Compact McEliece Keys from Goppa Codes

Rafael Misoczki[1] and Paulo S. L. M. Barreto[1⋆]

Departamento de Engenharia de Computação e Sistemas Digitais (PCS),
Escola Politécnica, Universidade de São Paulo, Brazil.
{rmisoczki,pbarreto}@larc.usp.br

Abstract. The classical McEliece cryptosystem is built upon the class of Goppa codes, which remains secure to this date in contrast to many other families of codes but leads to very large public keys. Previous proposals to obtain short McEliece keys have primarily centered around replacing that class by other families of codes, most of which were shown to contain weaknesses, and at the cost of reducing in half the capability of error correction. In this paper we describe a simple way to reduce significantly the key size in McEliece and related cryptosystems using a subclass of Goppa codes, while also improving the efficiency of cryptographic operations to $\tilde{O}(n)$ time, and keeping the capability of correcting the full designed number of errors in the binary case.

1 Introduction

Quantum computers can potentially break most if not all conventional cryptosystems actually deployed in practice, namely, all systems based on the integer factorization problem (like RSA) or the discrete logarithm problem (like traditional or elliptic curve Diffie-Hellman and DSA, and also all of pairing-based cryptography).

Certain classical cryptosystems, inspired on computational problems of a nature entirely different from the above and potentially much harder to solve, remain largely unaffected by the threat of quantum computing, and have thus been called quantum-resistant or, more suggestively, 'post-quantum' cryptosystems. These include lattice-based cryptosystems and syndrome-based cryptosystems like McEliece [17] and Niederreiter [21]. Such systems usually have even a speed advantage over conventional schemes; for instance, both McEliece and Niederreiter encryption over a code of length n has time complexity $O(n^2)$, while Diffie-Hellman/DSA and (private exponent) RSA with n-bit keys have time complexity $O(n^3)$. On the other hand, they are plagued by very large keys compared to their conventional counterparts.

It is therefore of utmost importance to seek ways to reduce the key sizes for post-quantum cryptosystems while keeping their security level. The first steps

⋆ Supported by the Brazilian National Council for Scientific and Technological Development (CNPq) under research productivity grant 312005/2006-7 and universal grant 485317/2007-9, and by the Science Foundation Ireland (SFI) as E. T. S. Walton Award fellow under grant 07/W.1/I1824.

toward this goal were taken by Monico et al. using low density parity-check codes [20], by Gaborit using quasi-cyclic codes [10], and by Baldi and Chiaraluce using a combination of both [1].

However, these proposals were all shown to contain weaknesses [22]. In those proposals the trapdoor is protected essentially by no other means than a private permutation of the underlying code. The attack strategy consists of obtaining a solvable system of linear equations that the components of the permutation matrix must satisfy, and was successfully mounted due to the very constrained nature of the secret permutation (since it has to preserve the quasi-cyclic structure of the result) and the fact that the secret code is a subcode of a public code.

A dedicated fix to the problems in [1] is proposed in [2]. More recently, Berger et al. [3] showed how to circumvent the drawbacks of Gaborit's original scheme and remove the weaknesses pointed out in [22] by means of two techniques:

1. Extracting block-shortened public codes from very large private codes, exploiting Wieschebrink's theorem on the NP-completeness of distinguishing punctured codes [30];
2. Working with subfield subcodes over an intermediate subfield between the base field and the extension field of the original code.

These two techniques were successfully applied to quasi-cyclic codes, yet we will see that their applicability is not restricted to that class.

Our contribution: In this paper we propose the class of *quasi-dyadic* Goppa codes, which admit a very compact parity-check or a generator matrix representation, for efficiently instantiating syndrome-based cryptosystems. We stress that we are not proposing any new cryptosystem, but rather a technique to obtain efficient parameters and algorithms for such systems, current or future. In contrast to many other proposed families of codes [12, 13, 22, 27], Goppa codes have withstood cryptanalysis quite well, and despite considerable progress in the area [15, 26] (see also [6] for a survey) they remain essentially unscathed since they were suggested with the very first syndrome-based cryptosystem known, namely, the original McEliece scheme. Our method produces McEliece-type keys that are up to a factor $t = \tilde{O}(n)$ smaller than keys produced from generic t-error correcting Goppa codes of length n in characteristic 2. In the binary case it also retains the ability of correcting the full designed number of errors rather than just half as many, a feature that is missing in all previous attempts at constructing compact codes for cryptographic purposes, including [3]. Moreover, the complexity of all typical cryptographic operations become $\tilde{O}(n)$; specifically, under the common cryptographic setting $t = O(n/\lg n)$, code generation, encryption and decryption all have asymptotic complexity $O(n \lg n)$.

The remainder of this paper is organized as follows. Section 2 introduces some basic concepts of coding theory. In section 3 we describe our proposal of using binary Goppa codes in quasi-dyadic form, and how to build them. We consider hardness issues in Section 4, and efficiency issues, including guidelines on how to choose parameters, in Section 5. We conclude in Section 6.

2 Preliminaries

In what follows all vector and matrix indices are numbered from zero onwards.

Definition 1. *Given a ring $\mathcal{R}$ and a vector $h = (h_0, \ldots, h_{n-1}) \in \mathcal{R}^n$, the* dyadic *matrix $\Delta(h) \in \mathcal{R}^{n \times n}$ is the symmetric matrix with components $\Delta_{ij} = h_{i \oplus j}$ where $\oplus$ stands for bitwise exclusive-or on the binary representations of the indices. The sequence h is called its* signature. *The set of dyadic $n \times n$ matrices over $\mathcal{R}$ is denoted $\Delta(\mathcal{R}^n)$. Given $t > 0$, $\Delta(t, h)$ denotes $\Delta(h)$ truncated to its first t rows.*

One can recursively characterize a dyadic matrix when n is a power of 2: any 1×1 matrix is dyadic, and for $k > 0$ any $2^k \times 2^k$ dyadic matrix M has the form

$$M = \begin{bmatrix} A & B \\ B & A \end{bmatrix}$$

where A and B are $2^{k-1} \times 2^{k-1}$ dyadic matrices. It is not hard to see that the signature of a dyadic matrix coincides with its first row. Dyadic matrices form a commutative subring of $\mathcal{R}^{n \times n}$ as long as $\mathcal{R}$ is commutative [14].

Definition 2. *A* dyadic permutation *is a dyadic matrix $\Pi^i \in \Delta(\{0,1\}^n)$ whose signature is the i-th row of the identity matrix.*

A dyadic permutation is clearly an involution, i.e. $(\Pi^i)^2 = I$. The i-th row (or equivalently the i-th column) of the dyadic matrix defined by a signature h can be written $\Delta(h)_i = h\Pi^i$.

Definition 3. *A* quasi-dyadic *matrix is a (possibly non-dyadic) block matrix whose component blocks are dyadic submatrices.*

Quasi-dyadic matrices are at the core of our proposal. We will be mainly concerned with the case $\mathcal{R} = \mathbb{F}_q$, the finite field with q (a prime power) elements.

Definition 4. *Given two disjoint sequences $z = (z_0, \ldots, z_{t-1}) \in \mathbb{F}_q^t$ and $L = (L_0, \ldots, L_{n-1}) \in \mathbb{F}_q^n$ of distinct elements, the* Cauchy matrix *$C(z, L)$ is the $t \times n$ matrix with elements $C_{ij} = 1/(z_i - L_j)$, i.e.*

$$C(z, L) = \begin{bmatrix} \frac{1}{z_0 - L_0} & \cdots & \frac{1}{z_0 - L_{n-1}} \\ \vdots & \ddots & \vdots \\ \frac{1}{z_{t-1} - L_0} & \cdots & \frac{1}{z_{t-1} - L_{n-1}} \end{bmatrix}.$$

Cauchy matrices have the property that all of their submatrices are nonsingular [25]. Notice that, in general, Cauchy matrices are not dyadic and vice-versa, although the intersection of these two classes is non-empty in characteristic 2.

Definition 5. *Given $t > 0$ and a sequence $L = (L_0, \ldots, L_{n-1}) \in \mathbb{F}_q^n$, the* Vandermonde matrix $\mathrm{vdm}(t, L)$ *is the $t \times n$ matrix with elements $V_{ij} = L_j^i$.*

Definition 6. *Given a sequence $L = (L_0, \ldots, L_{n-1}) \in \mathbb{F}_q^n$ of distinct elements and a sequence $D = (D_0, \ldots, D_{n-1}) \in \mathbb{F}_q^n$ of nonzero elements, the* Generalized Reed-Solomon code $GRS_r(L, D)$ *is the $[n, k, r]$ linear error-correcting code defined by the parity-check matrix*

$$H = \mathrm{vdm}(r-1, L) \cdot \mathrm{diag}(D).$$

An alternant code *is a subfield subcode of a Generalized Reed-Solomon code.*

Let p be a prime power, let $q = p^d$ for some d, and let $\mathbb{F}_q = \mathbb{F}_p[x]/b(x)$ for some irreducible polynomial $b(x) \in \mathbb{F}_p[x]$ of degree d. Given a code specified by a parity-check matrix $H \in \mathbb{F}_q^{t \times n}$, the *trace construction* derives from it an $\mathbb{F}_p$-subfield subcode by writing the $\mathbb{F}_p$ coefficients of each $\mathbb{F}_q$ component of H onto d successive rows of a parity-check matrix $T_d(H) \in \mathbb{F}_p^{dt \times n}$ for the subcode. The related *co-trace* parity-check matrix $T'_d(H) \in \mathbb{F}_p^{dt \times n}$, equivalent to $T_d(H)$ by a left permutation, is obtained from H by writing the $\mathbb{F}_p$ coefficients of terms of equal degree from all components on a column of H onto successive rows of $T'_d(H)$.

Thus, given elements $u_i(x) = u_{i,0} + \cdots + u_{i,d-1}x^{d-1} \in \mathbb{F}_q = \mathbb{F}_p[x]/b(x)$, the trace construction maps a column $(u_0, \ldots, u_{t-1})^\mathsf{T}$ from H to the column $(u_{0,0}, \ldots, u_{0,d-1}; \ldots; u_{t-1,0}, \ldots, u_{t-1,d-1})^\mathsf{T}$ on the trace matrix $T_d(H)$, and to the column $(u_{0,0}, \ldots, u_{t-1,0}; \ldots; u_{0,d-1}, \ldots, u_{t-1,d-1})^\mathsf{T}$ on the co-trace matrix $T'_d(H)$.

Finally, one of the most important families of linear error-correcting codes for cryptographic purposes is that of Goppa codes:

Definition 7. *Given a prime power p, $q = p^d$ for some d, a sequence $L = (L_0, \ldots, L_{n-1}) \in \mathbb{F}_q^n$ of distinct elements and a polynomial $g(x) \in \mathbb{F}_q[x]$ of degree t such that $g(L_i) \neq 0$ for $0 \leqslant i < n$, the* Goppa code $\Gamma(L, g)$ *over $\mathbb{F}_p$ is the alternant code over $\mathbb{F}_p$ corresponding to $GRS_t(L, D)$ where $D = (g(L_0)^{-1}, \ldots, g(L_{n-1})^{-1})$, and its minimum distance is at least $2t + 1$.*

An irreducible Goppa code in characteristic 2 can correct up to t errors using Patterson's algorithm [23], or slightly more using Bernstein's list decoding method [5], and t errors can still be corrected by suitable decoding algorithms if the generator $g(x)$ is not irreducible[1]. In all other cases no algorithm is known that can correct more than $t/2$ errors (or just a few more).

3 Goppa codes in Cauchy and dyadic form

A property of Goppa codes that is central to our proposal is that they admit a parity-check matrix in Cauchy form:

[1] For instance, one can equivalently view the binary Goppa code as the alternant code defined by the generator polynomial $g^2(x)$, in which case any alternant decoder will decode t errors. We are grateful to Nicolas Sendrier for pointing this out.

Theorem 1 ([28]). *The Goppa code generated by a monic polynomial* $g(x) = (x - z_0) \dots (x - z_{t-1})$ *without multiple zeros admits a parity-check matrix of the form* $H = C(z, L)$*, i.e.* $H_{ij} = 1/(z_i - L_j)$*,* $0 \leqslant i < t$*,* $0 \leqslant j < n$*.*

This theorem (also appearing in [16, Ch. 12, §3, Pr. 5]) is entirely general when one considers the factorization of the Goppa polynomial over its splitting field, in which case a single root of g is enough to completely characterize the code. For simplicity, we will restrict our attention to the case where all roots of that polynomial are in the field $\mathbb{F}_q$ itself.

3.1 Building a binary Goppa code in dyadic form

We now show how to build a binary Goppa code that admits a parity-check matrix in dyadic form. To this end we seek a way to construct dyadic Cauchy matrices. The following theorem characterizes all matrices of this kind.

Theorem 2. *Let* $H \in \mathbb{F}_q^{n \times n}$ *with* $n > 1$ *be simultaneously a dyadic matrix* $H = \Delta(h)$ *for some* $h \in \mathbb{F}_q^n$ *and a Cauchy matrix* $H = C(z, L)$ *for two disjoint sequences* $z \in \mathbb{F}_q^n$ *and* $L \in \mathbb{F}_q^n$ *of distinct elements. Then* $\mathbb{F}_q$ *is a field of characteristic 2,* h *satisfies*

$$\frac{1}{h_{i \oplus j}} = \frac{1}{h_i} + \frac{1}{h_j} + \frac{1}{h_0}, \tag{1}$$

and $z_i = 1/h_i + \omega$*,* $L_j = 1/h_j + 1/h_0 + \omega$ *for some* $\omega \in \mathbb{F}_q$*.*

Proof. Since a dyadic matrix is symmetric, the sequences that define it must satisfy $1/(z_i - L_j) = 1/(z_j - L_i)$, hence $L_j = z_i + L_i - z_j$ for all i and j. Then $z_i + L_i$ must be a constant α, and taking $i = 0$ in particular this simplifies to $L_j = \alpha - z_j$. Substituting back into the definition $M_{ij} = 1/(z_i - L_j)$ one sees that $H_{ij} = 1/(z_i + z_j + \alpha)$. But dyadic matrices also have constant diagonal, namely, $H_{ii} = 1/(2z_i + \alpha) = h_0$. This is only possible if all z_i are equal (contradicting the definition of a Cauchy matrix), or else if the characteristic of the field is 2, as claimed.

In this case we see that $\alpha = 1/h_0$, and hence $H_{ij} = 1/(z_i + z_j + 1/h_0)$. Plugging in the definition $H_{ij} = h_{i \oplus j}$ we get $1/H_{ij} = 1/h_{i \oplus j} = z_i + z_j + 1/h_0$, and taking $j = 0$ in particular this yields $1/h_i = z_i + z_0 + 1/h_0$, or simply $z_i = 1/h_i + 1/h_0 + z_0$. Substituting back one obtains $1/h_{i \oplus j} = z_i + z_j + 1/h_0 = 1/h_i + 1/h_0 + z_0 + 1/h_j + 1/h_0 + z_0 + 1/h_0 = 1/h_i + 1/h_j + 1/h_0$, as expected.

Finally, define $\omega = 1/h_0 + z_0$ and substitute into the derived relations $z_i = 1/h_i + 1/h_0 + z_0$ and $L_j = \alpha - z_j$ to get $z_i = 1/h_i + \omega$ and $L_j = 1/h_j + 1/h_0 + \omega$, as desired. □

Therefore all we need is a method to solve Equation 1. The technique we propose consists of simply choosing distinct nonzero h_0 and h_i at random where i scans all powers of two smaller than n, and setting all other values as

$$h_{i+j} \leftarrow \frac{1}{\dfrac{1}{h_i} + \dfrac{1}{h_j} + \dfrac{1}{h_0}}$$

for $0 < j < i$ (so that $i + j = i \oplus j$), as long as this value is well-defined. Algorithm 1 captures this idea. Since each element of the signature h is assigned a value exactly once, its running time is $O(n)$ steps. The notation $u \xleftarrow{\$} U$ means that variable u is uniformly sampled at random from set U. For convenience we also define the *essence* of h to be the sequence $\eta_s = 1/h_{2^s} + 1/h_0$ for $s = 0, \ldots, \lceil \lg n \rceil - 1$ together with $\eta_{\lceil \lg n \rceil} = 1/h_0$, so that, for $i = \sum_{k=0}^{\lceil \lg n \rceil - 1} i_k 2^k$, $1/h_i = \eta_{\lceil \lg n \rceil} + \sum_{k=0}^{\lceil \lg n \rceil - 1} i_k \eta_k$.

Algorithm 1 Constructing a binary Goppa code in dyadic form

INPUT: q (a power of 2), $n \leqslant q/2$, t.

OUTPUT: Support L, generator polynomial g, dyadic parity-check matrix H for a binary Goppa code $\Gamma(L, g)$ of length n and design distance $2t + 1$ over $\mathbb{F}_q$, and the essence η of the signature of H.

$U \leftarrow \mathbb{F}_q \setminus \{0\}$
▷ Choose the dyadic signature $(h_0, \ldots, h_{n-1})$. N.B. Whenever h_j with $j > 0$ is taken from U, so is $1/(1/h_j + 1/h_0)$ to prevent a potential spurious intersection between z and L.
$h_0 \xleftarrow{\$} U$
$\eta_{\lceil \lg n \rceil} \leftarrow 1/h_0$
$U \leftarrow U \setminus \{h_0\}$
for $s \leftarrow 0$ **to** $\lceil \lg n \rceil - 1$ **do**
    $i \leftarrow 2^s$
    $h_i \xleftarrow{\$} U$
    $\eta_s \leftarrow 1/h_i + 1/h_0$
    $U \leftarrow U \setminus \{h_i,\ 1/(1/h_i + 1/h_0)\}$
    for $j \leftarrow 1$ **to** $i - 1$ **do**
        $h_{i+j} \leftarrow 1/(1/h_i + 1/h_j + 1/h_0)$
        $U \leftarrow U \setminus \{h_{i+j},\ 1/(1/h_{i+j} + 1/h_0)\}$
    end for
end for
$\omega \xleftarrow{\$} \mathbb{F}_q$
▷ Assemble the Goppa generator polynomial:
for $i \leftarrow 0$ **to** $t - 1$ **do**
    $z_i \leftarrow 1/h_i + \omega$
end for
$g(x) \leftarrow \prod_{i=0}^{t-1} (x - z_i)$
▷ Compute the support:
for $j \leftarrow 0$ **to** $n - 1$ **do**
    $L_j \leftarrow 1/h_j + 1/h_0 + \omega$
end for
$h \leftarrow (h_0, \ldots, h_{n-1})$
$H \leftarrow \Delta(t, h)$
return L, g, H, η

Theorem 3. *Algorithm 1 produces up to* $\prod_{i=0}^{\lceil \lg n \rceil} (q - 2^i)$ *Goppa codes in dyadic form.*

Proof. Each dyadic signature produced by Algorithm 1 is entirely determined by the values h_0 and h_{2^s} for $s = 0, \ldots, \lceil \lg n \rceil - 1$ chosen at steps 2 and 7 (ω only produces equivalent codes). Along the loop at line 5, exactly $2i = 2^{s+1}$ elements are erased from U, corresponding to the choices of $h_{2^s} \ldots h_{2^{s+1}-1}$. At the end of that loop, $2 + 2\sum_{\ell=0}^{s} 2^\ell = 2^{s+2}$ elements have been erased in total. Hence at the beginning of each step of the loop only 2^{s+1} elements had been erased from U, i.e. there are $q - 2^{s+1}$ elements in U to choose h_{2^s} from, and $q - 1$ possibilities for h_0. Therefore this construction potentially yields up to $(q-1)\prod_{s=0}^{\lceil \lg n \rceil - 1} (q - 2^{s+1}) = \prod_{i=0}^{\lceil \lg n \rceil} (q - 2^i)$ possible codes. □

Theorem 3 actually establishes the number of distinct essences of dyadic signatures corresponding to Cauchy matrices. The roots of the Goppa polynomial are completely specified by the first $\lceil \lg t \rceil$ elements of the essence η together with $\eta_{\lceil \lg n \rceil}$, namely, $z_i = \eta_{\lceil \lg n \rceil} + \sum_{k=0}^{\lceil \lg t \rceil - 1} i_k \eta_k$, disregarding the ω term which is implicit in the choice of $\eta_{\lceil \lg n \rceil}$. We see that any permutation of the essence elements $\eta_0, \ldots, \eta_{\lceil \lg t \rceil - 1}$ only changes the order of those roots. Since the Goppa polynomial itself is defined by its roots regardless of their order, the total number of possible Goppa polynomials is therefore $(\lceil \lg t \rceil !)^{-1} \prod_{i=0}^{\lceil \lg t \rceil} (q - 2^i) \approx (q-t)\binom{q}{\lceil \lg t \rceil}$.

For $n \approx q/2$ the number of dyadic codes can be approximated by $q^m Q = 2^{m^2} Q$ where $Q = \prod_{i=1}^{\infty} (1 - 1/2^i) \approx 0.2887881$. We will also see that the number of quasi-dyadic codes, which we describe next and propose for cryptographic applications, is larger than this. Before we proceed, however, it is interesting to notice that one of the reasons the attack proposed in [22] succeeds against certain quasi-cyclic codes, besides the constrained structure of the applied permutation, is that those schemes start from a known BCH or Reed-Solomon code which is unique up to the choice of a primitive element from the underlying finite field. Thus, in those proposals an initial code over $\mathbb{F}_{2^m}$ is at best chosen from a set of $O(2^m)$ codes. In comparison, we start from a secret code sampled from a much larger family of $O(2^{m^2})$ codes. For instance, while those proposals have only 2^{15} starting points over $\mathbb{F}_{2^{16}}$, our scheme can sample a family with more than 2^{254} codes over the same field. The main protection of the hidden trapdoor is, of course, the block puncturing process and the more complex blockwise permutation of the initial secret code, as detailed next.

3.2 Constructing quasi-dyadic, permuted subfield subcodes

To complete the construction it is necessary to choose a compact generator matrix for the subfield subcode. Although the parity check matrix H built by Algorithm 1 is dyadic over $\mathbb{F}_q$, the usual trace construction leads to a generator of the dual code that most probably violates the dyadic symmetry. However, by representing each field element to a basis of $\mathbb{F}_q$ over the base subfield $\mathbb{F}_2$, one can view H as a superposition of $m = [\mathbb{F}_q : \mathbb{F}_2]$ distinct binary dyadic matrices, and each of them can be stored in a separate dyadic signature.

A cryptosystem cannot be securely defined on a Goppa code specified directly by a parity-check matrix in Cauchy form, since this would immediately reveal the Goppa polynomial $g(x)$: it suffices to solve the overdefined linear system $z_i - L_j = 1/H_{ij}$ consisting of tn equations in $t + n$ unknowns.

Algorithm 1 generates fully dyadic codes. We now show how to integrate the techniques of Berger et al. with Algorithm 1 so as to build quasi-dyadic subfield subcodes whose parity-check matrix is a non-dyadic matrix composed of blocks of dyadic submatrices. The principle to follow here is to *select and permute* the columns of the original parity-check matrix so as to preserve quasi-dyadicity in the target subfield subcode and the distribution of introduced errors in cryptosystems. A similar process yields a generator matrix in convenient quasi-dyadic, systematic form.

For the desired security level (see the discussion in Section 5.1), choose $q = 2^m$ for some m, a code length n and a design number of correctable errors t such that $n = \ell t$ for some $\ell > m$. For simplicity we assume that t is a power of 2, but the following construction method can be modified to work with other values.

Run Algorithm 1 to produce a code over $\mathbb{F}_q$ whose length $N \gg n$ is a large multiple of t not exceeding the largest possible length $q/2$, so that the constructed $t \times N$ parity-check matrix $\hat{H}$ can be viewed as a sequence of N/t dyadic blocks $[B_0 \mid \cdots \mid B_{N/t-1}]$ of size $t \times t$ each. Select uniformly at random ℓ distinct blocks $B_{i_0}, \ldots, B_{i_{\ell-1}}$ in any order from $\hat{H}$, together with ℓ dyadic permutations $\Pi^{j_0}, \ldots, \Pi^{j_{\ell-1}}$ of size $t \times t$. Let $\hat{H}' = [B_{i_0}\Pi^{j_0} \mid \cdots \mid B_{i_{\ell-1}}\Pi^{j_{\ell-1}}] \in (\mathbb{F}_q^{t\times t})^\ell$. Compute the co-trace matrix $H' = T'_m(\hat{H}') \in (\mathbb{F}_2^{t\times t})^{m\times\ell}$ and finally the systematic form H of H'.

The resulting parity-check matrix defines a binary code of length n and dimension $k = n - mt$, and since all block operations performed during the Gaussian elimination are carried out in the ring $\Delta(\mathbb{F}_2^t)$, the result still consists of dyadic submatrices which can be represented by a signature of length t. Hence the whole matrix can be stored in space a factor t smaller than a general matrix. However, the dyadic submatrices that appear in this process are not necessarily nonsingular, as they are not associated to a Cauchy matrix anymore; should all the submatrices on a column be found to be singular (above or below the diagonal, according to the direction of this process) so that no pivot is possible, the whole block containing that column may be replaced by another block $B_{j'}$ chosen at random from $\hat{H}$ as above.

The trapdoor information consisting of the essence η of h, the sequence $(i_0, \ldots, i_{\ell-1})$ of blocks, and the sequence $(j_0, \ldots, j_{\ell-1})$ of dyadic permutation identifiers, relates the public code defined by H with the private code defined by $\hat{H}$. The space occupied by the trapdoor information is thus $m^2 + \ell \lg N$ bits. If one starts with the largest possible $N = 2^{m-1}$, this simplifies to the maximal size of $m^2 + \ell(m-1)$ bits.

The total space occupied by the essential part of the resulting binary generator (or parity-check) matrix is $mt \times (n - mt)/t = mk$ bits, a factor t better than generic Goppa codes, which occupy $k(n-k) = mkt$ bits. Had t not been chosen to be a power of 2, say, $t = 2^u v$ where $v > 1$ is odd, the cost of multiplying $t \times t$

matrices would be in general $O(2^u uv^3)$ rather than simply $O(2^u u)$, and the final parity-check matrix would be compressed by only a factor 2^u.

For each code produced by Algorithm 1, the number of codes generated by this construction is $\binom{N/t}{\ell} \times \ell! \times t^\ell$, hence $\binom{N/t}{\ell} \times \ell! \times t^\ell \times \prod_{i=0}^{\lceil \lg N \rceil} (q - 2^i)$ codes are possible in principle.

3.3 A toy example

Let $\mathbb{F}_{2^5} = \mathbb{F}_2[u]/(u^5 + u^2 + 1)$. The dyadic signature

$$h = (u^{20}, u^3, u^6, u^{28}, u^9, u^{29}, u^4, u^{22}, u^{12}, u^5, u^{10}, u^2, u^{24}, u^{26}, u^{25}, u^{15})$$

and the offset $\omega = u^{21}$ define a 2-error correcting binary Goppa code of length $N = 16$ with $g(x) = (x - u^{12})(x - u^{15})$ and support $L = (u^{21}, u^{29}, u^{19}, u^{26}, u^6, u^{16}, u^7, u^5, u^{25}, u^3, u^{11}, u^{28}, u^{27}, u^9, u^{22}, u^2)$. The associated parity-check matrix built according to Theorem 1 is

$$\hat{H} = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc} u^{20} & u^3 & u^6 & u^{28} & u^9 & u^{29} & u^4 & u^{22} & u^{12} & u^5 & u^{10} & u^2 & u^{24} & u^{26} & u^{25} & u^{15} \\ u^3 & u^{20} & u^{28} & u^6 & u^{29} & u^9 & u^{22} & u^4 & u^5 & u^{12} & u^2 & u^{10} & u^{26} & u^{24} & u^{15} & u^{25} \end{array}\right],$$

with eight 2×2 blocks $B_0, \ldots, B_7$ as indicated. From this we extract the shortened, rearranged and permuted sequence $\hat{H}' = [B_7\Pi^0 \mid B_5\Pi^1 \mid B_1\Pi^0 \mid B_2\Pi^1 \mid B_3\Pi^0 \mid B_6\Pi^1 \mid B_4\Pi^0]$ (because in this example the subfield is the base field itself, all scale factors have to be 1), i.e.:

$$\hat{H} = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc} u^{25} & u^{15} & u^2 & u^{10} & u^6 & u^{28} & u^{29} & u^9 & u^4 & u^{22} & u^{26} & u^{24} & u^{12} & u^5 \\ u^{15} & u^{25} & u^{10} & u^2 & u^{28} & u^6 & u^9 & u^{29} & u^{22} & u^4 & u^{24} & u^{26} & u^5 & u^{12} \end{array}\right],$$

whose co-trace matrix over $\mathbb{F}_2$ has the systematic form:

$$H = \left[\begin{array}{cccc|cccccccccc} 0&1&0&1&1&0&0&0&0&0&0&0&0&0 \\ 1&0&1&0&0&1&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&1&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&1&1&0&0&0&0&1&0&0&0&0&0 \\ 0&0&1&1&0&0&0&0&0&1&0&0&0&0 \\ 0&1&1&0&0&0&0&0&0&0&1&0&0&0 \\ 1&0&0&1&0&0&0&0&0&0&0&1&0&0 \\ 1&1&0&0&0&0&0&0&0&0&0&0&1&0 \\ 1&1&0&0&0&0&0&0&0&0&0&0&0&1 \end{array}\right] = [M^\mathsf{T} \mid I_{n-k}],$$

from which one readily obtains the $k \times n = 4 \times 14$ generator matrix in systematic form:

$$G = \left[\begin{array}{cccc|cccccccccc} 1&0&0&0&0&1&0&1&0&0&0&1&1&1 \\ 0&1&0&0&1&0&1&0&0&0&1&0&1&1 \\ 0&0&1&0&0&1&0&0&1&1&1&0&0&0 \\ 0&0&0&1&1&0&0&0&1&1&0&1&0&0 \end{array}\right] = [I_k \mid M],$$

where both G and H share the essential part M:

$$M = \begin{bmatrix} \mathbf{0\,1\,0\,1\,0\,0\,0\,1\,1\,1} \\ 1\,0\,1\,0\,0\,0\,1\,0\,1\,1 \\ \mathbf{0\,1\,0\,0\,1\,1\,1\,0\,0\,0} \\ 1\,0\,0\,0\,1\,1\,0\,1\,0\,0 \end{bmatrix},$$

which is entirely specified by the elements in boldface and can thus be stored in 20 bits instead of, respectively, $4 \cdot 14 = 56$ and $10 \cdot 14 = 140$ bits.

4 Assessing the hardness of decoding quasi-dyadic codes

The original McEliece (or, for that matter, the original Niederreiter) schemes are perhaps better described as a candidate *trapdoor one-way functions* rather than full-fledged public-key encryption schemes. Such functions are used in cryptography in many different settings, each with different security requirements, and we do not consider such applications in this paper. Instead we focus purely on the question of inverting the trapdoor function, in other words, decoding.

As we pointed out in Section 1, the well-studied class of Goppa codes remains one of the best choices to instantiate McEliece-like schemes. Although our proposal is ultimately based on Goppa codes, one may wonder whether or not the highly composite nature of the Goppa generator polynomial $g(x)$, or the peculiar structure of the quasi-dyadic parity-check and generator matrices, leak any information that might facilitate decoding without knowledge of the trapdoor.

Yet, any alternant code can be written in Goppa-like fashion by using the diagonal component of its default parity-check matrix (see Definition 6) to interpolate a generating polynomial (not necessarily of degree t) that is composite with high probability. We are not aware of any way this fact could be used to facilitate decoding without full knowledge of the code structure, and clearly any result in this direction would affect most of the alternant codes proposed for cryptographic purposes to date.

Otmani et al.'s attack against quasi-cyclic codes [22] could be modified to work against Goppa codes in dyadic form. For this reason we adopt the same countermeasures proposed by Berger et al. to thwart it for cyclic codes, namely, working with a block-shortened subcode of a very large code as described in Section 3.2. This idea also build upon the work of Wieschebrink [30] who proved that deciding whether a code is equivalent to a shortened code is NP-complete. In our case, the result is to hide the Cauchy structure of the private code in a general dyadic structure, rather than disguising a quasi-cyclic code as another one with the same symmetry.

We now give a reduction of the problem of decoding the particular class of quasi-dyadic codes to the well-studied syndrome decoding problem, classical in coding theory and known to be NP-complete [4].

Definition 8 (Syndrome decoding). *Let $\mathbb{F}_q$ be a finite field, and let (H, w, s) be a triple consisting of a matrix $H \in \mathbb{F}_q^{r \times n}$, an integer $w < n$, and a vector*

$s \in \mathbb{F}_q^r$. *Does there exist a vector* $e \in \mathbb{F}_q^n$ *of Hamming weight* $\mathsf{wt}(e) \leqslant w$ *such that* $He^\mathsf{T} = s^\mathsf{T}$*?*

The corresponding problem for quasi-dyadic matrices reads:

Definition 9 (Quasi-dyadic syndrome decoding). *Let* $\mathbb{F}_q$ *be a finite field, and let* (H, w, s) *be a triple consisting of a quasi-dyadic matrix* $H \in \Delta(\mathbb{F}_q^\ell)^{r \times n}$*, an integer* $w < \ell n$*, and a vector* $s \in \mathbb{F}_q^{\ell r}$*. Does there exist a vector* $e \in \mathbb{F}_q^{\ell n}$ *of Hamming weight* $\mathsf{wt}(e) \leqslant w$ *such that* $He^\mathsf{T} = s^\mathsf{T}$*?*

Theorem 4. *The quasi-dyadic syndrome decoding problem (QD-SDP) is polynomially equivalent to the syndrome decoding problem (SDP). In other words, decoding quasi-dyadic codes is as hard in the worst case as decoding general codes.*

Proof. The QD-SDP, being an instance of the SDP restricted to a particular class of codes, is clearly a decision problem in NP.

Consider now a generic instance $(H', w', s') \in \mathbb{F}_q^{r \times n} \times \mathbb{Z} \times \mathbb{F}_q^r$ of the SDP. Assume one is given an oracle that solves the QD-SDP over $\Delta(\mathbb{F}_q^\ell)$ for some given $\ell > 0$. Let $v_\ell \in \mathbb{F}_q^\ell$ be the all-one vector, i.e. $(v_\ell)_j = 1$ for all j. Define the quasi-dyadic matrix $H = H' \otimes I_\ell \in \Delta(\mathbb{F}_q^\ell)^{r \times n}$ with blocks $H_{ij} = H'_{ij} I_\ell$, the vector $s = s' \otimes v_\ell \in (\mathbb{F}_q^\ell)^r$ with blocks $s_i = s'_i v_\ell$, and $w = \ell w'$. It is evident that the instance $(H, w, s) \in \Delta(\mathbb{F}_q^\ell)^{r \times n} \times \mathbb{Z} \times (\mathbb{F}_q^\ell)^r$ of the QD-SDP can be constructed in polynomial time.

Assume now that there exists $e \in \mathbb{F}_q^{\ell n}$ of Hamming weight $\mathsf{wt}(e) \leqslant w$ such that $He^\mathsf{T} = s^\mathsf{T}$. For all $0 \leqslant i < \ell$, let $e'_i \in \mathbb{F}_q^n$ be the vector with elements $(e'_i)_j = e_{i+j\ell}$, $0 \leqslant j < n$, so that the e'_i are interleaved to compose e. Obviously at least one of the e'_i has Hamming weight not exceeding $w/\ell = w'$, and by the construction of H any of them satisfies $He'^\mathsf{T}_i = s'^\mathsf{T}$, constituting a solution to the given instance of the SDP. This effectively reduces the SDP to the QD-SDP for any given ℓ in polynomial time. Thus, the QD-SDP itself is NP-complete. □

Although this theorem does not say anything about hardness in the average case, it nevertheless strengthens our claim that the family of codes we propose is in principle no less suitable for cryptographic applications than a generic code, in the sense that, should the QD-SDP problem turn out to be feasible in the worst case, then *all* coding-based cryptosystems would definitely be ruled out, regardless of which code is used to instantiate them. Incidentally, the expected running time of all known algorithms for the SDP (and the QD-SDP) is exponential, so there is empirical evidence that the average case is also very hard. We stress, however, that particular cryptosystems based on quasi-dyadic codes will usually depend on more specific security assumptions, whose assessment transcends the scope of this paper.

4.1 Reduction to multivariate quadratic equations

Recently, Faugère *et al.* [9] proposed to reduce the decoding problem for quasi-dyadic codes (and others) to the problem of solving systems of multivariate

quadratic equations (MQE). The overall idea is to seek an alternant decoder for the public code directly, i.e. to write the public parity-check matrix as $H = VD$ for an unknown Vandermonde matrix V and an unknown diagonal matrix D defined over the public field $\mathbb{F}_{2^d}$, where $d \mid m$. A related technique is adopted in [29]. The unknown components of V and D in the defining equation $H = VD$ give rise to an instance of the MQE problem. By making careful use of the structure of H, many component equations become linear and the remaining quadratic part involves a smaller number of variables, greatly reducing the complexity of the resulting system. This way the authors are able to break all parameters proposed in [3] and [19] over extension fields (but not the binary parameters).

Apart from the fact that the attack complexity increases steeply as the codes are defined over ever smaller extension fields, we argue that this strategy cannot yield an attack against binary QD codes, even if it succeeds against e.g. quasi-cyclic codes. The reason is that the attack principle is to construct an *alternant* trapdoor directly from the public code defined by H, which is *not* a Goppa code except with overwhelmingly low probability. This trapdoor can be used to correct about $t/2$ errors at most, where t is the design number of errors. For all alternant codes except binary Goppa codes this is exactly the same as the number of errors that can be introduced and then successfully corrected using the *private* trapdoor, which explains why the attack is successful as long as the associated MQE instance can be solved in practice. However, for the specific case of binary Goppa codes, including binary QD codes, this attack can only correct *half* as many errors as can be introduced and then corrected using the private Goppa trapdoor. Since applying a $t/2$ decoder to a word with t errors produces garbage rather than a word with only the remaining $t/2$ errors, the attacker would have to guess $t/2$ errors before using the obtained alternant trapdoor to correct the remaining ones. This means repeating the attempted decoding $\binom{n}{t/2}/\binom{t}{t/2}$ times, which is infeasible for properly chosen practical parameters.

We conclude that existing attacks based on solving instances of the MQE problem fail against properly chosen, yet still practical, binary QD codes.

5 Efficiency considerations

Due to their simple structure the matrices in our proposal can be held on a simple vector not only for long-term storage or transmission, but for processing as well.

The operation of multiplying a vector by a (quasi-)dyadic matrix is at the core of McEliece encryption. The fast Walsh-Hadamard transform (FWHT) [14] approach for dyadic convolution via lifting[2] to characteristic 0 leads to the asymptotic complexity $O(n \lg n)$ for this operation and hence also for encoding. Sarwate's decoding method [24] sets the asymptotic cost of that operation at roughly $O(n \lg n)$ as well for the typical cryptographic setting $t = O(n/\lg n)$.

[2] We are grateful to Dan Bernstein for suggesting the lifting technique to emulate the FWHT in characteristic 2.

Inversion, on the other hand, can be carried out in $O(n)$ steps: one can show by induction that a binary dyadic matrix $\Delta(h)$ of dimension n satisfies $\Delta^2 = (\sum_i h_i)^2 I$, and hence its inverse, when it exists, is $\Delta^{-1} = (\sum_i h_i)^{-2}\Delta$, which can be computed in $O(n)$ steps since it is entirely determined by its first row.

Converting a quasi-dyadic matrix to systematic (echelon) form involves a Gaussian elimination incurring about $m^2\ell$ products of dyadic $t \times t$ submatrices, implying a complexity $O(m^2\ell t \lg t) = O(m^2 n \lg n)$, which simplifies to $O(n \lg^3 n)$ in the typical cryptographic setting $m = O(\lg n)$.

Table 1 summarizes the asymptotic complexities of code generation (mainly due to systematic formatting), encoding and decoding, which coincide with the complexities of key generation, encryption and decryption of typical cryptosystems based on codes.

Table 1. Operation complexity relative to the code length n.

operation	generic	ours
Code generation	$O(n^3)$	$O(n \lg^3 n)$
Encode/Decode	$O(n^2)$	$O(n \lg n)$

5.1 Suggested parameters

Table 2 contains a variety of balanced parameters for practical security levels. Although we do not recommend these for actual deployment before further analysis is carried out, these parameters were chosen to stress the possibilities of our proposal while giving a realistic impression of what one might indeed adopt in practice. The number of errors is always a power of 2 to enable maximum size reduction, and the original code from which the binary Goppa code is extracted is always defined over $\mathbb{F}_{2^{16}}$. The actual security levels[3] computed according to the attack strategy in [7] are, respectively, 84.3, 112.3, 136.5, 201.6, and 265.1, while the number of possible codes ranges between $2^{669.6}$ and $2^{792.4}$. The target security level, roughly corresponding to the estimated logarithmic cost of the best known attack according to the guidelines in [7], is shown on the 'level' column. The 'size' column contains the amount of bits effectively needed to store a quasi-dyadic generator or parity-check matrix in systematic form. The size of a corresponding systematic matrix for a generic Goppa code at roughly the same security level as suggested in [7] is given on column 'generic'. In both cases we take into account only the redundancy part of the key in systematic form (that it is safe to do so is proven in [8]). The 'shrink' column contains the size ratio between such a generic matrix and a matching quasi-dyadic matrix.

For the parameters on Table 2, we observed the timings on Table 3 (measured in ms) for generic Goppa codes and quasi-dyadic (QD) codes, and also for RSA

[3] We are grateful to Christiane Peters for kindly providing these estimates.

Table 2. Sample parameters for a $[n, k, 2t + 1]$ binary Goppa code.

level	n	k	t	size	generic	shrink
80	2304	1280	64	20480	460647	22.5
112	3584	1536	128	24576	1047600	42.6
128	4096	2048	128	32768	1537536	46.9
192	6912	2816	256	45056	4185415	92.9
256	8192	4096	256	65536	7667855	117.0

to assess the efficiency relative to a very common pre-quantum cryptosystem. We made no serious attempt at optimizing the implementation, which was done in C++ and tested on an AMD Turion 64X2 2.4 GHz. Benchmarks for RSA-15360 were omitted due to the enormous time needed to generate suitable parameters.

Table 3. Benchmarks for typical parameters.

level	generation			encoding			decoding		
	RSA	generic	QD	RSA	generic	QD	RSA	generic	QD
80	563	375	17.2	0.431	0.736	0.817	15.61	1.016	3.685
112	1971	1320	18.7	1.548	1.696	1.233	110.34	2.123	4.463
128	4998	2196	20.5	3.467	2.433	1.575	349.91	3.312	5.261
192	628183	13482	47.6	22.320	6.872	4.695	5094.10	8.822	17.783
256	–	27161	54.8	–	12.176	6.353	–	15.156	21.182

6 Conclusion and further research

We have described how to generate Goppa codes in quasi-dyadic form suitable for cryptographic applications. Key sizes for a typical McEliece-like cryptosystem are roughly a factor $t = \tilde{O}(n)$ smaller than generic Goppa codes, and keys can be kept in this compact size not only for storing and transmission but for processing as well. In the binary case these codes can correct the full design number of errors. This brings the size of cryptographic keys to within a factor 4 or less of equivalent RSA keys, comparable to NTRU keys. Our work provides an alternative to conventional cyclic and quasi-cyclic codes, and benefits from the same trapdoor-hiding techniques proposed by Wieschebrink in general [30], and by Berger et al. for that family of codes [3].

The complexity of all operations in McEliece and related cryptosystems is reduced to $\tilde{O}(n)$. Other cryptosystems can also benefit from dyadic codes, e.g. entity identification and certain digital signatures for which double circulant codes have been proposed [11] could use dyadic codes instead, even random ones without a Goppa trapdoor. One further line of research is whether one can securely combine the techniques in [2] with ours to define quasi-dyadic,

low-density parity-check (QD-LDPC) codes that are suitable for cryptographic purposes and potentially even shorter than plain quasi-dyadic codes.

Interestingly, it is equally possible to define *lattice*-based cryptosystems with short keys using dyadic lattices entirely analogous to ideal (cyclic) lattices as proposed by Micciancio [18], and achieving comparable size reduction. We leave this line of inquiry for future research since it falls outside the scope of this paper.

Acknowledgments

We are most grateful and deeply indebted to Marco Baldi, Dan Bernstein, Pierre-Louis Cayrel, Philippe Gaborit, Steven Galbraith, Robert Niebuhr, Christiane Peters, Nicolas Sendrier, and the anonymous reviewers for their valuable comments and feedback during the preparation of this work.

References

1. M. Baldi and F. Chiaraluce. Cryptanalysis of a new instance of McEliece cryptosystem based on QC-LDPC code. In *IEEE International Symposium on Information Theory – ISIT'2007*, pages 2591–2595, Nice, France, 2007. IEEE.
2. M. Baldi, F. Chiaraluce, and M. Bodrato. A new analysis of the mceliece cryptosystem based on qc-ldpc codes. In *Security and Cryptography for Networks – SCN'2008*, volume 5229 of *Lecture Notes in Computer Science*, pages 246–262. Springer, 2008.
3. T. P. Berger, P.-L. Cayrel, P. Gaborit, and A. Otmani. Reducing key length of the McEliece cryptosystem. In *Progress in Cryptology – Africacrypt'2009*, Lecture Notes in Computer Science. Springer, 2009. To appear. Preliminary (2008) version at `http://www.unilim.fr/pages_perso/philippe.gaborit/reducing.pdf`.
4. E. Berlekamp, R. McEliece, and H. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.
5. D. J. Bernstein. List decoding for binary Goppa codes. Preprint, 2008. `http://cr.yp.to/papers.html#goppalist`.
6. D. J. Bernstein, J. Buchmann, and E. Dahmen. *Post-Quantum Cryptography*. Springer, 2008.
7. D. J. Bernstein, T. Lange, and C. Peters. Attacking and defending the McEliece cryptosystem. In *Post-Quantum Cryptography Workshop – PQCrypto'2008*, volume 5299 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2008. `http://www.springerlink.com/content/68v69185x478p53g`.
8. D. Engelbert, R. Overbeck, and A. Schmidt. A summary of McEliece-type cryptosystems and their security. *Journal of Mathematical Cryptology*, 1:151–199, 2007.
9. J.-C. Faugère, A. Otmani, L. Perret, and J.-P. Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. In *Advances in Cryptology – Eurocrypt'2010*, LNCS. Springer, 2010. To appear.
10. P. Gaborit. Shorter keys for code based cryptography. In *International Workshop on Coding and Cryptography – WCC'2005*, pages 81–91, Bergen, Norway, 2005. ACM Press.

11. P. Gaborit and M. Girault. Lightweight code-based authentication and signature. In *IEEE International Symposium on Information Theory – ISIT'2007*, pages 191–195, Nice, France, 2007. IEEE.
12. J. K. Gibson. Severely denting the Gabidulin version of the McEliece public key cryptosystem. *Designs, Codes and Cryptography*, 6(1):37–45, 1995.
13. J. K. Gibson. The security of the Gabidulin public key cryptosystem. In *Advances in Cryptology – Eurocrypt'1996*, volume 1070 of *Lecture Notes in Computer Science*, pages 212–223, Zaragoza, Spain, 1996. Springer.
14. M. N. Gulamhusein. Simple matrix-theory proof of the discrete dyadic convolution theorem. *Electronics Letters*, 9(10):238–239, 1973.
15. P. Loidreau and N. Sendrier. Some weak keys in McEliece public-key cryptosystem. In *IEEE International Symposium on Information Theory – ISIT'1998*, page 382, Boston, USA, 1998. IEEE.
16. F. J. MacWilliams and N. J. A. Sloane. *The theory of error-correcting codes*, volume 16. North-Holland Mathematical Library, 1977.
17. R. McEliece. A public-key cryptosystem based on algebraic coding theory. The Deep Space Network Progress Report, DSN PR 42–44, 1978. `http://ipnpr.jpl.nasa.gov/progressreport2/42-44/44N.PDF`.
18. D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 16(4):365–411, 2007.
19. R. Misoczki and P. S. L. M. Barreto. Compact McEliece keys from Goppa codes. Submitted preprint, 2009. `http://eprint.iacr.org/2009/187`.
20. C. Monico, J. Rosenthal, and A. Shokrollahi. Using low density parity check codes in the McEliece cryptosystem. In *IEEE International Symposium on Information Theory – ISIT'2000*, page 215, Sorrento, Italy, 2000. IEEE.
21. H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.
22. A. Otmani, J.-P. Tillich, and L. Dallot. Cryptanalysis of two McEliece cryptosystems based on quasi-cyclic codes. Preprint, 2008. `http://arxiv.org/abs/0804.0409v2`.
23. N. J. Patterson. The algebraic decoding of Goppa codes. *IEEE Transactions on Information Theory*, 21(2):203–207, 1975.
24. D. V. Sarwate. On the complexity of decoding Goppa codes. *IEEE Transactions on Information Theory*, 23(4):515–516, 1977.
25. S. Schechter. On the inversion of certain matrices. *Mathematical Tables and Other Aids to Computation*, 13(66):73–77, 1959. `http://www.jstor.org/stable/2001955`.
26. N. Sendrier. Finding the permutation between equivalent linear codes: the support splitting algorithm. *IEEE Transactions on Information Theory*, 46(4):1193–1203, 2000.
27. V. Sidelnikov and S. Shestakov. On cryptosystems based on generalized Reed-Solomon codes. *Discrete Mathematics*, 4(3):57–63, 1992.
28. K. K. Tzeng and K. Zimmermann. On extending Goppa codes to cyclic codes. *IEEE Transactions on Information Theory*, 21:721–716, 1975.
29. V. G. Umana and G. Leander. Practical key recovery attacks on two McEliece variants. Cryptology ePrint Archive, Report 2009/509, 2009. `http://eprint.iacr.org/`.
30. C. Wieschebrink. Two NP-complete problems in coding theory with an application in code based cryptography. In *IEEE International Symposium on Information Theory – ISIT'2006*, pages 1733–1737, Seattle, USA, 2006. IEEE.