} \subseteq \{k_1, \dots, k_\nu\}$ by Proposition 1. We reorder the set of error positions in such a way that $T = \{k_1, \dots, k_m\}$ with $m \leq \nu$. If $\deg h = 0$, then $m = \nu$ and $\deg v = \nu$ by (10), since $\{\alpha_{k_1}, \dots, \alpha_{k_\nu}\}$ is left P-independent. Conversely, if $m = \deg v$, then

$$v = \left[\{x - \alpha_{k_j} \mid 1 \leq j \leq m\}\right]_\ell$$

by (10) and Hypothesis 1. Recall that $\lambda = \rho_{k_j}(x - \alpha_{k_j})$ and write $v = \rho'_j(x - \alpha_{k_j})$ for all $1 \leq j \leq m$. Since

$$\deg r = \deg \omega - \deg h = \deg \omega + \deg v - \deg \lambda \leq \nu - 1 + m - \nu = m - 1,$$

we get from Lemma 3 that $r = \sum_{i=1}^m \rho'_i a_i$ for some $a_1, \dots, a_m \in L$. On the other hand, $\lambda = hv$. Thus, for any $1 \leq j \leq m$, $\rho_{k_j}(x - \alpha_{k_j}) = h\rho'_j(x - \alpha_{k_j})$, so $\rho_{k_j} = h\rho'_j$. Now, $hr = \omega$, so

$$\sum_{j=1}^m \rho_{k_j} a_j = h\left(\sum_{j=1}^m \rho'_j a_j\right) = hr = \omega = \sum_{j=1}^m \rho_{k_j} e_j + \sum_{j=m+1}^{\nu} \rho_{k_j} e_j. \tag{11}$$

By Lemma 3, $\{\rho_{k_1}, \dots, \rho_{k_\nu}\}$ is a basis of $R/\lambda R$ as a right L–vector space. Therefore, since $e_j \neq 0$ for every $1 \leq j \leq \nu$, Eq. (11) implies that $m = \nu$ and, thus, $\deg h = 0$. □

Theorem 1 and Proposition 2 ensure the correctness of the decoding algorithm described in Algorithm 2.

3.2 Solving decoding failures

Proposition 2 gives a sufficient condition which tells us if we have actually found the solution of (9), and, therefore, the output of Algorithm 2 is the error polynomial. Nevertheless, a decoding failure may occur, see Example 3, and we might not had compute the error locator polynomial, but only a proper right divisor. So we need to find new right roots of λ.

Proposition 3 *Let $u, v, r \in R$ such that $ug + vs = r$, $hu = \kappa$, $hv = \lambda$ and $hr = \omega$ for some $h \in R$. Let $k \in \{0, \dots, n-1\}$ such that $x - \alpha_k \nmid_r v$ but $x - \alpha_k \mid_r \lambda$. Set $v' = [x - \alpha_k, v]_\ell$ and let $h'' \in R$ such that $h''v = v'$. Define $u' = h''u$ and $r' = h''r$. Then $u'g + v's = r'$, $h'u' = \kappa$, $h'v' = \lambda$ and $h'r' = \omega$ for some $h' \in R$.*

Proof Since $\lambda = hv$, it follows that $[x - \alpha_k, v]_\ell \mid_r \lambda$, so there exists $h' \in R$ such that $\lambda = h'[x - \alpha_k, v]_\ell$. Then $hv = \lambda = h'h''v$, hence $h = h'h''$. Multiplying $ug + vs = r$ by h'' on the left, we get $u'g + v's = r'$. Moreover, $\kappa = hu = h'h''u = h'u'$, $\lambda = hv = h'h''v = h'v'$ and $\omega = hr = h'h''r = h'r'$. □

Proposition 4 *Assume $\lambda = hv$ with $\deg h \geq 1$. Let*

$$\{s_1, \dots, s_m\} = \left\{i \in \{0, \dots, n-1\} \mid (x - \alpha_i) \mid_r v\right\}$$

and $\{l_1, \dots, l_r\} = \{0, \dots, n-1\} \setminus \{s_1, \dots, s_m\}$. For any $1 \leq i \leq r$, let $f_i = \left[f_{i-1}, x - \alpha_{l_i}\right]_\ell$ with $f_0 = v$. Then:

Algorithm 2 Decoding algorithm for skew differential Goppa codes with unlikely decoding failure

```
Require: A skew differential Goppa code $\mathcal{C}$ of length $n$, correction capability $t$, position points
    $\{\alpha_0, \ldots, \alpha_{n-1}\} \subseteq L$, $\eta_0, \ldots, \eta_{n-1} \in L^*$, skew differential Goppa invariant polynomial $g \in L[x; \sigma, \partial]$
    with $\deg(g) = 2t$, and parity check polynomials $h_0, \ldots, h_{n-1} \in L[x; \sigma, \partial]$ of degree $2t - 1$.
Require: A received word $y = (y_0, \ldots, y_{n-1}) \in F^n$.
Ensure: A vector $e \in F^n$ such that $\mathrm{w}(e) \leq t$ and $y - e \in \mathcal{C}$, or decoding failure.
 1: $s \leftarrow \sum_{i=0}^{n-1} h_i \eta_i y_i$
 2: if $s = 0$ then
 3:     return the zero vector
 4: end if
 5: $r_{prev} \leftarrow g, r_{curr} \leftarrow s, v_{prev} \leftarrow 0, v_{curr} \leftarrow 1$          ▷ LEEA
 6: while $\deg(r_{curr}) \geq t$ do
 7:     $f, r \leftarrow$ l-quo-rem$(r_{prev}, r_{curr})$
 8:     $v \leftarrow v_{prev} - f v_{curr}, v_{prev} \leftarrow v_{curr}, r_{prev} \leftarrow r_{curr}, v_{curr} \leftarrow v, r_{curr} \leftarrow r$
 9: end while
10: $pos \leftarrow \{\}, other = \{0, \ldots, n-1\}$          ▷ Finding error positions
11: for $0 \leq i \leq n-1$ do
12:     if $\alpha_i$ is a right root of $v_{curr}$ then
13:         $pos \leftarrow pos \cup \{i\}, other = other \setminus \{i\}$
14:     end if
15: end for
16: if $\deg(v_{curr}) >| pos |$ then
17:     'Decoding failure'
18:     stop
19: end if
20: for $j \in pos$ do          ▷ Finding error values
21:     $\rho_j \leftarrow$ l-quo$(v_{curr}, x - \alpha_j)$
22: end for
23: Solve the linear system $r_{curr} = \sum_{j \in pos} \rho_j \eta_j e_j$
24: $e(x) \leftarrow \sum_{j \in pos} e_j x^j$
25: return the vector associated to the polynomial $e(x)$
```

1. *There exists* $d \geq 0$ *such that* $\deg(f_{d-1}) = \deg(f_d)$*,*
2. *If* d_0 *is the minimal index such that* $\deg(f_{d_0-1}) = \deg(f_{d_0})$*, then* $d_0 \in \{k_1, \ldots, k_\nu\}$*.*

Proof For any $1 \leq i \leq r$, let $\lambda_i = \left[\lambda_{i-1}, x - \alpha_{l_i}\right]_\ell$ with $\lambda_0 = \lambda$. It is clear that $f_i \mid_r \lambda_i$ for any $1 \leq i \leq r$. Suppose that the sequence $\{\deg(f_i)\}_{0 \leq i \leq r}$ is strictly increasing. Hence $\deg(f_r) = r + \deg(v) = n - m + \deg(v) > n$ because, by Proposition 2, $\deg(v) > m$. This is not possible, since $f_r \mid_r \lambda_r = [\{x - \alpha_i \mid 0 \leq i \leq n-1\}]_r$ whose degree is bounded from above by n. So there exists a minimal $d_0 \geq 0$ such that $\deg(f_{d_0-1}) = \deg(f_{d_0})$. Now, $x - \alpha_{i_{d_0}} \mid_r f_{d_0-1} \mid_r \lambda_{d_0-1} = \left[\lambda, x - \alpha_{l_1}, \ldots, x - \alpha_{l_{d_0-1}}\right]_\ell$. Since, $l_{d_0} \neq l_1, \ldots, l_{d_0-1}$, $x - \alpha_{l_{d_0}} \mid_r \lambda$. Thus, $d_0 \in \{k_1, \ldots, k_\nu\}$. □

Propositions 3 and 4 provide a way to find the locator if a decoding failure happens. This is presented in Algorithm 3.

Remark 4 Concerning the complexity, the run-time of Algorithm 2 is dominated by the execution of the LEEA and a linear system resolution. In Algorithm 3, the internal loop, that finds an error position, computes the least common left multiple of a linear polynomial and v_{curr}, updating v_{curr} to this least common left multiple, until the process does not increase the degree. Its theoretical complexity is then dominated by an n-times execution of a least common left multiple of bounded polynomials. Now, this loop is executed, at most, the number of error positions, so that the complexity of Algorithm 3 is bounded polynomially

Algorithm 3 Solving decoding failures

```
Require: A skew differential Goppa code $\mathcal{C}$ of length $n$, correction capability $t$, position points
    $\{\alpha_0, \dots, \alpha_{n-1}\} \subseteq L$, $\eta_0, \dots, \eta_{n-1} \in L^*$, skew differential Goppa invariant polynomial $g \in R$ with
    $\deg(g) = 2t$, and parity check polynomials $h_0, \dots, h_{n-1} \in R$ of degree $2t - 1$.
Require: The polynomials $v_{curr}, r_{curr}$ and the sets $pos, other$ in Algorithm 2
Ensure: The locator polynomial $\lambda$.
    while $\deg(v_{curr}) > | pos |$ do
        $f \leftarrow v_{curr}, e \leftarrow \deg(f)$
        $i \leftarrow$ one element in $other$, $other = other \setminus \{i\}$
        $f \leftarrow [f, x - \alpha_i]_\ell$
5:      while $\deg(f) > e$ do
            $e \leftarrow e + 1$
            $i \leftarrow$ one element in $other$, $other = other \setminus \{i\}$
            $f \leftarrow [f, x - \alpha_i]_\ell$
        end while
10:     $pos = pos \cup \{i\}$, $other = \{0, \dots, n-1\} \setminus pos$
        $v \leftarrow v_{curr}$, $v_{curr} \leftarrow [v, x - \alpha_i]_\ell$, $h \leftarrow$ l-quo$(v_{curr}, v)$, $r_{curr} \leftarrow h r_{curr}$
        for $i \in other$ do
            if $\alpha_i$ is a right root of $v_{curr}$ then
                $pos \leftarrow pos \cup \{i\}$, $other = other \setminus \{i\}$
15:         end if
        end for
    end while
    return $v_{curr}, r_{curr}, pos$
```

with respect to the complexity of the LEEA. Consequently, in general, the conjunction of Algorithms 2 and 3 has polynomial run-time, in the worst case, with respect to the execution of the LEEA and a linear system resolution.

In the setting of the cryptosystem to be described in Sect. 5, that is, skew polynomials over a finite field with $\partial = 0$, according to [7, Lemma 3.3], the execution of the LEEA is in $\mathcal{O}(n^2)$ operations in the field, whilst the traditional approach to solve the linear system in Line 23 of Algorithm 2 is by Gaussian elimination, which can be done in $\mathcal{O}(t^3)$. Therefore the complexity of Algorithm 2 belongs to $\mathcal{O}(t^3 + n^2)$ operations in the field, whilst Algorithm 3 belongs to $\mathcal{O}(tn^3)$.

It was noticed by one of the referees that there is a fast computation of the left extended Euclidean algorithm in [2]. These results could be used to speed our algorithms up in the finite field case.

Observe that, by Theorem 1 and Proposition 2, decoding failure cannot happen if $(\lambda, \omega)_\ell = 1$. Next, we analyze this condition.

Proposition 5 *Under the notation of Sect. 2, the following statements are equivalent:*

1. $(\omega, \lambda)_\ell = 1$.
2. $\omega + \lambda R$ *generates* $R/\lambda R$ *as a right* R*–module.*
3. *The set* $\{(\omega + \lambda R)x^i \mid 0 \leq i \leq \nu - 1\}$ *is right linearly independent over* L.

Proof The equivalence between (1) and (2) is a direct consequence of Bezout's Theorem. It is clear that $\omega + \lambda R$ generates the right R–module $R/\lambda R$ if and only if $\{(\omega + \lambda R)x^i \mid 0 \leq i \leq \nu - 1\}$ spans $R/\lambda R$ as a right L–vector space. Since the dimension over L of $R/\lambda R$ is ν, the equivalence between (2) and (3) becomes clear. □

In the skew case, i.e. $\partial = 0$, a more precise analysis can be done. Besides the partial norms $N_i(a)$, for any $a \in L$ and $i \in \mathbb{N}$, the -ith norm of a is defined as $\mathrm{N}_{-i}(a) = \mathrm{N}^{\sigma}_{-i}(a) = \mathrm{N}^{\sigma^{-1}}_{i}(a)$, i.e.

$$\mathrm{N}_{-i}(a) = a\sigma^{-1}(a)\dots\sigma^{-i+1}(a).$$

Since σ has order μ, it follows that $\mathrm{N}_{\mu}(\gamma) = \mathrm{N}_{-\mu}(\gamma)$. Moreover, for each $f = \sum_j f_j x^j \in R$ and all $\gamma \in L$, there exists $h \in R$ such that

$$f = (x-\gamma)h + \sum_j \sigma^{-j}(f_j)\,\mathrm{N}_{-j}(\gamma). \tag{12}$$

Lemma 4 *The j-coordinate of $\omega x^i + \lambda R$ with respect to the basis $\{\rho_{k_1}, \dots, \rho_{k_\nu}\}$ is $\mathrm{N}_{-i}(\alpha_{k_j})\sigma^{-i}(\eta_{k_j})\sigma^{-i}(e_j)$, for any $1 \le j \le \nu$.*

Proof By (12), α_{k_j} is a left root of $x^i - \mathrm{N}_{-i}(\alpha_{k_j})$. Then $x^i - \mathrm{N}_{-i}(\alpha_{k_j}) \in (x-\alpha_{k_j})R$. Multiplying on the left by ρ_{k_j}, $\rho_{k_j}x^i - \rho_{k_j}\,\mathrm{N}_{-i}(\alpha_{k_j}) \in \lambda R$. Thus, in $R/\lambda R$,

$$\begin{aligned}\omega x^i &= \sum_{j=1}^{\nu} \rho_{k_j}\eta_{k_j}e_j x^i \\ &= \sum_{j=1}^{\nu} \rho_{k_j}x^i\sigma^{-i}(\eta_{k_j})\sigma^{-i}(e_j) \\ &= \sum_{j=1}^{\nu} \rho_{k_j}\,\mathrm{N}_{-i}(\alpha_{k_j})\sigma^{-i}(\eta_{k_j})\sigma^{-i}(e_j)\end{aligned}$$

and the result follows. □

Proposition 6 $(\omega, \lambda)_\ell = 1$ *if and only if*

$$\det\left(\mathrm{N}_{-i}(\alpha_{k_j})\sigma^{-i}(\eta_{k_j})\sigma^{-i}(e_j)\right)_{0\le i\le \nu-1 1\le j\le \nu} \neq 0.$$

Proof It follows from Lemma 4 and Proposition 5. □

Remark 5 Example 3 shows a system providing, under certain carefully chosen errors, a decoding failure. However, as we have pointed out above, this unlikely occurs. In the setting of Example 3, our experiments under randomized errors in the transmission result a probability of 0.003 of obtaining a decoding failure. Whenever the field extension is not trivial, which is the standard setting in practice, none of our experiments outputted a decoding error. For instance, under parameters $F = \mathbb{F}_{2^2}$, $n = 512$ and $t = 5$, after 4 millions executions, no decoding failure was found. This suggests that, for non trivial extensions, there is no decoding failure. Unfortunately, we have been unable to prove it, so we leave this assertion as an open problem.

4 Parity check matrices and position points for skew Goppa codes

This section deals with the computation of parity-check matrices and the choice of the position points for skew Goppa codes. Although most of results are still valid in the skew differential case, the presentation become less technical under the assumption $\partial = 0$. On the other hand, this level of generality suffices for our main purpose, namely, the design of a cryptosystem based on skew Goppa codes over a finite field.

For a given skew differential Goppa code, a parity check matrix can be derived from (2). We make it explicit in the skew Goppa case. So let $R = L[x;\sigma]$, where L is a finite extension of a given field F, and σ is a field automorphism of L of finite order μ. Let $\mathcal{C}$ be a skew Goppa code with Goppa polynomial $g \in R$, position points $\{\alpha_0, \dots, \alpha_{n-1}\}$, $\eta_0, \dots, \eta_{n-1} \in L^*$ and parity check polynomials $h_0, \dots, h_{n-1}$. Let $\deg(g) = \chi$, $h_i = \sum_{j=0}^{\chi-1} h_{i,j}x^j$ and

$$\widehat{H} = \begin{pmatrix} \sigma^{-0}(h_{0,0})\eta_0 & \sigma^{-0}(h_{1,0})\eta_1 & \cdots & \sigma^{-0}(h_{n-1,0})\eta_{n-1} \\ \sigma^{-1}(h_{0,1})\eta_0 & \sigma^{-1}(h_{1,1})\eta_1 & \cdots & \sigma^{-1}(h_{n-1,1})\eta_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma^{-\chi+1}(h_{0,\chi-1})\eta_0 & \sigma^{-\chi+1}(h_{1,\chi-1})\eta_1 & \cdots & \sigma^{-\chi+1}(h_{n-1,\chi-1})\eta_{n-1} \end{pmatrix}.$$

Proposition 7 *For each $\gamma \in L$, let $\mathfrak{v}(\gamma)$ denote its F–coordinates, as a column vector, with respect to a fixed F–basis of L. Let*

$$H = \left(\mathfrak{v}(\sigma^{-j}(h_{i,j})\eta_i) \right)_{\substack{0 \le j \le \chi-1 \\ 0 \le i \le n-1}} \in F^{(\chi m)\times n}.$$

Then H is a parity check matrix for $\mathcal{C}$.

Proof Observe that

$$\sum_{i=0}^{n-1} h_i \eta_i c_i = \sum_{i=0}^{n-1}\sum_{j=0}^{\chi-1} h_{i,j}x^j \eta_i c_i =$$

$$\sum_{i=0}^{n-1}\sum_{j=0}^{\chi-1} x^j \sigma^{-j}(h_{i,j})\eta_i c_i = \sum_{j=0}^{\chi-1} x^j \sum_{i=0}^{n-1} \sigma^{-j}(h_{i,j})\eta_i c_i,$$

so (2) is also equivalent to

$$\sum_{i=0}^{n-1} \sigma^{-j}(h_{i,j})\eta_i c_i = 0, \quad 0 \le j \le \chi - 1,$$

i.e.

$$(c_0, c_1, \dots, c_{n-1})\widehat{H}^{\mathrm{T}} = 0.$$

Since $\mathcal{C} \subseteq F^n$, $(c_0, c_1, \dots, c_{n-1}) \in \mathcal{C}$ if and only if $(c_0, c_1, \dots, c_{n-1})H^{\mathrm{T}} = 0$. □

We gather from [4, 14] the information on P-independent sets needed to describe every possible set of position points in the skew Goppa case.

It is well known that the center of $R = L[x;\sigma]$ is $K[x^\mu]$, where $K = L^\sigma$, the invariant subfield of L under σ. So, for every $a \in L$, the polynomial $x^\mu - \mathrm{N}(a)$ is central, where

$$\mathrm{N}(a) = a\sigma(a)\cdots\sigma^{\mu-1}(a)$$

is the norm of a. Define, following [13], the conjugate of a under $c \in L^*$ as ${}^{c}a = \sigma(c)ac^{-1}$, and the conjugacy class of a as

$$\Delta(a) = \{{}^{c}a : c \in L^*\}.$$

By virtue of Hilbert's 90 Theorem (see e.g. [16, Chap. VI, Theorem 6.1]), $\Delta(a) = \Delta(b)$ if, and only if, $\mathrm{N}(a) = \mathrm{N}(b)$. Hence,

$$\Delta(a) = \{b \in L : \mathrm{N}(a) = \mathrm{N}(b)\}. \tag{13}$$

Observe that these conjugacy classes form a partition of L.

For each $f = \sum_j f_j x^j \in R$ and any $b \in L$, by [13, Lemma 2.4], there exists $h \in R$ such that

$$f = h(x - b) + \sum_j f_j \, \mathrm{N}_j(b), \tag{14}$$

where $\mathrm{N}_j(b) \in L$ is defined as

$$\mathrm{N}_0(b) = 1, \text{ and } \mathrm{N}_j(b) = b\sigma(b)\ldots\sigma^{j-1}(b) \text{ for } j \geq 1.$$

Observe that $\mathrm{N}(b) = \mathrm{N}_\mu(b)$. We get thus from (14) and (13) that

$$\Delta(a) = \{b \in L : (x - b) \mid_r x^\mu - \mathrm{N}(a)\}. \tag{15}$$

We are now in a position to show how to build P-independent sets by using the general theory as established in [4, 14].

Proposition 8 *Given $a \in L^*$, the P-independent subsets of $\Delta(a)$ are those of the form $\{{}^{c_1}a, \ldots, {}^{c_m}a\}$, where $m \leq \mu$ and $\{c_1, \ldots, c_m\}$ is a K–linearly independent subset of L. Moreover, $m = \mu$ if and only if*

$$\left[x - {}^{c_1}a, \ldots, x - {}^{c_m}a\right]_\ell = x^\mu - \mathrm{N}(a).$$

Proof According to [4, Theorem 5.3], $\{{}^{c_1}a, \ldots, {}^{c_m}a\}$ is P-independent if and only if $\{c_1, \ldots, c_m\}$ is linearly independent over the σ–centralizer of a, given by

$$C^\sigma(a) = \{c \in L \setminus \{0\} : {}^c a = a\} \cup \{0\},$$

which is a subfield of L. Indeed, $C^\sigma(a) = K$, so we obtain the first statement. The second one is derived from (15). □

Proposition 9 *A subset $\Gamma \subseteq L^*$ is P-independent of and only if*

$$\Gamma = \Gamma_1 \cup \cdots \cup \Gamma_r,$$

where $\Gamma_i \subseteq \Delta(a_i)$ is P-independent for all $i = 1, \ldots, r$, and $a_1, \ldots, a_r \in L$ are nonzero elements of different norm.

Proof Since the conjugacy classes form a partition of L^* and subsets of P-independent sets are P-independent, we deduce that every P-independent set $\Gamma \subseteq L^*$ decomposes as $\Gamma = \Gamma_1 \cup \cdots \cup \Gamma_r$ for $\Gamma_i \subseteq \Delta(a_i)$ for $a_1, \ldots, a_r \in L^*$ of different norms.

To reason the converse, observe first that the equality (15) says that, for each $a \in L^*$, the conjugacy class $\Delta(a)$ is precisely the set of all right roots, in the sense of [14], of the skew polynomial $x^\mu - \mathrm{N}(a)$. So, these conjugacy classes are instances of full algebraic subsets of L^* to which [14, Corollary 4.4] can be applied. Thus, if $\Gamma_i \subseteq \Delta(a_i)$ is P-independent, then, by virtue of Proposition 8, it corresponds to a K–linearly subset of L, which is a subset of a K–basis B_i of L. Again by Proposition 8, B_i gives a maximal P-independent subset Λ_i of $\Delta(a_i)$ (a P-basis, in the words of [14]), that contains Γ_i. By [14, Corollary 4.4], $\Lambda_1 \cup \cdots \cup \Lambda_r$ is a P-basis of $\Delta_1(a_1) \cup \cdots \cup \Delta_r(a_r)$. As a consequence, $\Gamma_1 \cup \cdots \cup \Gamma_r$ is P-independent. □

As for the selection of the skew Goppa polynomial concerns, we may state:

Proposition 10 *Consider $h \in K[x^\mu]$ without roots in K. Then $g = x^a h \in L[x; \sigma]$ has no right roots in L^* for any $a \geq 0$.*

Proof If $\alpha \in L^*$ is a right root of g, then α is a right root of $h \in L[x;\sigma]$. Then, by Proposition 8, $x^\mu - \mathrm{N}(\alpha)$ is a right divisor in $L[x;\sigma]$ of the central polynomial h. This gives that $x^\mu - \mathrm{N}(\alpha)$ is a divisor of $h \in K[x^\mu]$, that is, $\mathrm{N}(\alpha) \in K$ is a root of $h(x^\mu)$. □

5 A McEliece cryptosystem based on skew Goppa codes

To design a skew Goppa code $\mathcal{C}$, we first choose, as alphabet, a finite field $F = \mathbb{F}_q$, where $q = p^d$ for a prime p. We set the length n and the correction capacity $t < n/2$. In practice, this parameter t is much smaller than n, as we will see below. Algorithms 2 and 3 guarantee that we may set

$$t = \left\lfloor \frac{\deg g}{2} \right\rfloor, \tag{16}$$

where g is the skew Goppa polynomial. We must build the skew polynomial ring $R = L[x;\sigma]$, where L is an extension of F of degree m, so $L = \mathbb{F}_{q^m}$. We choose t, n, m such that $2mt \leq n$ since, from (2), a parity check matrix over F has size $2mt \times n$. If $2mt$ is too close or too far from n, we get codes with very small or very large dimension. For instance, in the Classic McEliece NIST's Post-Quantum Cryptography Standardization Project proposal, see [1], the proposed code rates, the ratios between dimension and length, are ≈ 0.75.

From the relation

$$\dim_F \mathcal{C} = n - \operatorname{rank}(H) \geq n - 2mt, \tag{17}$$

by choosing

$$m \leq \frac{n}{4t},$$

we obtain that

$$\frac{\dim_F \mathcal{C}}{n} \geq 0.5.$$

If the dimension of $\mathcal{C}$ is strictly greater than $n - 2mt$, then we choose randomly a linear subcode $\mathcal{C}\simeq$ of $\mathcal{C}$ with that dimension. Setting

$$\frac{n}{10t} \leq m \leq \frac{n}{4t},$$

then

$$0.5 \leq \frac{\dim_F \mathcal{C}'}{n} \leq 0.8.$$

The field automorphism σ of L is given as a power of the Frobenius automorphism τ, that is $\tau(a) = a^p$, so we pick $1 \leq s \leq dm$ and set $\delta = \gcd(s, dm)$. Define $\sigma = \tau^s$, which has order

$$\mu = \frac{dm}{\delta},$$

and $K = L^\sigma = \mathbb{F}_{p^\delta}$. If $\delta = dm$, then the automorphism is the identity and we recover the classical Goppa codes as observed in Remark 2.

The definition of the skew Goppa code $\mathcal{C}$ requires the specification of a P-independent subset of L^*, the position points, and an invariant polynomial $g \in R$ having no right root among these points. As for the first task concerns, we describe all maximal P-independent subsets of L^*. Every other P-independent set is a subset of one of these.

Proposition 11 *Let γ be a primitive element in L. Every maximal P-independent subset of L^* if of the form*

$$\{\sigma(c_{ij})\gamma^i c_{ij}^{-1} : i = 0, \dots, p^\delta - 2, j = 0, \dots, \mu - 1\},$$

where $\{c_{i0}, \dots, c_{i\mu-1}\}$ is a K–basis of L for each $i = 0, \dots, p^\delta - 2$. As a consequence, if a P-independent subset of L^* has n elements, then $n \leq (p^\delta - 1)\mu$.

Proof It is well-known that $\mathrm{N}(\gamma)$ is a primitive element of K. Thus,

$$\{\mathrm{N}(\gamma^i) : i = 0, \dots, p^\delta - 2\}$$

is a set of representatives of the conjugacy classes of L^* according to (13). The proposition holds now from Propositions 8 and 9. □

Example 1 Let $\{\alpha, \sigma(\alpha), \dots, \sigma^{\mu-1}(\alpha)\}$ be a normal basis of L/K. For $0 \leq i \leq \mu - 1$, set $\beta_i = \sigma^{i+1}(\alpha)/\sigma^i(\alpha)$. Proposition 11 implies that

$$\left\{\gamma^i \beta_j \mid 0 \leq i \leq p^\delta - 2, 0 \leq j \leq \mu - 1\right\}$$

is a maximal P-independent set of L^*.

As for the choice of the skew Goppa polynomial concerns, we may set $g = x^a h$, for any central non constant polynomial $h \in K[x^\mu]$ without roots in K and $a \geq 0$ (Proposition 10) adjusted to condition (16).

Remark 6 If $g = x^a h$ with h irreducible, we have an isomorphism of rings

$$\frac{R}{Rg} \cong \frac{R}{Rx^a} \times \frac{R}{Rh}.$$

The first factor is a non-commutative serial ring of length a, while the second factor is isomorphic to a matrix ring with coefficients in a field extension of K. So, the group of units of R/Rg is a product of the group of units of a field (if $a > 0$) and a general linear group over a field extension of K.

By Proposition 11 we get the inequality

$$n \leq \mu(p^\delta - 1) = \frac{dm}{\delta}\left(p^\delta - 1\right). \tag{18}$$

So, given $n, t, q = p^d$, we want to find m, δ such that

$$\max\left\{\frac{n}{10t}, \frac{n\delta}{d(p^\delta - 1)}\right\} \leq m \leq \frac{n}{4t} \text{ and } \delta \mid dm. \tag{19}$$

Our proposal of a McEliece cryptosystem follows the dual version of Niederreiter [19], by means of a Key Encapsulations Mechanism like the one proposed in [1].

5.1 Key schedule

The input is $n \gg t$ and $F = \mathbb{F}_q$ with $q = p^d$.

5.1.1 Construction of additional parameters

In order to generate the public and private keys for a McEliece type cryptosystem, the parameters m and s have to be found. These can be computed randomly via an exhaustive search to find pairs (m, δ) satisfying (19) and then looking for an s such that $\delta = \gcd(s, dm)$. For instance, if $n = 4096, t = 25, q = p^d = 2$, we get the following combinations:

m	24	26	28	30	32	33	34	36	36	38	39	40
δ	12	13	14	15	16	11	17	12	18	19	13	20

plus those cases $m = \delta$, which correspond to classical Goppa codes. If $n = 2560, t = 22, q = p^d = 2^4$, we get 83 different combinations, among them 18 are classical Goppa codes corresponding the case $\delta = dm$, where $12 \leq m \leq 29$ and $12 \leq \delta \leq 116$.

We set $k = n - 2t \left\lfloor \frac{n}{4t} \right\rfloor$, the smallest possible dimension, according to (19). Next pick randomly $1 \leq s \leq dm$, and let $\delta = \gcd(s, dm)$, $\mu = \frac{dm}{\delta}$, $L = \mathbb{F}_{q^m}$, $K = \mathbb{F}_{p^\delta}$ and $\sigma = \tau^s : L \to L$. Fix a basis of L over F and denote $\mathfrak{v} : L \to F^m$ the map providing the coordinates with respect to this basis. Let also denote $R = L[x; \sigma]$.

5.1.2 Left P-independent set

The set of position points may be selected amongst the points in a maximal left P-independent set as computed in Example 1. So we need a normal basis and a primitive element of L.

Let first compute a normal basis of L over K. We point out that

$$\{\alpha, \sigma(\alpha), \ldots, \sigma^{\mu-1}(\alpha)\} = \{\alpha, \tau^\delta(\alpha), \ldots, \tau^{\delta(\mu-1)}(\alpha)\}$$

since both τ^δ and σ are generators of the cyclic Galois group of the field extension L of K.

For each $\phi \in K[z]$, let $\varphi_{p^\delta}(\phi)$ be the number of polynomials in the indeterminate z of degree smaller than $\deg \phi$ and relatively prime to ϕ. It is well known, see [17, Theorem 3.73], that $\varphi_{p^\delta}(z^\mu - 1)$ is the number of $\alpha \in L$ such that $\{\alpha, \sigma(\alpha), \ldots, \sigma^{\mu-1}(\alpha)\}$ is a normal basis. By [5, Theorem 2],

$$\varphi_{p^\delta}(z^\mu - 1) \geq \frac{p^{\delta\mu}}{e\left\lceil \log_{p^\delta} \mu \right\rceil},$$

so the probability ρ of picking randomly an element which generates a normal basis is bounded from below by

$$\rho \geq \frac{1}{e\left\lceil \log_{p^\delta} \mu \right\rceil}.$$

So, a random search should produce a normal element in a very few attempts. For instance, the probability of choosing randomly an element which generates a normal basis when $n = 4096, t = 25, q = p^d = 2$ or $n = 2560, t = 22, q = p^d = 2^4$ is $\rho \geq 0.36$.

It remains to provide a fast method to check if an element generates a normal basis. There are quite enough methods to do that for finite fields, see e.g. [12, 24], where randomized algorithms in $\mathcal{O}(\mu^2 + \mu \log p^\delta)$ and $\mathcal{O}(\mu^{1.82} \log p^\delta)$, respectively, are provided. In our experiments we have just used the classical Hensel test, see [10] or [17, Theorem 2.39], which says that, for a given $\alpha \in L = \mathbb{F}_{p^{dm}}$, $\{\alpha, \alpha^{p^\delta}, \dots, \alpha^{p^{(\mu-1)\delta}}\}$ is a normal basis if and only if

$$\gcd\left(z^\mu - 1, \alpha z^{\mu-1} + \alpha^{p^\delta} z^{\mu-2} + \cdots + \alpha^{p^{(\mu-1)\delta}}\right) = 1.$$

A similar analysis can be done for primitive elements. As mentioned in the introduction of [21], all known algorithms to compute primitive elements work in two steps: compute a reasonable small subset containing a primitive element and test all elements of this subset until a primitive elements is found. Since the number of primitive elements in L is $\varphi(\mid L \mid -1) = \varphi(p^{dm} - 1)$ and, by [9, Theorem 328], $\varphi(p^{dm} - 1)/(p^{dm} - 1)$ is asymptotically bounded from below by a constant multiple of $\log\log(p^{dm} - 1)$, a random search would produce quite fast a primitive element. For instance, in case $n = 4096$, $t = 25$, $q = p^d = 2$, this lower bound is always greater than 0.168, or, in case $n = 2560$, $t = 22$, $q = p^d = 2^4$, than 0.127.

Testing if a randomly chosen $\gamma \in L$ is primitive can be done with the classical equivalence

$$\gamma \text{ is primitive } \iff \gamma^{\frac{p^{dm}-1}{p_i}} \neq 1 \text{ for all prime factor } p_i \text{ of } p^{dm} - 1$$

which requires factoring $p^{dm} - 1$. Since $p^{dm} - 1$ is reasonably small, this can also be done efficiently.

Once a primitive element γ and a normal element α have been computed, a maximal set of left P-independent elements is

$$\mathsf{P} = \left\{\gamma^i \frac{\sigma^{j+1}(\alpha)}{\sigma^j(\alpha)} \mid 0 \le i \le p^\delta - 2, 0 \le j \le \mu - 1\right\}.$$

In the classical case $\delta = dm$, being P-independent just means different position points.

5.1.3 Position points, skew Goppa polynomial and parity check polynomials

The list E of position points is obtained by a random selection of n points in P. Observe that we have chosen the parameter to have $n \le\mid \mathsf{P} \mid$.

$$\mathsf{E} = \{\alpha_0, \dots, \alpha_{n-1}\} \subseteq \mathsf{P}.$$

For the skew Goppa polynomial, we randomly choose a monic polynomial $h(y) \in K[y]$ without roots in K, see Proposition 10, such that $\deg_y(h) = \lfloor 2t/\mu \rfloor$ and set $g = h(x^\mu)x^{2t \bmod \mu}$, which has degree $2t$.

Finally, the REEA allow to compute $h_0, \dots, h_{n-1} \in R$ such that, for each $0 \le i \le n-1$, $\deg(h_i) < 2t$ and

$$(x - \alpha_i)h_i - 1 \in Rg.$$

In fact $\deg(h_i) = 2t - 1$ by a degree argument.

5.1.4 Parity check matrix and public key

By Proposition 7, a parity check matrix for the skew Goppa code is

$$H = \left(\mathfrak{v}(\sigma^{-j}(h_{i,j})\eta_i)\right)_{\substack{0 \le j \le 2t-1 \\ 0 \le i \le n-1}} \in F^{2tm \times n}$$

where $h_i = \sum_{j=0}^{2t-1} h_{i,j} x^j$. Once H is computed, the public key of our cryptosystem can be calculated as follows: set $k = n - 2t \left\lfloor \frac{n}{4t} \right\rfloor$, $r_H = \operatorname{rank}(H)$ and $A \in F^{(n-k-r_H)\times n}$, a random full rank matrix. The matrix H_{pub} is formed by the non zero rows of the reduced row echelon form of the block matrix $\left(\frac{H}{A}\right)$. If H_{pub} has less that $n - k$ rows, pick a new A. This H_{pub} defines a linear subcode of $\mathcal{C}$ of dimension k.

After this Key Schedule in the Key Encapsulation Mechanism, the different values remain as follows:

Parameters: $t \ll n$, $q = p^d$ and $k = n - 2t \left\lfloor \frac{n}{4t} \right\rfloor$.
Public key: $H_{\text{pub}} \in F^{(n-k)\times n}$.
Private key: $L, \sigma, \mathsf{E} = \{\alpha_0, \ldots, \alpha_{n-1}\}$, g and $h_0, \ldots, h_{n-1}$.

Remark 7 The security of this system is limited by the strength of information-set decoding attacks. From this point of view, the size of the public key has to be large enough to avoid those kind of attacks. Therefore the key size cannot to be smaller than the ones in the classic McEliece's cryptosystem. However, there are interesting sets of parameters such that the family of proper skew Goppa codes is larger than the classical ones. For instance, if we pick the parameters, $n = 6960$, $t = 119$, $p^d = 2$, there are around 2^{85347} classical binary Goppa codes. This number can be obtained by means of the Gauss formula which computes the number of monic irreducible polynomials over $\mathbb{F}_q$ of degree t, see e. g. [17, Theorem 3.25]. For these parameters there are three possible values for (m, δ), concretely $(24, 12)$, $(26, 13)$, $(28, 14)$, which can be used to build skew Goppa codes. In all cases $h = \delta$ and $\mu = 2$. Fixing a normal element, a primitive element and the corresponding maximal set of P-independent elements, the number of skew Goppa codes can be bounded from below by 2^{85236}, 2^{96470} and 2^{104922}, respectively. If the alphabet $\mathbb{F}_{p^d}$ is larger, there are usually more options to build skew Goppa codes. For instance, the parameters $n = 2560$, $t = 22$, $p^d = 2^4$ allow to build around 2^{29722} classical Goppa codes with $m = 3$. According to 5.1.1 there are 65 pairs (m, δ) which we can use to build skew Goppa codes. Each one of these choices has at least 2^{64305} skew Goppa codes on average.

5.2 Encryption and decryption procedures

The encryption process goes as follows. We pick a random error vector, i.e. $e \in F^n$ such that $\mathrm{w}(e) = t$, with corresponding error polynomial $e(x) = \sum_{j=1}^{t} e_j x^{k_j}$, and $0 \leq k_1 < k_2 < \cdots < k_t \leq n - 1$. The sender can easily derive a shared secret key from e by means of a fixed and publicly known hash function $\mathcal{H}$. The cryptogram is

$$c = eH_{\text{pub}}^{\mathrm{T}} \in F^{n-k}.$$

In order to decrypt, the receiver can easily compute $y \in F^n$ such that

$$c = yH_{\text{pub}}^{\mathrm{T}},$$

since H_{pub} is in row reduced echelon form. Algorithms 2 and 3 can be applied to y in order to compute e. Then the shared secret key can be retrieved by the receiver as $\mathcal{H}(e)$.

5.3 Examples

Next, we give some concrete examples. All the computations have been done with aid of the computational system SageMath [22].

Example 2 Let us describe here a toy-example showing an execution of our cryptosystem. Let $F = \mathbb{F}_{16} = \frac{\mathbb{F}_2[a]}{\langle a^4+a+1\rangle}$ be the field with 2^4 elements such that $a^4 + a + 1 = 0$. The elements of F may be represented by a hexadecimal character. For instance, $a^3 + a + 1 = 1011 = \mathtt{B}$. Set $n = 16$ and $t = 2$. In this case, we can only consider $m = 2$ and $\delta = 4$. Then $\mu = 2$ and we choose randomly $s = 4$.

Let $L = \frac{F[b]}{\langle b^2+\mathtt{F}b+\mathtt{B}\rangle}$, and consider $\{1, b\}$ a basis as F-vector space. We choose randomly $\gamma = \mathtt{B}b + 2$, a primitive element in L, and $\alpha = 9b + 8 \in L$, an element that generates a normal basis. We also choose randomly 16 position points in L,

$$\begin{array}{rrrr}
4\alpha_0 = 4b + 5 & \alpha_1 = 1b + \mathtt{F} & \alpha_2 = 8b + 3 & \alpha_3 = 3b + \mathtt{D} \\
\alpha_4 = 3b + 7 & \alpha_5 = 9 & \alpha_6 = 8b + \mathtt{B} & \alpha_7 = 3b + \mathtt{A} \\
\alpha_8 = 4b + 9 & \alpha_9 = 5b + 2 & \alpha_{10} = \mathtt{C}b + 6 & \alpha_{11} = 7b + 6 \\
\alpha_{12} = 2b + 4 & \alpha_{13} = \mathtt{A}b + \mathtt{B} & \alpha_{14} = \mathtt{C}b + 1 & \alpha_{15} = 1b + 1,
\end{array}$$

and 16 non-zero elements in L,

$$\begin{array}{rrrr}
4\eta_0 = \mathtt{F}b + \mathtt{D} & \eta_1 = 5b + \mathtt{F} & \eta_2 = 1b + 9 & \eta_3 = 3b + 4 \\
\eta_4 = 3b + 4 & \eta_5 = 1b + \mathtt{D} & \eta_6 = 4b + \mathtt{F} & \eta_7 = 7b + \mathtt{B} \\
\eta_8 = 7b & \eta_9 = 2b + 8 & \eta_{10} = \mathtt{D}b + \mathtt{F} & \eta_{11} = 9b + 7 \\
\eta_{12} = 2b + 6 & \eta_{13} = \mathtt{A}b + \mathtt{B} & \eta_{14} = 3b + 6 & \eta_{15} = \mathtt{A}b + 8.
\end{array}$$

We choose randomly the Goppa polynomial

$$g = x^4 + 7x^2 + 9 \in L[x; \tau^4]$$

which allows the calculation of the parity check polynomials

$$\begin{aligned}
h_0 &= 2x^3 + (8b + \mathtt{B})\, x^2 \\
h_1 &= \mathtt{D}x^3 + \mathtt{D}bx^2 + 3x + 3b \\
h_2 &= \mathtt{D}x^3 + (2b + 9)\, x^2 + 3x + \mathtt{B}b + 6 \\
h_3 &= 8x^3 + (\mathtt{B}b + 1)\, x^2 + 8x + \mathtt{B}b + 1 \\
h_4 &= 3x^3 + (5b + \mathtt{F})\, x^2 + x + 3b + 5 \\
h_5 &= 9x^3 + \mathtt{D}x^2 + 5x + \mathtt{B} \\
h_6 &= \mathtt{F}x^3 + (b + \mathtt{C})\, x^2 + x + 8b + \mathtt{A} \\
h_7 &= 3x^3 + (5b + \mathtt{B})\, x^2 + 8x + \mathtt{B}b + \mathtt{C} \\
h_8 &= 3x^3 + \mathtt{C}bx^2 + 8x + 6b \\
h_9 &= 9x^3 + (\mathtt{B}b + 2)\, x^2 + 5x + 2b + 7 \\
h_{10} &= 8x^3 + (\mathtt{A}b + 9)\, x^2 + 5x + 9b + 3 \\
h_{11} &= 2x^3 + (\mathtt{E}b + 9)\, x^2 \\
h_{12} &= \mathtt{F}x^3 + (\mathtt{D}b + \mathtt{E})\, x^2 + \mathtt{A}x + 7b + 5 \\
h_{13} &= \mathtt{B}x^3 + (2b + 4)\, x^2 + \mathtt{A}x + 8b + 3 \\
h_{14} &= 9x^3 + (6b + \mathtt{D})\, x^2 + \mathtt{F}x + 8b + \mathtt{E} \\
h_{15} &= \mathtt{E}x^3 + (\mathtt{E}b + \mathtt{B})\, x^2 + \mathtt{F}x + \mathtt{F}b + 5.
\end{aligned}$$

Hence, a parity check matrix is given by

$$\left(\begin{matrix} \mathtt{0} & \mathtt{8}b+\mathtt{3} & \mathtt{9}b+\mathtt{A} & \mathtt{C}b+\mathtt{C} & \mathtt{5}b+\mathtt{6} & \mathtt{B}b+\mathtt{6} & \mathtt{B}b+\mathtt{3} & \mathtt{2}b+\mathtt{7} \\ \mathtt{0} & \mathtt{F}b+\mathtt{2} & \mathtt{3}b+\mathtt{8} & \mathtt{B}b+\mathtt{6} & \mathtt{3}b+\mathtt{4} & \mathtt{5}b+\mathtt{C} & \mathtt{4}b+\mathtt{F} & \mathtt{D}b+\mathtt{7} \\ \mathtt{E}b+\mathtt{D} & \mathtt{F}b+\mathtt{D} & \mathtt{5}b+\mathtt{8} & \mathtt{C}b+\mathtt{C} & \mathtt{F}b+\mathtt{A} & \mathtt{D}b+\mathtt{E} & \mathtt{3}b+\mathtt{2} & \mathtt{4}b+\mathtt{E} \\ \mathtt{D}b+\mathtt{9} & \mathtt{C}b+\mathtt{7} & \mathtt{D}b+\mathtt{F} & \mathtt{B}b+\mathtt{6} & \mathtt{5}b+\mathtt{C} & \mathtt{9}b+\mathtt{F} & \mathtt{9}b+\mathtt{A} & \mathtt{9}b+\mathtt{E} \end{matrix}\right.$$

$$\left.\begin{matrix} \mathtt{F}b+\mathtt{B} & \mathtt{4}b+\mathtt{7} & \mathtt{1} & \mathtt{0} & \mathtt{E}b+\mathtt{5} & \mathtt{D} & \mathtt{7}b+\mathtt{B} & \mathtt{D}b+\mathtt{3} \\ \mathtt{D}b & \mathtt{A}b+\mathtt{E} & \mathtt{C}b+\mathtt{6} & \mathtt{0} & \mathtt{7}b+\mathtt{9} & \mathtt{8}b+\mathtt{2} & \mathtt{2}b+\mathtt{4} & \mathtt{C}b+\mathtt{1} \\ \mathtt{D}b+\mathtt{5} & \mathtt{5}b+\mathtt{2} & \mathtt{7} & \mathtt{A}b+\mathtt{E} & \mathtt{9}b+\mathtt{E} & \mathtt{E} & \mathtt{F}b+\mathtt{A} & \mathtt{F}b+\mathtt{8} \\ \mathtt{9}b & b+\mathtt{4} & \mathtt{2}b+\mathtt{1} & b+\mathtt{E} & \mathtt{D}b+\mathtt{4} & \mathtt{2}b+\mathtt{9} & \mathtt{8}b+\mathtt{3} & \mathtt{6}b+\mathtt{9} \end{matrix}\right),$$

whose expansion with coefficients in F is given by

$$H = \begin{pmatrix} \mathtt{0} & \mathtt{3} & \mathtt{A} & \mathtt{C} & \mathtt{6} & \mathtt{6} & \mathtt{3} & \mathtt{7} & \mathtt{B} & \mathtt{7} & \mathtt{1} & \mathtt{0} & \mathtt{5} & \mathtt{D} & \mathtt{B} & \mathtt{3} \\ \mathtt{0} & \mathtt{8} & \mathtt{9} & \mathtt{C} & \mathtt{5} & \mathtt{B} & \mathtt{B} & \mathtt{2} & \mathtt{F} & \mathtt{4} & \mathtt{0} & \mathtt{0} & \mathtt{E} & \mathtt{0} & \mathtt{7} & \mathtt{D} \\ \mathtt{0} & \mathtt{2} & \mathtt{8} & \mathtt{6} & \mathtt{4} & \mathtt{C} & \mathtt{F} & \mathtt{7} & \mathtt{0} & \mathtt{E} & \mathtt{6} & \mathtt{0} & \mathtt{9} & \mathtt{2} & \mathtt{4} & \mathtt{1} \\ \mathtt{0} & \mathtt{F} & \mathtt{3} & \mathtt{B} & \mathtt{3} & \mathtt{5} & \mathtt{4} & \mathtt{D} & \mathtt{D} & \mathtt{A} & \mathtt{C} & \mathtt{0} & \mathtt{7} & \mathtt{8} & \mathtt{2} & \mathtt{C} \\ \mathtt{D} & \mathtt{D} & \mathtt{8} & \mathtt{C} & \mathtt{A} & \mathtt{E} & \mathtt{2} & \mathtt{E} & \mathtt{5} & \mathtt{2} & \mathtt{7} & \mathtt{E} & \mathtt{E} & \mathtt{E} & \mathtt{A} & \mathtt{8} \\ \mathtt{E} & \mathtt{F} & \mathtt{5} & \mathtt{C} & \mathtt{F} & \mathtt{D} & \mathtt{3} & \mathtt{4} & \mathtt{D} & \mathtt{5} & \mathtt{0} & \mathtt{A} & \mathtt{9} & \mathtt{0} & \mathtt{F} & \mathtt{F} \\ \mathtt{9} & \mathtt{7} & \mathtt{F} & \mathtt{6} & \mathtt{C} & \mathtt{F} & \mathtt{A} & \mathtt{E} & \mathtt{0} & \mathtt{4} & \mathtt{1} & \mathtt{E} & \mathtt{4} & \mathtt{9} & \mathtt{3} & \mathtt{9} \\ \mathtt{D} & \mathtt{C} & \mathtt{D} & \mathtt{B} & \mathtt{5} & \mathtt{9} & \mathtt{9} & \mathtt{9} & \mathtt{9} & \mathtt{1} & \mathtt{2} & \mathtt{1} & \mathtt{D} & \mathtt{2} & \mathtt{8} & \mathtt{6} \end{pmatrix}.$$

Now, we compute the public key. Since $k = 8$ and the rank of H is $n - k = 8$, no additional random row is needed, so the public key H_{pub} is simply the row reduced echelon form of H, that is,

$$H_{\mathrm{pub}} = \begin{pmatrix} \mathtt{1} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{C} & \mathtt{2} & \mathtt{3} & \mathtt{2} & \mathtt{9} & \mathtt{9} & \mathtt{A} & \mathtt{4} \\ \mathtt{0} & \mathtt{1} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{9} & \mathtt{1} & \mathtt{A} & \mathtt{C} & \mathtt{7} & \mathtt{3} & \mathtt{8} & \mathtt{6} \\ \mathtt{0} & \mathtt{0} & \mathtt{1} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{A} & \mathtt{8} & \mathtt{2} & \mathtt{4} & \mathtt{D} & \mathtt{6} & \mathtt{5} & \mathtt{B} \\ \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{1} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{9} & \mathtt{0} & \mathtt{1} & \mathtt{3} & \mathtt{B} & \mathtt{7} & \mathtt{8} & \mathtt{9} \\ \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{1} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{E} & \mathtt{9} & \mathtt{B} & \mathtt{C} & \mathtt{F} & \mathtt{2} & \mathtt{6} & \mathtt{6} \\ \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{1} & \mathtt{0} & \mathtt{0} & \mathtt{A} & \mathtt{B} & \mathtt{1} & \mathtt{6} & \mathtt{9} & \mathtt{1} & \mathtt{1} & \mathtt{5} \\ \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{1} & \mathtt{0} & \mathtt{3} & \mathtt{F} & \mathtt{1} & \mathtt{2} & \mathtt{F} & \mathtt{B} & \mathtt{E} & \mathtt{1} \\ \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{1} & \mathtt{A} & \mathtt{D} & \mathtt{8} & \mathtt{6} & \mathtt{4} & \mathtt{1} & \mathtt{2} & \mathtt{B} \end{pmatrix}.$$

We select now the shared secret, a vector $e \in F^{16}$ with $t = 2$ non-zero components,

$$(\mathtt{4},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{C},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ 0).$$

In this case the non-zero components correspond to the positions 0 and 9. We encrypt the secret by multiplying by the transpose of H_{pub} obtaining a cyphertext

$$c = (\mathtt{F},\ \mathtt{C},\ \mathtt{A},\ \mathtt{0},\ \mathtt{6},\ \mathtt{D},\ \mathtt{8},\ \mathtt{3}) \in F^8.$$

The receiver solves the linear system $c = yH_{\mathrm{pub}}^{\mathrm{T}}$ obtaining, for instance,

$$y = (\mathtt{F},\ \mathtt{C},\ \mathtt{A},\ \mathtt{0},\ \mathtt{6},\ \mathtt{D},\ \mathtt{8},\ \mathtt{3},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ \mathtt{0},\ 0).$$

Finally, applying the decoding algorithm in Algorithm 2 to y, we find the vector e, decrypting the secret.

Example 3 This example shows that decoding failures, although quite unusual, can happen. Let $F = \mathbb{F}_{2^8} = \frac{\mathbb{F}_2[z]}{\langle z^8+z^4+z^3+z^2+1\rangle}$, $n = 16$, and $t = 2$. The possible values for the pair (m, δ) are $(1, 4)$, $(2, 1)$, $(2, 4)$ and $(2, 8)$. We fix then $m = 1$ and $\delta = 4$. Choose the automorphism

$\sigma : L \to L$ defined by $\sigma(a) = a^{2^4}$. So, $\mu = 2$ and $K = \mathbb{F}_{2^4}$, which may be presented as $K = \frac{\mathbb{F}_2[w]}{\langle w^4+w+1\rangle}$, with embedding $w \mapsto z^{34}$ into F. As a consequence, $k = n - 2t\lfloor \frac{n}{4t} \rfloor = 8$, the smallest dimension for this set of given parameters. The chosen normal and primitive elements are $\alpha = z^{37}$ and $\gamma = z^{41}$.

The list $\mathsf{E} = \{\alpha_0, \dots, \alpha_{15}\}$ of evaluation points contains the elements

$$\begin{aligned}
\alpha_0 &= \gamma^0 \tfrac{\sigma(\alpha)}{\alpha} = z^{45}, & \alpha_1 &= \gamma^9 \tfrac{\sigma(\alpha)}{\alpha} = z^{159},\\
\alpha_2 &= \gamma^{13} \tfrac{\sigma(\alpha)}{\alpha} = z^{68}, & \alpha_3 &= \gamma^{13} \tfrac{\sigma^2(\alpha)}{\sigma(\alpha)} = z^{233},\\
\alpha_4 &= \gamma^{10} \tfrac{\sigma^2(\alpha)}{\sigma(\alpha)} = z^{110}, & \alpha_5 &= \gamma^7 \tfrac{\sigma(\alpha)}{\alpha} = z^{77},\\
\alpha_6 &= \gamma^{12} \tfrac{\sigma(\alpha)}{\alpha} = z^{27}, & \alpha_7 &= \gamma^{10} \tfrac{\sigma(\alpha)}{\alpha} = z^{200},\\
\alpha_8 &= \gamma^2 \tfrac{\sigma^2(\alpha)}{\sigma(\alpha)} = z^{37}, & \alpha_9 &= \gamma^0 \tfrac{\sigma^2(\alpha)}{\sigma(\alpha)} = z^{210},\\
\alpha_{10} &= \gamma^6 \tfrac{\sigma^2(\alpha)}{\sigma(\alpha)} = z^{201}, & \alpha_{11} &= \gamma^3 \tfrac{\sigma(\alpha)}{\alpha} = z^{168},\\
\alpha_{12} &= \gamma^{11} \tfrac{\sigma^2(\alpha)}{\sigma(\alpha)} = z^{151}, & \alpha_{13} &= \gamma^2 \tfrac{\sigma(\alpha)}{\alpha} = z^{127},\\
\alpha_{14} &= \gamma^1 \tfrac{\sigma^2(\alpha)}{\sigma(\alpha)} = z^{251}, & \alpha_{15} &= \gamma^{12} \tfrac{\sigma^2(\alpha)}{\sigma(\alpha)} = z^{192}.
\end{aligned}$$

Let $g = x^4 + z^{238}x^2 + z^{68}$ be the skew Goppa polynomial. The corresponding parity check polynomials are

$$\begin{aligned}
h_0 &= z^{136}x^3 + z^{91}x^2 + z^{187}x + z^{142}, & h_1 &= z^{68}x^3 + z^{62}x^2 + z^{136}x + z^{130},\\
h_2 &= z^{102}x^3 + z^{170}x^2 + z^{204}x + z^{17}, & h_3 &= z^{102}x^3 + z^{5}x^2 + z^{204}x + z^{107},\\
h_4 &= z^{85}x^3 + z^{60}x^2 + z^{34}x + z^{9}, & h_5 &= z^{238}x^3 + z^{195}x^2 + z^{204}x + z^{161},\\
h_6 &= z^{85}x^3 + z^{7}x^2 + z^{170}x + z^{92}, & h_7 &= z^{85}x^3 + z^{225}x^2 + z^{34}x + z^{174},\\
h_8 &= z^{170}x^3 + z^{252}x^2 + z^{187}x + z^{14}, & h_9 &= z^{136}x^3 + z^{181}x^2 + z^{187}x + z^{232},\\
h_{10} &= z^{102}x^3 + z^{3}x^2 + z^{238}x + z^{139}, & h_{11} &= z^{136}x^3 + z^{19}x^2 + z^{136}x + z^{19},\\
h_{12} &= z^{170}x^3 + z^{36}x^2 + z^{34}x + z^{155}, & h_{13} &= z^{170}x^3 + z^{162}x^2 + z^{187}x + z^{179},\\
h_{14} &= z^{51}x^3 + z^{242}x^2 + z^{221}x + z^{157}, & h_{15} &= z^{85}x^3 + z^{97}x^2 + z^{170}x + z^{182}.
\end{aligned}$$

From these parity check polynomials, we may compute the matrix $H \in F^{4\times 16}$. Since H has rank 4, according to Sect. 5.1.4, we append to H a random matrix in $F^{4\times 16}$, whose row reduced echelon form yields the following public key matrix

$$H_{\text{pub}} = \begin{pmatrix}
1&0&0&0&0&0&0&0&z^{142}&z^{92}&z^{126}&z^{156}&z^{187}&z^{178}&z^{234}&z^{88}\\
0&1&0&0&0&0&0&0&z^{73}&z^{103}&z^{157}&z^{113}&z^{188}&z^{253}&z^{222}&z^{152}\\
0&0&1&0&0&0&0&0&z^{109}&z^{109}&z^{64}&z^{165}&z^{131}&z^{204}&z^{138}&z^{145}\\
0&0&0&1&0&0&0&0&z^{180}&z^{78}&z^{202}&z^{230}&z^{82}&z^{81}&z^{185}&z^{224}\\
0&0&0&0&1&0&0&0&z^{70}&z^{247}&z^{51}&z^{65}&z^{49}&z^{162}&z^{111}&z^{36}\\
0&0&0&0&0&1&0&0&z^{119}&z^{236}&z^{50}&z^{243}&z^{136}&z^{56}&z^{133}&z^{225}\\
0&0&0&0&0&0&1&0&z^{89}&z^{172}&z^{152}&z^{209}&z^{234}&z^{22}&z^{231}&z^{96}\\
0&0&0&0&0&0&0&1&z^{70}&z^{152}&z^{157}&z^{32}&z^{247}&z^{180}&z^{172}&z^{106}
\end{pmatrix}$$

Let the error vector be

$$e = \left(z^{249}, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0\right).$$

Hence the cryptogram is

$$c = eH_{\text{pub}}^{\mathrm{T}} = \left(z^{133}, z^{103}, z^{109}, z^{78}, z^{247}, z^{236}, z^{172}, z^{152})\right),$$

which is transmitted to the receiver. A solution of

$$c = yH_{\mathrm{pub}}^{\mathrm{T}}$$

is

$$y = \left(z^{133}, z^{103}, z^{109}, z^{78}, z^{247}, z^{236}, z^{172}, z^{152}, 0, 0, 0, 0, 0, 0, 0, 0\right),$$

which allows to compute the syndrome polynomial

$$s = z^{36}x^3 + z^{81}x^2 + z^{87}x + z^{132}.$$

The LEEA applied to g, s until we find the first remainder with degree below 2, and we get

$$v_I = z^{189}x + z^{174}, \quad r_I = z^{119}.$$

The only right root of v_I is z^{240}, which is not in E. Therefore, the cardinal of the evaluation points which are roots of v_I is $0 < 1 = \deg v_I$. There is a decoding failure which we can solve with Algorithm 3. The 10th evaluation point, α_9, does not increment the degree, so it is a root of the locator polynomial λ. Since $[v_I, x - \alpha_9]_\ell = x^2 + 1$ which have α_0 and α_9 as roots, we get that $\lambda = x^2 + 1$. The corresponding evaluator polynomial is $\omega = z^{155}x + z^{200}$. The error positions are 0 and 9, and the evaluator polynomial allows to compute the error values, z^{249} and 1 respectively, as expected.

5.4 McEliece's original approach

The approach in [18] can also be followed. Private key is computed as in Subsection 5.1. The public key is obtained as follows: Let H be the matrix computed in 5.1.4 from Proposition 7. Compute a full rank generator matrix G for the left kernel of H^{T}. Let $S \in F^{r \times r}$ a random non singular matrix where $r = \operatorname{rank}(G)$. Then G_{pub} consists in the first k rows of SG. These concludes the key schedule.

The encryption procedure starts with a message which is a word $m \in F^k$. In order to encrypt, we select a random $e \in F^n$ such that $\mathrm{w}(e) = t$. The cryptogram is

$$y = mG_{\mathrm{pub}} + e \in F^n.$$

To decrypt, let $y(x) = \sum_{i=0}^{n-1} y_i x^i$. Apply Algorithms 2 and 3 in order to compute the vector e. Then the message can be recovered multiplying $y - e$ by a suitable right inverse of G_{pub}.

Acknowledgements Research funded by grant (PID2019-110525GB-I00 / AEI / 10.13039/501100011033) and by the IMAG-María de Maeztu grant (CEX2020-001105-M / AEI / 10.13039/501100011033)

Funding Funding for open access publishing: Universidad de Granada/CBUA

References

1. Albrecht M.R., Bernstein D.J., Chou T., Cid C., Gilcher J., Lange T., Maram V., von Maurich I., Misoczki R., Niederhagen R., Paterson K.G., Persichetti E., Peters C., Schwabe P., Sendrier N., Szefer J., Tjhai C.J., Tomlison M., Wang W.: Classic McEliece: conservative code-based cryptography. Tech. Report. NIST's Post-Quantum Cryptography Standardization Project 10 (2020). https://classic.mceliece.org/.
2. Bartz H., Jerkovitz T., Puchinger S., Rosenkilde J.: Fast decoding of codes in the rank, subspace, and sum-rank metric. IEEE Trans. Inform. Theory **67**, 5026–5050 (2021). https://doi.org/10.1109/TIT.2021.3067318.
3. Bueso J.L., Gómez-Torrecillas J., Verschoren A.: Algorithmic Methods in Non-commutative Algebra. Applications to Quantum groups. Springer, Dordrecht (2003) https://doi.org/10.1007/978-94-017-0285-0.
4. Delenclos J., Leroy A.: Noncommutative symmetric functions and w-polynomials. J. Algebra Appl. **06**, 815–837 (2007). https://doi.org/10.1142/S021949880700251X.
5. Frandsen G.S.: On the density of normal bases in finite fields. Finite Fields Appl. **6**, 23–38 (2000). https://doi.org/10.1006/ffta.1999.0263.
6. Gómez-Torrecillas J., Lobillo F.J., Navarro G.: A Sugiyama-like decoding algorithm for convolutional codes. IEEE Trans. Inform. Theory **63**, 6216–6226 (2017). https://doi.org/10.1109/TIT.2017.2731774. arXiv:1607.07187.
7. Gómez-Torrecillas J., Lobillo F.J., Navarro G.: Computing the bound of an Ore polynomial. Applications to factorization. J. Symb. Comput. **92**, 269–297 (2019).
8. Gómez-Torrecillas J., Lobillo F.J., Navarro G.: Procedimiento y dispositivo de cifrado/descifrado post-cuántico usando códigos lineales (February 2022). OEPM, patente solicitud número P202230118.
9. Hardy G.H., Wright E.M.: An Introduction to the Theory of Numbers, 4th edn Oxford University Press, Oxford (1960).
10. Hensel K.: Ueber die darstellung der zahlen eines gattungsbereiches für einen beliebigen primdivisor. J. Reine Angew. Math. **103**, 230–237 (1888). https://doi.org/10.1515/crll.1888.103.230.
11. Jacobson N.: Finite-Dimensional Division Algebras over Fields. Springer, Berlin (1996) https://doi.org/10.1007/978-3-642-02429-0.
12. Kaltofen E., Shoup V.: Subquadratic-time factoring of polynomials over finite fields. Math. Comput. **67**, 1179–1197 (1998). https://doi.org/10.1090/S0025-5718-98-00944-2.
13. Lam T., Leroy A.: Vandermonde and Wronskian matrices over division rings. J. Algebra **119**, 308–336 (1988). https://doi.org/10.1016/0021-8693(88)90063-4.
14. Lam T., Leroy A.: Wedderburn polynomials over division rings, I. J. Pure Appl. Algebra **186**, 43–76 (2004). https://doi.org/10.1016/S0022-4049(03)00125-7.
15. Lam T.Y., Leroy A.: Algebraic conjugacy classes and skew polynomial rings. In: van Oystaeyen F., Le Bruyn L. (eds.) Perspectives in Ring Theory, pp. 153–203. Springer, Dordrecht (1988). https://doi.org/10.1007/978-94-009-2985-2_15.
16. Lang S.: Algebra. Graduate Texts in Mathematics, vol. 211, revised 3rd edn. Springer, New York (2002).
17. Lidl R., Niederreiter H.: Finite Fields. Encyclopedia of Mathematics and Its Applications, vol. 20. Cambridge University Press, Cambridge (1997).
18. McEliece R.J.: A public-key cryptosystem based on algebraic coding theory. Tech. Report 42-44. National Aeronautics and Space Administration, January and February (1978).
19. Niederreiter H.: Knapsack-type cryptosystems and algebraic coding theory. Probl. Control Inform. Theory **15**(2), 159–166 (1986).
20. Ore O.: Theory of non-commutative polynomials. Ann. Math. (2) **34**, 480–508 (1933). https://doi.org/10.2307/1968173.
21. Shparlinski I.E.: On constructing primitive roots in finite fields with advice. IEEE Trans. Inform. Theory **64**, 7132–7136 (2018). https://doi.org/10.1109/TIT.2018.2810938.
22. Stein W.A. et al.: Sage Mathematics Software (Version 9.6). The Sage Development Team (2021). www.sagemath.org.
23. von zur Gathen J., Gerhard J.: Modern Computer Algebra. Cambridge University Press, Cambridge (2003).
24. von zur Gathen J., Giesbrecht M.: Constructing normal bases in finite fields. J. Symbol. Comput. **10**, 547–570 (1990). https://doi.org/10.1016/S0747-7171(08)80158-7.
25. Wang, L.-P.: Linearized Goppa codes. In: 2018 IEEE International Symposium on Information Theory (ISIT), pp. 2496–2500 (2018). https://doi.org/10.1109/ISIT.2018.8437579