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, keeping the capability of correcting the full designed number of errors while also improving the efficiency of cryptographic operations to subquadratic time.

Keywords: post-quantum cryptography, syndrome decoding, efficient parameters and algorithms.

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 [16] and Niederreiter [19]. Such systems usually have even a speed advantage over conventional schemes; for instance, McEliece or Niederreiter encryption over a code of length n has time complexity $O(n^2)$, while (private exponent) RSA and DSA 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 toward this goal were taken by Monico et al. using low density parity-check codes [18], by Gaborit using quasi-cyclic codes [8], and by Baldi and Chiaraluce using a combination of both [1]. However, these proposals were all shown to contain weaknesses [21].

Recently Berger et al. [2] showed how to circumvent the drawbacks of Gaborit's original scheme and remove the weaknesses pointed out in [21] by means of two techniques:

1. Extracting block-shortened public codes from very large private codes, adapting an idea proposed by Wieschebrink [28];

⋆ 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.

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. Besides, the class of quasi-cyclic codes proposed in [2] has the drawback, common to most other classes of alternant codes, that no decoding algorithm capable of correcting more than $t/2$ errors is currently known (i.e. in practice only half the designed number of errors can be corrected). The ability to correct as many errors as possible with the decoding trapdoor is important in schemes like CFS signatures, since it is closely related to the probability of finding a decodable syndrome (and thus producing a signature) by random sampling [7]. It also allows for smaller keys than would be possible using a different kind of code, including the quasi-cyclic codes as proposed in [2].

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 [10, 11, 21, 26], Goppa codes have withstood cryptanalysis quite well, and despite considerable progress in the area [14, 25] (see also [5] 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 binary Goppa codes, while retaining the capability 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 [2]. Furthermore, the complexity of all typical cryptographic operations become no more than $O(n^{\lg 3})$; specifically, code generation (excluding conversion to systematic form) has complexity $O(n \lg n)$, while systematic formatting, encryption and decryption take no more than $O(n^{\lg 3})$ time.

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 [12].

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 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 [24]. 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 *$GRS_d(L, D)$ is an $[n, k, d]$ linear error-correcting code defined by the parity-check matrix*

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

The classical Reed-Solomon code itself is $GRS_d(\boldsymbol{\alpha}, \boldsymbol{\alpha})$ for $\boldsymbol{\alpha} = (1, \alpha, \alpha^2, \ldots, \alpha^{n-1})$ and some primitive element $\alpha \in \mathbb{F}_q$.

Definition 7. *An* alternant code *of length n and dimension k over $\mathbb{F}_q$ for $q = p^m$ is a subfield subcode of a Generalized Reed-Solomon code over $\mathbb{F}_p$.*

Subfield subcodes are easily derived from a given code by means of the trace construction:

Definition 8. *The* trace construction *produces an $\mathbb{F}_p$-subfield subcode of a given code over $\mathbb{F}_{p^m} = \mathbb{F}_p[x]/r(x)$ (for some irreducible polynomial $r(x) \in \mathbb{F}_p[x]$ of degree m) by writing the $\mathbb{F}_p$ coefficients of each $\mathbb{F}_{p^m}$ component of a parity-check matrix $H_{t\times n}$ of the original code onto m successive rows of a parity-check matrix $H'_{mt\times n}$ of the subcode. The related* co-trace *parity-check matrix $H''_{mt\times n}$, equivalent to 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 H''.*

Thus, given $\mathbb{F}_{p^m}$ elements $u_i(x) = u_{i,0} + \cdots + u_{i,m-1}x^{m-1}$, the trace construction maps a column $(u_0, \ldots, u_{t-1})^\mathsf{T}$ from H to the column $(u_{0,0}, \ldots, u_{0,m-1}; \ldots; u_{t-1,0}, \ldots, u_{t-1,m-1})^\mathsf{T}$ on the trace matrix H', and to the column $(u_{0,0}, \ldots, u_{t-1,0}; \ldots; u_{0,m-1}, \ldots, u_{t-1,m-1})^\mathsf{T}$ on the co-trace matrix H''.

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

Definition 9. *Given $q = p^m$, a sequence $L = (L_0, \ldots, L_{n-1}) \in \mathbb{F}_q^n$ of distinct elements and a polynomial $g(x) \in \mathbb{F}_q[x]$ such that $g(L_i) \neq 0$ for $0 \leqslant i < n$, the* Goppa code *is the alternant code over $\mathbb{F}_p$ defined by the parity-check matrix $H = GRS_d(L, D)$ where $D = (g(L_0)^{-1}, \ldots, g(L_{n-1})^{-1})$.*

A binary irreducible Goppa code with design distance $2t + 1$ can correct up to t bits using Patterson's algorithm [22], or slightly more using Bernstein's list decoding method [4], and t errors can still be corrected by suitable decoding algorithms if the binary generator $g(x)$ is not irreducible[1]. In all other cases no more than $t/2$ errors can in general be corrected.

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:

Theorem 1 ([27]). *The Goppa code defined by a polynomial $g(x) = (x - z_0) \ldots (x - z_{t-1})$ 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 [15, Ch. 12, §3, Pr. 5]) is completely general when one considers the factorization of the Goppa polynomial over its split 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 zeroes of that polynomial are in the field $\mathbb{F}_q$ itself.

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. Actually this system has rank $t + n - 1$, but the solutions this incurs are all equivalent in the sense that, for any fixed $\omega \in \mathbb{F}_q$, the polynomial defined by the roots $z'_i = z_i + \omega$ on the support $L'_j = L_j + \omega$ leads to the same parity-check matrix $H_{ij} = 1/(z'_i - L'_j)$ and hence to the same syndromes, decodable using any of those codes.

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. We do so by constructing a sequence $h = (h_0, \ldots, h_{n-1}) \in \mathbb{F}_q^n$, with $q = 2^m$ for some m, of distinct nonzero values such that

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

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

choosing an element $\omega \in \mathbb{F}_q$, and then defining the Goppa code $\Gamma(L, g)$ by assembling the polynomial g from its zeros $z_i = 1/h_i + \omega$ and the code support $L_j = 1/h_j + 1/h_0 + \omega$. Clearly the parity-check matrix in Cauchy form $H_{ij} = 1/(z_i - L_j)$ is actually a dyadic matrix $H_{ij} = h_{i \oplus j}$.

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

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

for $0 < j < i$ (so that $i + j = i \oplus j$), as long as this value is well-defined. Algorithm 1, whose running time is $O(n \lg n)$ steps, captures this idea. The notation $u \xleftarrow{\$} U$ means that variable u is uniformly sampled at random from set U.

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

$U \leftarrow \mathbb{F}_q \setminus \{0\}$
▷ 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.
$h_0 \xleftarrow{\$} U$
$U \leftarrow U \setminus \{h_0\}$
for $s \leftarrow 0$ **to** $\lceil \lg n \rceil - 1$ **do**
    $i \leftarrow 2^s$
    $h_i \xleftarrow{\$} U$
    $U \leftarrow U \setminus \{h_i, 1/(1/h_i + 1/h_0)\}$
    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

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 subfield $\mathbb{F}_p$, one can view H as

a superposition of $m = [\mathbb{F}_q : \mathbb{F}_p]$ distinct dyadic matrices over $\mathbb{F}_p$, and each of them can be stored in a separate dyadic signature.

Theorem 2. *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 6 (ω only produces equivalent codes). Along the loop at line 4, 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. □

We will see, however, that the number of permuted subfield subcodes that we describe next and propose for cryptographic applications is larger than this.

3.2 Constructing quasi-dyadic, permuted subfield subcodes

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, permute, and scale* 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 $r = 2^m$ for some m, $q = r^d$ for some d, a code length n and a design number of correctable errors t such that $n = \ell t$ for some $\ell > d$. 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 parity-check matrix $H_{t\times N}$ 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 ℓ distinct blocks $B_{i_0}, \ldots, B_{i_{\ell-1}}$ in any order from H together with ℓ random dyadic permutations $(\Pi^{j_0}, \ldots, \Pi^{j_{\ell-1}})$ of size $t \times t$ and ℓ nonzero scale factors $(\sigma_0, \ldots, \sigma_{\ell-1}) \in \mathbb{F}_r^\ell$. Let $H''_{dt\times \ell t}$ be the co-trace matrix over $\mathbb{F}_r$ constructed from $[B_{i_0}\sigma_0\Pi^{j_0} \mid \cdots \mid B_{i_{\ell-1}}\sigma_{\ell-1}\Pi^{j_{\ell-1}}]$. By construction, H'' consists of d layers of ℓ dyadic submatrices each. The private information consisting of the signature h (which gives rise to the polynomial g and the support L), the sequence $(i_0, \ldots, i_{\ell-1})$ of blocks, the sequence $(j_0, \ldots, j_{\ell-1})$ of dyadic permutation identifiers, and the sequence of scale factors $(\sigma_0, \ldots, \sigma_{\ell-1})$, relates the public code defined by (the systematic form of) H'' with the private code defined by H. The space occupied by the private information is thus about $ndm + \ell \lg(N/t) + \ell \lg t + \ell m$ bits.

Writing H'' in systematic form, when possible, involves a Gaussian elimination incurring some ℓ multiplications of such submatrices per layer to put each column of submatrices in echelon form; since there are d columns to process and d layers, the overall cost is about $d^2\ell$ products of dyadic $t \times t$ submatrices, implying a complexity $O(d^2\ell t^{\lg 3})$. For typical cryptosystems one can set $t = \tilde{O}(n)$ or even $t = O(n)$, and hence the cost of formatting is $O(n^{\lg 3})$ as long as d is a small constant, which is indeed the case in practice since maximum size reduction is achieved when $\mathbb{F}_p$ is a large proper subfield of $\mathbb{F}_q$ (see Section 5.1).

The resulting parity-check matrix defines a code of length n and dimension $k = n - dt$, and still consists of dyadic submatrices which can be represented by a signature of length t, and hence the whole matrix can be stored in an area a factor t smaller than a general matrix. During the process the dyadic submatrices are not guaranteed to be nonsingular as they are not associated to a Cauchy matrix any longer; should all the submatrices on a column be found to be singular during the Gaussian elimination (above or below the diagonal, according to the direction of this process) so that no pivot is possible, the whole column containing may be replaced by another block $B_{j'}$ chosen at random from H' as above.

The total space occupied by the essential part of the resulting generator (or parity-check) matrix over $\mathbb{F}_p$ is $dt \times (n - dt)/t = dk$ $\mathbb{F}_p$ elements, or mk bits – a factor t better than plain Goppa codes, which occupy $k(n - k) = mkt$ bits. The size reduction is about the same even if the matrix is kept in plain (rather than systematic) for, since in that case the matrix occupies mn 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 $O(v^3 3^u)$ rather than simply $O(3^u)$, 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 $\binom{N/t}{\ell} \times \ell! \times t^\ell \times (r-1)^\ell$, hence $\binom{N/t}{\ell} \times \ell! \times t^\ell \times (r-1)^\ell \times \prod_{i=0}^{\lceil \lg N \rceil} (q - 2^i)$ codes are possible in principle.

3.3 Decoding

While there are algorithms to efficiently decode a Goppa syndrome computed with a parity-check matrix H in alternant form, no such algorithm is known to directly decode a syndrome computed with a parity-check matrix Δ in systematic quasi-dyadic form.

However, knowledge of z, L and the overall scaled permutation P resulting from the process described in Section 3.2 is enough to build the easily decodable parity-check matrix H, which is linearly related to Δ as $H = S\Delta P$ for some nonsingular matrix S, which can be retrieved as $S = HP^\mathsf{T}\Delta^\mathsf{T}(\Delta\Delta^\mathsf{T})^{-1}$. Thus, given a syndrome $s = \Delta\, c^\mathsf{T}$ for some (corrupted) codeword c, compute $s' = Ss = Hc^\mathsf{T}$ and use this syndrome for decoding.

3.4 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

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

with eight 2×2 blocks $B_0, \ldots, B_7$ as indicated. From this we extract the shortened, rearranged and permuted sequence $H'' = [B_7\Pi^0 \mid B_5\Pi^1 \mid B_1\Pi^0 \mid B_2\Pi^1 \mid B_3\Pi^0 \mid B_6\Pi^1 \mid B_4\Pi^0]$ (for simplicity we choose all scale factors equal to 1), i.e.:

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

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:

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

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 Definitions 6 and 7) 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 [21] 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 [28] 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 [3].

Definition 10 (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 $\mathsf{w}(e) \leqslant w$ such that $He^\mathsf{T} = s^\mathsf{T}$?*

The corresponding problem for quasi-dyadic matrices reads:

Definition 11 (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 $\mathsf{w}(e) \leqslant w$ such that $He^\mathsf{T} = s^\mathsf{T}$?*

Theorem 3. *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 an 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 $\ell > 0$. Let u_ℓ be the first row of the identity matrix I_ℓ, i.e. $I_\ell = \Delta(u_\ell)$. Define $w = \ell w'$, the quasi-dyadic matrix $H \in \Delta(\mathbb{F}_q^\ell)^{r\times n}$ with blocks $H_{ij} = H'_{ij} I_\ell$, and the vector $s \in (\mathbb{F}_q^\ell)^r$ with blocks $s_i = s'_i u_\ell$. 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 $\mathsf{w}(e) \leqslant w$ such that $He^\mathsf{T} = s^\mathsf{T}$. Let $e'_i \in \mathbb{F}_q^n$ be the vector with elements $(e'_i)_j = e_{i+j\ell}$, so that the e'_j 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. Incidentally, the expected running time of all known algorithms for the 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.

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. A Karatsuba-like approach for dyadic convolution[2] leads to the asymptotic complexity $O(n^{\lg 3})$ for this operation and hence also for encryption. Sarwate's decoding method [23] sets the asymptotic cost of decryption at roughly $O(n^{\lg 3})$ as well.

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 $\Delta^{\mathsf{T}}\Delta = \Delta\Delta^{\mathsf{T}} = \Delta^2 = (\sum_i h_i)^2 I$, 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.

The cost of converting a dyadic matrix to systematic (echelon) form is the same as the cost of a dyadic convolution: write the original matrix as $M_{k\times n} = [A_{k\times k} \mid B_{k\times(n-k)}]$, invert the square dyadic matrix A in linear time, and then compute the echelon form $[I_k \mid A^{-1}B]$, which involves the product of two dyadic matrices (one of them possibly truncated) and hence has the claimed $O(n^{\lg 3})$ overall complexity. If M is a parity-check matrix, write it as $M_{r\times n} = [A_{r\times(n-r)} \mid B_{r\times r}]$ and obtain the systematic form $[B^{-1}A \mid I_r]$. Notice that, contrary to systems based on quasi-circulant matrices [8, Proposition 3.4], our proposal does not require a lengthy process, involving expensive $O(n^3)$ matrix rank computations to construct a generator matrix in suitable form, often larger than one would expect for a code of the given dimension.

Table 1 summarizes the asymptotic complexities of key generation, systematic formatting, and encryption or decryption. Preliminary tests in C/C++ for the range of code dimensions and lengths that are expected to be found in practice (see Section 5.1) indicate that vector multiplication by a quasi-dyadic matrix using iterative Karatsuba for purely dyadic submatrices runs twice as fast as the textbook algorithm.

Table 1. Cryptographic operation complexity relative to the code length n.

operation	generic	ours
Key generation	$O(n^3)$	$O(n \lg n)$
Systematic form	$O(n^3)$	$O(n^{\lg 3})$
Encrypt/Decrypt	$O(n^2)$	$O(n^{\lg 3})$

5.1 Suggested parameters

Several trade-offs are possible when choosing parameters for a particular application. One may wish to minimize the key size, or increase speed, or simplify the underlying arithmetic, or attaining a balance between them. We present here some non-exhaustive combinations. The number of possible codes is large; for instance, it is about 2^{655} for an original code length of $N = 2^{15}$ even for the least secure example below, at the 80-bit level entry on table 3.

[2] Regrettably the fast Walsh-Hadamard transform [12], which reduces the complexity of the dyadic convolution to $O(n \lg n)$, is only possible in fields of characteristic other than 2.

Table 2 shows the influence of varying the subfield degree while keeping the security level and the number of errors fixed. The 'size' columns give the bit size of the resulting parity-check matrix in plain and in systematic (echelon) form. In general, codes over larger subfields allow for smaller keys as already indicated in [2].

Table 2. Sample 128-bit security level parameters for a fixed number of errors ($t = 32$), using a subcode over the subfield $\mathbb{F}_{2^s}$ of $\mathbb{F}_{2^{16}}$.

s	n	k	size	
			plain	echelon
1	2752	2240	44032	35840
2	1696	1440	27136	23040
4	1024	896	16384	14336
8	576	512	9216	8192

Table 3 displays a different trade-off whereby the key size and the subfield are kept constant at the cost of varying the number of errors and the code length. The estimated security level on column 'level' refers to the approximate logarithmic cost of the best known attack according to the guidelines in [6]. The number of errors is always a power of 2 to enable maximum reduction of the key size.

Table 3. Sample parameters for a fixed key size (8192 bits, corresponding to $k = 512$), using a subcode over the subfield $\mathbb{F}_{2^8}$ of $\mathbb{F}_{2^{16}}$.

n	t	level
576	32	130
640	64	188
768	128	261

Table 4 contains a variety of balanced parameters for practical security levels. The target security level and the estimated actual cost of the best known attack according to the guidelines in [6] are shown on the columns labeled 'level'. We consider the size of the parity-check matrix obtained with our method in plain rather than systematic form. The 'shrink' column is the size ratio between the size occupied by a generic Goppa parity-check or generator matrix as suggested in [6], and a quasi-dyadic Goppa parity-check matrix for roughly the same security level. On the 'RSA' column we list the typical size of a (quantum-susceptible) RSA modulus at the specified security level (more accurate RSA estimates can be found in [20]). To assess our results against what can be achieved by different post-quantum settings, the 'NTRU' columns contains size-optimal and speed-optimal NTRU key sizes as suggested in the draft IEEE 1363.1 standard [13], and column 'cyclic' lists key sizes for quasi-cyclic codes of approximately the specified security level, as suggested in [2]. This last comparison illustrates how important and effective the ability to correct all design errors is to obtain short keys.

Table 4. Sample parameters for a subcode over the subfield $\mathbb{F}_{2^8}$ of $\mathbb{F}_{2^{16}}$.

level		n	k	t	size		shrink	RSA	NTRU		cyclic
target	actual				ours	generic			size	speed	
80	90	256	128	64	4096	460647	112	1024	–	–	6510
112	115	320	192	64	5120	1047600	205	2048	4411	7249	11160
128	135	384	256	64	6144	1537536	250	3072	4939	8371	20800
192	223	640	384	128	10240	4185415	409	7680	7447	11957	–
256	261	768	512	128	12288	7667855	624	15360	11957	16489	–

6 Conclusion and further research

We have described how to generate Goppa codes in quasi-dyadic form for cryptographic applications. Key sizes in 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, without loosing the ability to correct the design number of errors, sometimes even more. This brings the size of cryptographic keys to within a factor 4 or less of equivalent RSA keys (breaking even at high security levels), and 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 [28], and by Berger et al. for that family of codes [2], while retaining the ability to correct the full designed number of errors t instead of only $t/2$.

The complexity of all cryptographic operations in McEliece and related cryptosystems is reduced to no more than $O(n^{\lg 3})$. An obvious research problem is how to reduce the cost to $\tilde{O}(n)$ or to extend the proposed construction to higher characteristics for which $\tilde{O}(n)$ algorithms are immediately available. Another problem is that of finding the largest class of quasi-dyadic, efficiently decodable codes.

Other cryptosystems can also benefit from dyadic codes, e.g. entity identification and certain digital signatures for which double circulant codes have been proposed [9] could use dyadic codes instead, even random ones without a Goppa trapdoor.

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 [17], and achieving comparable size reduction. We leave this line of inquiry for future research since it falls outside the scope of this paper.

Acknowledgements

We are most grateful and indebted to Pierre-Louis Cayrel, Philippe Gaborit, Steven Galbraith, Robert Niebuhr, Christiane Peters, and Nicolas Sendrier 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. 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`.
3. 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.
4. D. J. Bernstein. List decoding for binary Goppa codes. Preprint, 2008. `http://cr.yp.to/papers.html#goppalist`.
5. D. J. Bernstein, J. Buchmann, and E. Dahmen. *Post-Quantum Cryptography*. Springer, 2008.
6. 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`.
7. N. Courtois, M. Finiasz, and N. Sendrier. How to achieve a McEliece-based digital signature scheme. In *Advances in Cryptology – Asiacrypt'2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 157–174, Gold Coast, Australia, 2001. Springer.
8. 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.
9. 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.
10. J. K. Gibson. Severely denting the Gabidulin version of the McEliece public key cryptosystem. *Designs, Codes and Cryptography*, 6(1):37–45, 1995.
11. 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.
12. M. N. Gulamhusein. Simple matrix-theory proof of the discrete dyadic convolution theorem. *Electronics Letters*, 9(10):238–239, 1973.
13. IEEE P1363 Working Group. *IEEE 1363-1: Standard Specifications for Public-Key Cryptographic Techniques Based on Hard Problems over Lattices (Draft)*, 2009. `http://grouper.ieee.org/groups/1363/lattPK/index.html`.
14. 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.
15. F. J. MacWilliams and N. J. A. Sloane. *The theory of error-correcting codes*, volume 16. North-Holland Mathematical Library, 1977.
16. 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`.
17. D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 16(4):365–411, 2007.
18. 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.
19. H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.
20. European Network of Excellence in Cryptology. ECRYPT yearly report on algorithms and keysizes (2007-2008). D.SPA.28 Rev. 1.1, IST-2002-507932 ECRYPT, 07/2008, 2008. `http://www.ecrypt.eu.org/ecrypt1/documents/D.SPA.28-1.1.pdf`.
21. 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`.
22. N. J. Patterson. The algebraic decoding of Goppa codes. *IEEE Transactions on Information Theory*, 21(2):203–207, 1975.
23. D. V. Sarwate. On the complexity of decoding Goppa codes. *IEEE Transactions on Information Theory*, 23(4):515–516, 1977.
24. 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`.
25. N. Sendrier. Finding the permutation between equivalent linear codes: the support splitting algorithm. *IEEE Transactions on Information Theory*, 46(4):1193–1203, 2000.
26. V. Sidelnikov and S. Shestakov. On cryptosystems based on generalized Reed-Solomon codes. *Discrete Mathematics*, 4(3):57–63, 1992.
27. K. K. Tzeng and K. Zimmermann. On extending Goppa codes to cyclic codes. *IEEE Transactions on Information Theory*, 21:721–716, 1975.

28. 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.