How Can Reed-Solomon Codes Improve Steganographic Schemes?

Caroline Fontaine[⋆] and Fabien Galand

CNRS/IRISA-TEMICS,
Campus de Beaulieu, 35 042 Rennes cedex, France
caroline.fontaine@irisa.fr, fabien.galand@irisa.fr

Abstract. The use of syndrome coding in steganographic schemes tends to reduce distortion during embedding. The more complete model comes from the wet papers [FGLS05] which allow to lock positions that cannot be modified. Recently, BCH codes have been investigated, and seem to be good candidates in this context [SW06]. Here, we show that Reed-Solomon codes are twice better with respect to the number of locked positions and that, in fact, they are optimal. We propose two methods for managing these codes in this context: the first one is based on a naive decoding process through Lagrange interpolation; the second one, more efficient, is based on list decoding techniques and provides an adaptive trade-off between the number of locked positions and the embedding efficiency.

1 Introduction

Steganography aims at sending a message through a cover-medium, in an *undetectable* way. *Undetectable* means that nobody, except the intended receiver of the message, should be able to tell if the medium is carrying a message or not [Sim84]. Hence, if we speak about still images as cover-media, the embedding should work with the smallest possible distortion, but also not being detectable with the quite powerful analysis tools available [BW04, Fra02]. A lot of papers have been published on this topic, and it appears that modeling the embedding and detection/extraction processes with an error correcting code point of view, usually called matrix embedding by the steganographic community, may be helpful to achieve these goals [Cra98, GK03, FGLS05, FGS05a, FGS06, FS06, SW06]. The main interest of this approach is that it decreases the number of component modifications during the embedding process. As a side effect, it was remarked in [FGLS05] that matrix embedding could be used to provide an effective answer to the adaptive selection channel: the sender can embed the messages adaptively with the cover-medium to minimize the distortion, and the receiver can extract the messages without being aware of the sender's choices. A typical steganographic application is the perturbed quantization [FGS05b]: during quantization process, e.g. JPEG compression, real values v have to be rounded between possible quantized values $x_0, ..., x_j$; when v lies close to the middle of an interval

⋆ Corresponding author.

$[x_i, x_{i+1}]$, one can choose between x_i and x_{i+1} without adding too much distortion. This embeds messages under the condition that the receiver does not need to know which positions where modified.

It has been shown that if random codes may seem interesting for their asymptotic behavior, they impose to solve really hard problems: syndrome decoding and covering radius computation, which are proved to be NP-complete and Π_2-complete respectively (the Π_2 complexity class includes the NP class) [Var97, McL84]. Moreover, no efficient decoding algorithm is known, even for a small non trivial family of codes. From a practical point of view, this implies that the related steganographic schemes are too much complex to be considered as acceptable for real life applications. Hence, it is of great interest to have a deeper look at other kinds of codes, structured codes, which are more accessible and lead to efficient decoding algorithms. In this way, some previous papers studied the Hamming code [Cra98, Wes01, FGS05a], the Simplex code [FS06] and BCH codes [SW06]. Here, we focus on this latter paper, that pointed out the interest in using BCH codes. The authors distinguish two cases, as previously introduced in [FGLS05]. The first one is the more classical one: the embedder modifies any position of the cover-data (a vector which is extracted from the cover-medium, and processed by the encoding scheme), the only constraint being the maximum number of modifications. In this case, they showed that BCH codes behave well, but also pointed out that choosing the most appropriate code among the BCH family is quite hard: we do not know good complete syndrome decoding algorithm for BCH codes. In the second case, some positions are locked and cannot be used for embedding; this is due to the fact that modifying these positions lead to a degradation of the cover-medium that is noticeable. Hence, in order to remain undetectable, the sender restricts himself to keep these positions and lock them. This case is more realistic. The authors showed there is a trade-off between the number of elements that can be locked and the efficiency of the code.

Here, we propose to focus on a particular family of BCH codes: the Reed-Solomon (RS) codes. We first recall in Section 2 the framework of matrix embedding/syndrome coding. Then, we discuss the interest of using Reed-Solomon codes in this context: in Section 3, Reed-Solomon codes are presented, explicitly showing in Section 4 how they can improve realistic steganographic schemes. We show in Section 4.1 that with these codes we can go beyond the limits of BCH codes: we can lock twice the number of positions. In fact, we see that RS codes are optimal according to this criterion, since they enable to manage as many locked positions as possible. In Section 4.2, we also propose an improved algorithm based on Guruswami-Sudan list-decoding, that enables to make an adaptive trade-off between the embedding efficiency and the number of locked positions.

Before going deeper in the subject, please note that we made the choice to represent vectors horizontally . For general references to error correcting codes, we orientate the reader towards [HP03].

2 Syndrome Coding

The behavior of a steganographic algorithm can be sketched in the following way: a *cover-medium* is processed to extract a sequence of bits v, sometimes called *cover-data*; v is modified into s to embed the message m; s is sometimes called the *stego-data*; modifications on s are translated on the cover-medium to obtain the *stego-medium*. Here, we assume that the detectability of the embedding increases with the number of bits that must be changed to go from v to s (see [Wes01, KDR06] for some examples of this framework).

Syndrome coding deals with this number of changes. The key idea is to use some syndrome computation to embed the message into the cover-data. In fact, this scheme uses a linear code $\mathcal{C}$, more precisely its cosets, to hide m. A word s hides the message m if s lies in a particular coset of $\mathcal{C}$, related to m. Since cosets are uniquely identified by the so called syndromes, embedding/hiding consists exactly in searching s with syndrome m, close enough to v.

We first set up the notation and describe properly the syndrome coding scheme, and its inherent problems. Let $\mathbb{F}_q = GF(q)$ denote the finite field with q elements[1]. Let $v \in {\mathbb{F}_q}^n$ denote the cover-data and $m \in {\mathbb{F}_q}^r$ the message. We are looking for two mappings, embedding Emb and extraction Ext, such that:

$$\forall (v,m) \in {\mathbb{F}_q}^n \times {\mathbb{F}_q}^r,\ \mathrm{Ext}(\mathrm{Emb}(v,m)) = m \tag{1}$$

$$\forall (v,m) \in {\mathbb{F}_q}^n \times {\mathbb{F}_q}^r,\ d_H(v, \mathrm{Emb}(v,m)) \leq T \tag{2}$$

Eq. (1) means that we want to recover the message in all cases; Eq. (2) means that we authorize the modification of at most T coordinates in the vector v.

Let $\mathcal{C}$ be a q-ary linear code of length n, dimension k and parity check matrix H. That is, $\mathcal{C} = \{c \mid c \cdot H^t = 0\}$ is a vector subspace of ${\mathbb{F}_q}^n$ of dimension k. The *syndrome* of a vector y, with respect to the code $\mathcal{C}$, is the row vector $y \cdot H^t$ of length $n-k$; we denote it by $E(y)$. The *covering radius* of $\mathcal{C}$ is the minimum integer ρ such that $\{E(y) \mid w_H(y) \leq \rho\} = {\mathbb{F}_q}^{n-k}$. Let us denote by D the mapping that associates with a syndrome m a vector a of Hamming weight less than or equal to ρ, and which syndrome is precisely equal to m (that is, $w_H(a) \leq \rho$ and $E(a) = a \cdot H^t = m$). Remark that effective computation of D is the complete syndrome decoding problem, which is hard. It is quite easy to show that the scheme defined by

$$\mathrm{Emb}(v,m) = v + D(m - E(v))$$
$$\mathrm{Ext}(y) = E(y) = y \cdot H^t$$

enables to embed messages of length $r = n-k$ in a cover-data of length n, while modifying at most $T = \rho$ elements of the cover-data.

The parameter $(n-k)/\rho$ represents the (worst) embedding efficiency[2], that is, the number of embedded symbols per embedding changes in the worst case.

[1] Recall that when q is a power of two, elements of $\mathbb{F}_q$ can be regarded as blocks of bits.

[2] Remark this is with respect to symbols and not bits. If elements of $\mathbb{F}_q$ are viewed as blocks of ℓ bits, changing a symbol by an other roughly leads to $\ell/2$ flips.

In a similar way, one defines the average embedding efficiency $(n-k)/\omega$, where ω is the average weight of the output of D for uniformly distributed inputs.

A problem raised by the syndrome coding, as presented above, is that any position in the cover-data v can be changed. In some cases, it is more reasonable to keep some coordinates unchanged because they would produced too big artifacts in the stego-medium. This can be done in the following way. Let $\mathcal{I} = \{i_1, ..., i_j\}$ be the coordinates that must not be changed, let $H_{\mathcal{I}}$ be the matrix obtained from H by removing[3] the columns $i_1, ..., i_j$, and $E_{\mathcal{I}}$ and $D_{\mathcal{I}}$ the corresponding mappings. That is, $E_{\mathcal{I}}(y) = y \cdot H_{\mathcal{I}}^t$ for $y \in \mathbb{F}_q{}^{n-|\mathcal{I}|}$, and $D_{\mathcal{I}}(m) \in \mathbb{F}_q{}^{n-|\mathcal{I}|}$ is a vector of weight at most $\rho_{\mathcal{I}}$ such that its syndrome, with respect to $H_{\mathcal{I}}$, is m. Here, $\rho_{\mathcal{I}}$ is the covering radius of $\mathcal{C}_{\mathcal{I}}$, the code obtained from $\mathcal{C}$ by removing the coordinates in $\mathcal{I}$ from all the codewords. Of course, this is also the code of parity check matrix $H_{\mathcal{I}}$. Finally, let us define $D_{\mathcal{I}}^*$ as the vector of $\mathbb{F}_q{}^n$ such that the coordinates in $\mathcal{I}$ are zeros and the vector obtained by removing these coordinates is precisely $D_{\mathcal{I}}$. Now, we have $D_{\mathcal{I}}^*(m) \cdot H = D_{\mathcal{I}}(m) \cdot H_{\mathcal{I}}^t = m$ and, by definition, $D_{\mathcal{I}}^*(m)$ has zeros at coordinates set by $\mathcal{I}$. Naturally, the scheme defined by

$$\begin{aligned} \mathrm{Emb}(v, m) &= v + D_{\mathcal{I}}^*(m - E(v)) \\ \mathrm{Ext}(y) &= E(y) = y \cdot H^t \end{aligned}$$

performs syndrome coding without disturbing the positions in $\mathcal{I}$. But, it is worth noting that for some sets $\mathcal{I}$, the mapping $D_{\mathcal{I}}$ cannot be defined for all possible values of m because the equation $y \cdot H_{\mathcal{I}}^t = m$ has no solution. This always happens when $|\mathcal{I}| > k$, since $H_{\mathcal{I}}$ has dimension $(n-k) \times (n - |\mathcal{I}|)$, but can also happen for smaller sets.

Please, keep in mind that using syndrome coding leads to essentially two problems. First, the parameters n, r, ρ depend on the choice of $\mathcal{C}$, and most of the time ρ is hard to compute. Second, the mapping D is difficult to compute.

3 What Reed-Solomon Codes Are, and Why They May Be Interesting

Reed-Solomon codes over the finite field $\mathbb{F}_q$ are optimal linear codes. The *narrow-sense RS codes* have length $n = q - 1$ and can be defined as a particular subfamily of the BCH codes. But, we prefer the alternative, and larger, definition as an evaluation code, which leads to the *Generalized Reed-Solomon codes (GRS codes)*.

Roughly speaking, a GRS code of length $n \leq q$ and dimension k is a set of words corresponding to polynomials of degree less than k evaluated over a subset of $\mathbb{F}_q$ of size n. More precisely, let $\{\gamma_0, ..., \gamma_{n-1}\}$ be a subset of $\mathbb{F}_q$ and define $ev(P) = (P(\gamma_0), P(\gamma_1), \ldots, P(\gamma_{n-1}))$, for $P(X)$ a polynomial over $\mathbb{F}_q$. Then, we define $GRS(n, k)$ as

[3] In coding theory, this is called *shortening* the code on $\mathcal{I}$: we only keep codewords that have zero on $\mathcal{I}$, and then we remove the coordinates set by $\mathcal{I}$.

$$GRS(n,k) = \{ev(P) \mid \deg(P) < k\} \ .$$

This definition, *a priori*, depends on the choice of the γ_i and the order of evaluation, but as far as we are concerned, only the number of γ_i is important, so we consider a fixed set of γ_i and a fixed order. Remark that when $\gamma_i = \beta^i$ with β a primitive element of $\mathbb{F}_q$ and $i \in \{0, ..., q-2\}$, we obtain the *narrow-sense Reed-Solomon codes.*

GRS codes are optimal: they reach the Singleton bound, that is, the minimal distance of $GRS(n,k)$ is $d = n - k + 1$, which is the largest possible. On the other hand, the covering radius of $GRS(n,k)$ is known and equal to $\rho = n - k$.

Concerning the evaluation function, recall that if we consider $n \leq q$ elements of $\mathbb{F}_q$, then it is known that there is a unique polynomial of degree at most $n-1$ taking particular values on these n elements. This means that for every v in ${\mathbb{F}_q}^n$, one can find a polynomial V with $\deg(V) \leq n-1$, such that $ev(V) = v$; moreover, V is unique. Of course, ev is a linear mapping, $ev(\alpha{\cdot}P+\beta{\cdot}Q) = \alpha{\cdot}ev(P)+\beta{\cdot}ev(Q)$ for any polynomials P, Q and field elements α, β.

For convenience, in the sequel, we identify any polynomial of degree less than n with a vector of length n, the i-th coordinate of the vector being the coefficient of the monomial of degree i. Thus, the evaluation mapping can be represented by the matrix

$$\Gamma = \begin{pmatrix} ev(X^0) \\ ev(X^1) \\ ev(X^2) \\ \cdots \\ ev(X^{n-1}) \end{pmatrix} = \begin{pmatrix} \gamma_0^0 & \gamma_1^0 & \cdots & \gamma_{n-1}^0 \\ \gamma_0 & \gamma_1 & \cdots & \gamma_{n-1} \\ \gamma_0^2 & \gamma_1^2 & \cdots & \gamma_{n-1}^2 \\ & & \vdots & \\ \gamma_0^{n-1} & \gamma_1^{n-1} & \cdots & \gamma_{n-1}^{n-1} \end{pmatrix} \ .$$

If we denote by $\mathrm{Coeff}(V) \in {\mathbb{F}_q}^n$ the vector consisting in the coefficients of V, then $\mathrm{Coeff}(V) \cdot \Gamma = ev(V)$. On the other hand, Γ being non-singular, its inverse Γ^{-1} computes $\mathrm{Coeff}(V)$ from $ev(V)$. For our purpose, it is noteworthy that the coefficients of monomials of degree at least k can be easily computed from $ev(V)$: splitting Γ^{-1} in two parts,

$$\Gamma^{-1} = (\underbrace{\ A\ }_{k \text{ columns}} \qquad \underbrace{\ B\ }_{n-k \text{ columns}} \ ) \ ,$$

$ev(V) \cdot B$ is precisely the coefficients vector of the monomials of degree at least k in V. In fact, B is the transpose of a parity check matrix of the GRS code since a vector c is an element of the code if and only if we have $c \cdot B = 0$. So, instead of B, we write H^t, as it is usually done.

Now, let us look at the cosets of $GRS(n,k)$. A coset is a set of the type $y + GRS(n,k)$, with $y \in {\mathbb{F}_q}^n$ not in $GRS(n,k)$. As usual with linear codes, a coset is uniquely identified by the vector $y \cdot H^t$, syndrome of y. In the case of GRS code, this vector consists in the coefficients of monomials of degree at least k.

4 What Can Reed-Solomon Codes Do

Our problem is the following. We have a vector v of length n of symbols of $\mathbb{F}_q$, extracted from the cover-medium, and a message m of length r of symbols of $\mathbb{F}_q$. We want to modify v into s such that m is embedded in s, changing at most T coordinates in v.

The basic principle is to use syndrome coding with a GRS code: we use the cosets of $GRS(n, k)$ to embed the message, finding a vector s in the proper coset, close enough to v. Thus, k must be equal to $n - r$, and we suppose we have fixed $\gamma_0,...,\gamma_{n-1} \in \mathbb{F}_q$, constructed the matrix Γ whose i-th row is $ev(X^i)$, and inverted it. In particular, we denote by H^t the last $n - k$ columns of Γ^{-1} and, therefore, according to the previous section, H is a parity-check matrix. Recall that a word s embeds the message m if $s \cdot H^t = m$.

To construct s, we need a word y such that its syndrome is $m - v \cdot H^t$; thus, we can set $s = y + v$, which leads to $s \cdot H^t = y \cdot H^t + v \cdot H^t = m$. Moreover, the Hamming weight of y is precisely the number of changes we apply to go from v to s; so, we need $w(y) \leq T$.

When T is equal to the covering radius of the code corresponding to H, such a vector y always exists. But, explicit computation of such a vector y, known as the bounded syndrome decoding problem, is proved to be NP-hard for general linear codes. Even for well structured codes, we usually do not have polynomial time (in the length n) algorithm to solve the bounded syndrome decoding problem up to the covering radius. This is precisely the problem faced by [SW06].

GRS codes overcome this problem in a nice fashion. It is easy to find a vector with syndrome m: let us consider the polynomial $M(X)$ that has coefficient m_i for the monomial X^{k+i}, $i \in \{0, ..., n-1-k\}$; according to the previous section, we have $ev(M) \cdot H^t = m$. Now, finding y can be done by computing a polynomial P of degree less than k such that for at least k elements $\gamma \in \{\gamma_0, ..., \gamma_{n-1}\}$ we have $P(\gamma) = M(\gamma) - V(\gamma)$. With such a P, the vector $y = ev(M - V - P)$ has at least k coordinates equal to zero, and the correct syndrome value. Hence, T can be as high as the covering radius $\rho = n - k$, and the challenge lies in the construction of P.

It is noteworthy to remark that locking the position i, that is, requiring $s_i = v_i$, is equivalent to ask for $y_i = 0$ and, thus, $P(\gamma_i) = M(\gamma_i) - V(\gamma_i)$.

4.1 A Simple Construction of *P*

Using Lagrange Interpolation. A very simple way to construct P is by using the Lagrange interpolating polynomials. We choose k coordinates $\mathcal{I} = \{i_1, ..., i_k\}$, and compute

$$P(X) = \sum_{i \in \mathcal{I}} (M(\gamma_i) - V(\gamma_i)) \cdot L_{\mathcal{I}}^{(i)}(X) \ ,$$

where $L_{\mathcal{I}}^{(i)}$ is the unique polynomial of degree at most $k - 1$ taking values 0 on γ_j, $j \neq i$ and 1 on γ_i, that is,

$$L_{\mathcal{I}}^{(i)}(X) = \prod_{j \in \mathcal{I} \setminus \{i\}} (\gamma_i - \gamma_j)^{-1}(X - \gamma_j) \ .$$

The polynomial P we obtain this way clearly satisfies $P(\gamma_i) = V(\gamma_i) - M(\gamma_i)$ for every $i \in \mathcal{I}$ and, thus, we can set $y = ev(M - V - P)$. As pointed out earlier, since, for $i \in \mathcal{I}$, we have $y_i = 0$, we also have $s_i = v_i + y_i = v_i$, *i.e.* positions in $\mathcal{I}$ are locked.

The above proposed solution has a nice feature: we can choose the coordinates on which s and v are equal, and this does not require any loss in computational complexity nor embedding efficiency. This means that we can perform the syndrome decoding directly with the additional requirement of wet papers, keeping unchanged the coordinates whose modifications are detectable.

So far, what do GRS codes allow?

Optimal Management of Locked Positions. We can embed $r = n - k$ elements of $\mathbb{F}_q$, changing not more than $T = n - k$, so the embedding efficiency is equal to 1 in the worst case. But, we can lock *any* k positions to embed our information.

This is to be compared with [SW06], where BCH codes are used. The maximal number of locked positions, without failing to embed the message m, is experimentally estimated to be $k/2$. To be able to lock up to $k-1$ positions, it is necessary to allow a non-zero probability of non embedding. It is also noteworthy that the average embedding efficiency decreases fast.

In fact, embedding $r = n - k$ symbols while locking k symbols amongst n is optimal. We said in Section 2 that locking the positions in $\mathcal{I}$ leads to an equation $y \cdot H_{\mathcal{I}}^t = m$, where $H_{\mathcal{I}}$ has dimension $(n-k) \times (n - |\mathcal{I}|)$. So, when $|\mathcal{I}| > k$, there exist some values m for which there is no solution y. On the other hand, let us suppose we have a code with parity check matrix H such that for any $\mathcal{I}$ of size k, and any m, this equation has a solution, that is, $H_{\mathcal{I}}$ is invertible. This means that any $(n-k) \times (n-k)$ submatrix of H is invertible. But, it is known that this is equivalent to require the code to be MDS (see for example [HP03, Cor 1.4.14]), which is the case of GRS code. Hence, GRS codes are optimal in the sense that we can lock as many positions as possible, that is, up to k for a message length of $r = n - k$.

4.2 A More Efficient Construction of P

Using List Decoding. A natural idea to improve the results of the last section is to use decoding algorithms for GRS codes, whenever it is possible. Such algorithms compute, from a vector $ev(Q)$, polynomials P of degree at most $k-1$, such that $ev(P)$ are close to $ev(Q)$, according to the Hamming distance. Stated differently, they provide good approximations of Q. Using these algorithms reduce the average number of changes required by the embedding and, thus, improve the average efficiency.

Essentially, the output of the decoding algorithms may be: a single polynomial P, if it exists, such that the vector $ev(P)$ is at distance at most $\lfloor (n-k+1)/2 \rfloor$

from $ev(Q)$ (remark that if such a P exists, it is unique), and nothing otherwise; or, a list of all polynomials P such that the vectors $ev(P)$ are at distance at most T from $ev(Q)$.

The second case corresponds to the so called list decoding; an efficient algorithm for GRS codes was initially provided by [Sud97], and was improved by [GS99], leading to what is known as the Guruswami-Sudan algorithm. Clearly, list decoding is the more interesting: like the first kind of decoding, it provides the solution of minimum weight if it exists; moreover, the possibility to choose between different vectors improves the undetectability targeted by the steganographic applications.

Guruswami-Sudan algorithm outlines. The reader interested in detailed exposition may refer to [GS99, McE03, HP03]. The Guruswami-Sudan algorithm uses a parameter called the interpolation multiplicity μ. For an input vector $(a_0, ..., a_{n-1})$, the algorithm computes a bivariate polynomial $R(X, Y)$ such that[4] each couple (γ_i, a_i) is a root of R with multiplicity μ. The second and last step is to compute the list of factors of R, of the form $Y - P(X)$, with $\deg(P) \leq k-1$. For a fixed μ, the list contains all the polynomials which are at distance at most $\lambda_\mu \approx n - \sqrt{(1 + \frac{1}{\mu})(k-1)n}$. The maximum decoding radius is, thus, $\lambda_{GS} = n - 1 - \sqrt{n \cdot (k-1)}$. Moreover, the overall algorithm can be performed in less than $\mathcal{O}(n^2\mu^4)$ arithmetic operations over $\mathbb{F}_q$.

Guruswami-Sudan for shortened GRS codes. The Guruswami-Sudan algorithm can be used for decoding shortened GRS codes: for a fixed set $\mathcal{I}$ of indices, we are looking for polynomials P such that $\deg(P) < k$, $P(\gamma_i) = 0$ for $i \in \mathcal{I}$ and $P(\gamma_i) = Q(\gamma_i)$ for as many $i \notin \mathcal{I}$ as possible. Such P can be written $P(X) = F(X)G(X)$ with $F(X) = \prod_{i \in \mathcal{I}}(X - \gamma_i)$. Hence, decoding the shortened code reduces to obtain G such that $\deg(G) < k - |\mathcal{I}|$ and $G(\gamma_i) = Q(\gamma_i)/F(\gamma_i)$ for as many $i \notin \mathcal{I}$ as possible. This means, we are using the GS algorithm to decode a word of $GRS(n - |\mathcal{I}|, k - |\mathcal{I}|)$.

Algorithm Description. Our general scheme becomes: try to perform list decoding on $ev(M - V)$, in order to get a P as close as possible to $ev(M - V)$; if it fails, fall back onto Lagrange interpolation – as in the previous section – to compute P.

In fact, it is still possible to keep some positions locked: Let $\mathcal{I}$ be the set of coordinates to be untouched, construct the polynomial P such that $P(\gamma_i) = M(\gamma_i) - V(\gamma_i)$; Let us consider $Y = M - V - P$ and use GS decoding to compute an approximation U of Y of degree at most $k-1$, such that $U(\gamma_i) = 0$ for $i \in \mathcal{I}$; If GS decoding fails, add a new position to $\mathcal{I}$ and retry until it succeeds or $\mathcal{I} = k$; If no GS decoding succeeds (and, so, $\mathcal{I} = k$), define $U(X) = 0$; Finally, the stegoword is $v + ev(Y - U)$.

Figure 1 depicts the complete algorithm. The description uses two external procedures. The `GSdecode` procedure refers to the Guruswami-Sudan list decod-

[4] R must also satisfy another important constraint on the so called weighted degree.

ing: it decodes the polynomial $Y(X)$ of degree at most $k-1$, with respect to the code $GRS(n,k)$ defined by the evaluation on (γ_i), and shortened on positions set by $\mathcal{I}$. So, this procedure returns a good approximation $U(X)$ of $Y(X)$, on the evaluation set, of degree less than $k-1$, with the additional condition that $U(\gamma_i)=0$ for $i \in \mathcal{I}$. Remark that when $\mathcal{I}=\emptyset$, we simply use the GS decoding, whereas when $\mathcal{I} \neq \emptyset$, we use the modified decoding for shortened codes. The `selectposition` procedure returns an integer from the set given as a parameter. This procedure is used to choose the new position to lock before retrying the list decoding.

The correctness of this algorithm follows from the fact that through the whole algorithm we have $ev(Y)\cdot H^t = m - v\cdot H^t$ and $Y(\gamma_i)=0$ for $i \in \mathcal{I}$.

```
Inputs:  $v=(v_0,...,v_{n-1})$, the cover-data
         $m=(m_0,...,m_{n-k-1})$, symbols to hide
         $\mathcal{I}$, set of coordinates to remain unchanged, $|\mathcal{I}| \leq k$
Output:  $s=(s_0,...,s_{n-1})$, the stego-data
         $(s\cdot H^t = m;\ s_i = v_i,\ i\in\mathcal{I};\ d_H(s,v)\leq n-k)$

 1: $V(X) \Longleftarrow v_0X^0+\cdots+v_{n-1}X^{n-1}$
 2: $M(X) \Longleftarrow m_0X^k+\cdots+m_{n-k-1}X^{n-1}$
 3: $Y(X) \Longleftarrow M(X)-V(X)$
 4: for all $i\in\mathcal{I}$ do
 5:    $L_{\mathcal{I}}^{(i)}(X) \Longleftarrow \prod_{j\in\mathcal{I}\setminus\{i\}} (\gamma_i-\gamma_j)^{-1}(X-\gamma_j)$
 6: end for
 7: $P(X) \Longleftarrow \sum_{i\in\mathcal{I}} a_i\cdot L_{\mathcal{I}}^{(i)}(X)$
 8: $Y(X) \Longleftarrow Y(X)-P(X)$
 9: while $|\mathcal{I}|<k$ and GSdecode$(Y(X),\mathcal{I})=\emptyset$ do
10:    $i \Longleftarrow$ selectposition$(\{0,\ldots,n-1\}\setminus\mathcal{I})$
11:    $\mathcal{I} \Longleftarrow \mathcal{I}\cup\{i\}$
12:    $L_{\mathcal{I}}^{(i)}(X) \Longleftarrow \prod_{j\in\mathcal{I}\setminus\{i\}} (\gamma_i-\gamma_j)^{-1}(X-\gamma_j)$
13:    $Y(X) \Longleftarrow Y(X)-Y(\gamma_i)\cdot L_{\mathcal{I}}^{(i)}(X)$
14: end while
15: if GSdecode$(Y(X),\mathcal{I})\neq\emptyset$ then
16:    $U(X) \Longleftarrow$ GSdecode$(Y(X),\mathcal{I})$
17:    $Y(X) \Longleftarrow Y(X)-U(X)$
18: end if
19: $s \Longleftarrow v+ev(Y)$
20: return $s$
```

Fig. 1. Algorithm for embedding with locked positions using a $GRS(n,k)$ code, $(\gamma_0,...,\gamma_{n-1})$ fixed. It embeds $r=n-k$ symbols of $\mathbb{F}_q$ with up to k locked positions and at most $n-k$ changes.

Analysis. The most important property of an embedding algorithm is the number of changes introduced during the embedding. This analysis, for our algorithm, depends on two parameters.

The first parameter is the probability $p(n,k)$ that list decoding of a word in ${\mathbb{F}_q}^n$ outputs a non-empty list of codewords in GRS(n,k). We denote by $q(n,k)$ the probability of the complementary event, namely the return of an empty list. Thus, the probability that the first $\ell-1$ list decodings fail and the ℓ-th succeeds is $p(n-|\mathcal{I}|-\ell, k-|\mathcal{I}|-\ell)\prod_{e=0}^{\ell-1} q(n-|\mathcal{I}|-e, k-|\mathcal{I}|-e)$.

The second parameter is the average distance $\delta(n,k)$ between the closest codewords in the (non-empty) list and the word to decode. This last parameter leads to the average number of changes required to perform the embedding:

$$\omega = \left(\sum_{\ell=0}^{k'-1} \delta'(\ell)\cdot p'(\ell)\prod_{e=0}^{\ell-1} q'(e)\right) + (n-k)\prod_{e=0}^{k'-1} q'(e) \ ,$$

where $p'(e) = p(n-|\mathcal{I}|-e, k-|\mathcal{I}|-e)$, $q'(e) = q(n-|\mathcal{I}|-e, k-|\mathcal{I}|-e)$ and $\delta'(e) = \delta(n-|\mathcal{I}|-e, k-|\mathcal{I}|-e)$.

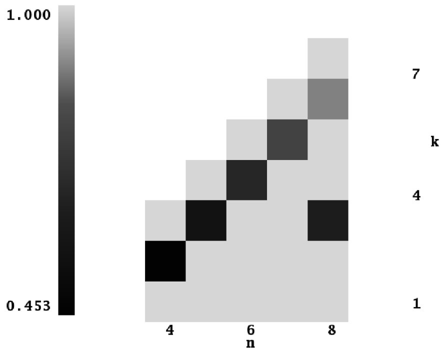
Estimating p and δ. To (upper) estimate $p(n,k)$, we proceed as follows. Let us denote by Z the random variable equal to the size of the output list of the decoding algorithm. The Markov inequality yields $Pr(Z \geq 1) \leq \mathbb{E}(Z)$, where $\mathbb{E}(Z)$ denotes the expectation of Z. But, $Pr(Z \geq 1)$ is the probability that the list is non-empty and, thus, $Pr(Z \geq 1) = p(n,k)$. Now, $\mathbb{E}(Z)$ is the average number of elements in the output list, but this is exactly the average number of codewords in a Hamming ball of radius λ_{GS}. Unfortunately, no adequate information can be found in the literature to properly estimate it; the only paper studying a similar quantity is [McE03], but it cannot be used for our $\mathbb{E}(Z)$. So, we set

$$\mathbb{E}(Z) = \frac{q^k}{q^n}\cdot V_{\lambda_{GS}} = \frac{\displaystyle\sum_{i=0}^{\lambda_{GS}} (q-1)^i \binom{n}{i}}{q^{n-k}} \ ,$$

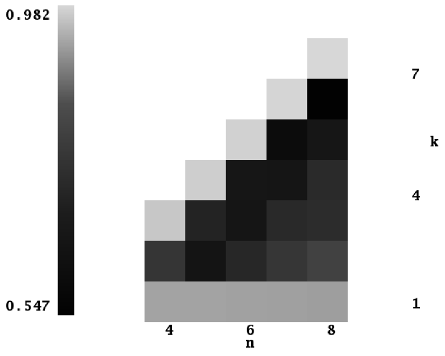
where $V_{\lambda_{GS}}$ is the volume of a ball of radius λ_{GS}. This would be the correct value if GRS codes were *random* codes over $\mathbb{F}_q$ of length n, with q^k codewords uniformly drawn from ${\mathbb{F}_q}^n$. That is, we estimate $\mathbb{E}(Z)$ as if GRS codes were random codes. Thus, we use $\overline{p} = \min(1, q^{k-n}V_{\lambda_{GS}})$ to upper estimate p.

The second parameter we need is the average number of changes required when the list is non-empty. We consider that the closest codeword is uniformly distributed over the ball of radius λ_{GS} and, therefore, we have

$$\delta(n,k) = \frac{\displaystyle\sum_{i=0}^{\lambda_{GS}} (q-1)^i \binom{n}{i} i}{V_{\lambda_{GS}}} \ .$$

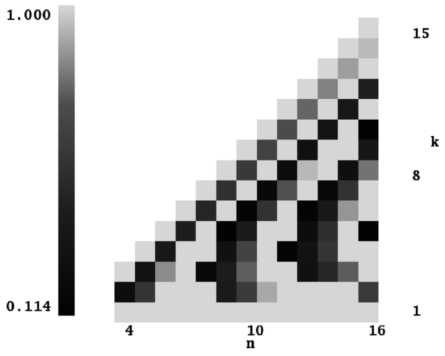


(a) $\overline{p}$

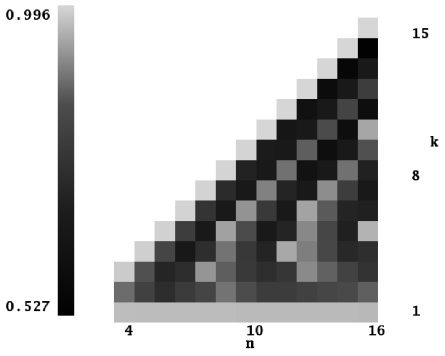


(b) α

Fig. 2. Figure (a) plots the estimated probability $\overline{p}$ of list decoding success on a random input vector for $GRS(n, k)$ over $\mathbb{F}_8$. Figure (b) plots the relative average number of changes α. As usual, n is the length and k the dimension.



(a) $\overline{p}$



(b) α

Fig. 3. Same as figure 2 for $\mathbb{F}_{16}$

A Simplified Analysis. A simple (upper) estimate of the average number of changes can be obtained by setting $\mathcal{I} = \emptyset$ and considering that if the first list decoding fails, the others will fail too. Doing so, we clearly underestimate the performance of our algorithm. This leads to the very simple quantity

$$\alpha = \frac{\delta(n,k) \cdot \overline{p}(n,k) + (n-k) \cdot (1 - \overline{p}(n,k))}{n-k}.$$

This value is plotted in Figures 2, 3, 4, 5 and 6 for small values of q (the number of elements of the field). For each figure, the left part (a) plots $\overline{p}$ and the right part (b) plots α.

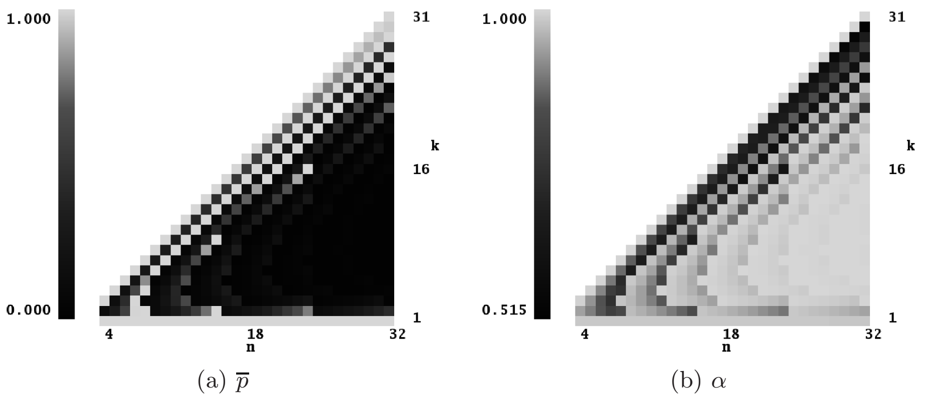


Fig. 4. Same as figure 2 for $\mathbb{F}_{32}$

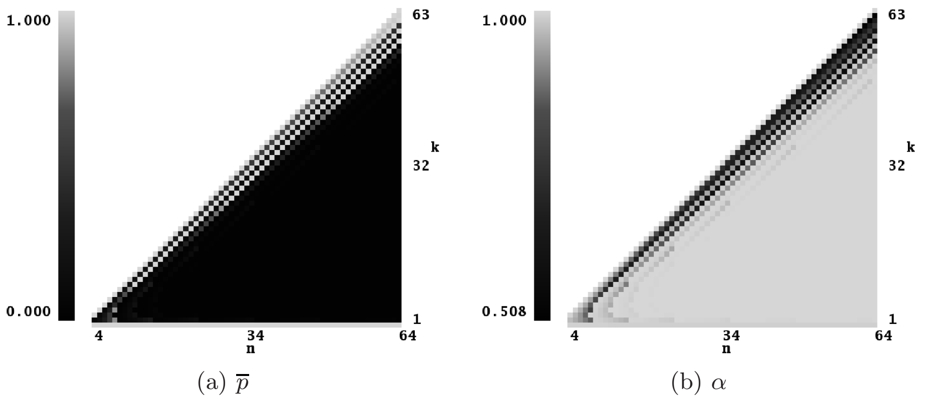
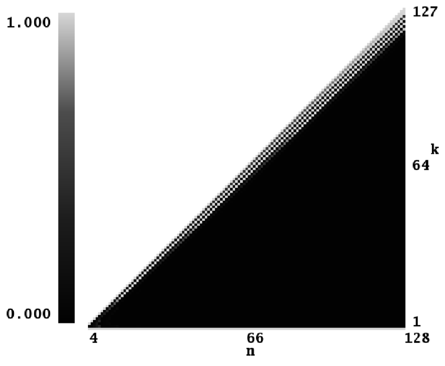
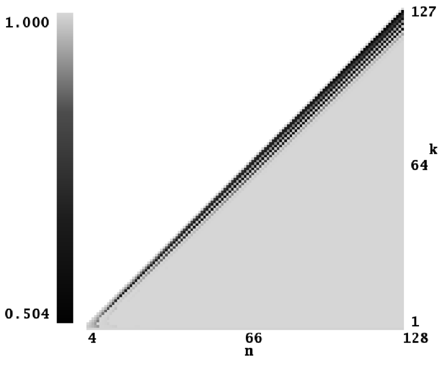


Fig. 5. Same as figure 2 for $\mathbb{F}_{64}$

Let us first briefly depict the meaning of the colors for both figure sides. In all figures, dark colors correspond to small values, and bright colors to high values. So, on the left hand side figures, dark areas mean a decoding failure (small $\overline{p}$), and bright areas mean a successful list decoding. On the right hand side figures, dark areas correspond to a number of coordinate modifications that remains far less than $n - k$, which is the maximum value; bright pixels mean we are close to the maximum. These figures show that, when k is close to n, the code is sufficiently dense in the space to warranty a high value of $\mathbb{E}(Z)$; hence, $\overline{p}$ is close to 1 and the list decoding is successful. Other favorable cases for decoding are for small values of k, where the radius λ_{GS} is close to n, and the decoding balls cover the space quite well. On the contrary, when k is far away from its extremal values (1 and $n-1$), the decoding balls are too small to contain enough codewords, and the decoding fails. Clearly, these figures also show that this behavior increases when q becomes higher. The previous analysis



(a) $\overline{p}$



(b) α

Fig. 6. Same as figure 2 for $\mathbb{F}_{128}$

also explains what we observe on the right hand side figures, since the relative average number of changes α heavily depends on the probability of successful decoding. Remark that the improvement in the embedding efficiency may be significant, compared with the algorithm given in Section 4.1. As an example, for $q = 8$, $GRS(7, 3)$ embeds 4 symbols with up to 3 locked positions and an embedding efficiency improvement up to 37.4% compared with the Lagrange interpolation algorithm. Over $\mathbb{F}_{16}$, $GRS(14, 9)$ embeds 5 symbols with up to 9 locked positions and an embedding efficiency improvement up to 67.6%.

5 Conclusion

We have shown in this paper that Reed-Solomon codes are good candidates for designing realistic efficient steganographic schemes. If we compare them to the previous studied codes, like BCH codes, Reed-Solomon codes improve the management of locked positions during embedding, hence ensuring a better management of the distortion: they are able to lock twice the number of positions, that is, they are optimal in the sense that they enable to lock the maximal number of positions. We proposed two methods for managing these codes in this context: the first one is based on a naive decoding process through Lagrange interpolation; the second one, more efficient, is based on the Guruswami-Sudan list decoding and provides an adaptive trade-off between the number of locked positions and the embedding efficiency.

Acknowledgment

Dr. C. Fontaine is supported (in part) by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT.

We are in debt to Daniel Augot for numerous comments on this work, in particular for pointing out the adaptation of the Guruswami-Sudan algorithm to shortened GRS used in the embedding algorithm.

References

[BW04] Böhme, R., Westfeld, A.: Exploiting preserved statistics for steganalysis. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 82–96. Springer, Heidelberg (2004)

[Cra98] Crandall, R.: Some notes on steganography. Posted on steganography mailing list (1998), `http://os.inf.tu-dresden.de/~westfeld/crandall.pdf`

[FGLS05] Fridrich, J., Goljan, M., Lisonek, P., Soukal, D.: Writing on wet paper. IEEE Transactions on Signal Processing 53(10), 3923–3935 (2005) (special issue "Supplement on Secure Media III")

[FGS05a] Fridrich, J., Goljan, M., Soukal, D.: Efficient wet paper codes. In: Barni, M., Herrera-Joancomartí, J., Katzenbeisser, S., Pérez-González, F. (eds.) IH 2005. LNCS, vol. 3727, pp. 204–218. Springer, Heidelberg (2005)

[FGS05b] Fridrich, J., Goljan, M., Soukal, D.: Perturbed quantization steganography. ACM Multimedia and Security Journal 11(2), 98–107 (2005)

[FGS06] Fridrich, J., Goljan, M., Soukal, D.: Wet paper codes with improved embedding efficiency. IEEE Transactions on Information Security and Forensics 1(1), 102–110 (2006)

[Fra02] Franz, E.: Steganography preserving statistical properties. In: Petitcolas, F.A.P. (ed.) IH 2002. LNCS, vol. 2578, pp. 278–294. Springer, Heidelberg (2003)

[FS06] Fridrich, J., Soukal, D.: Matrix embedding for large payloads. IEEE Transactions on Information Security and Forensics 1(3), 390–394 (2006)

[GK03] Galand, F., Kabatiansky, G.: Information hiding by coverings. In: Proc. ITW 2003, pp. 151–154 (2003)

[GS99] Guruswami, V., Sudan, M.: Improved decoding of Reed-Solomon and algebraic-geometry codes. IEEE Transactions on Information Theory 45(6), 1757–1767 (1999)

[HP03] Huffman, W.C., Pless, V.: Fundamentals of Error-Correcting Codes. Cambridge University Press, Cambridge (2003)

[KDR06] Kim, Y., Duric, Z., Richards, D.: Modified matrix encoding technique for minimal distortion steganography. In: Proc. of the 8th International Workshop on Information Hiding. LNCS, Springer, Heidelberg (2006)

[McE03] McEliece, R.J.: The Guruswami-Sudan decoding algorithm for Reed-Solomon codes. Technical Report 42-153, IPN Progress Report (May 2003) `http://tmo.jpl.nasa.gov/progress_report/42-153/153F.pdf`

[McL84] McLoughlin, A.: The complexity of computing the covering radius of a code. IEEE Transactions on Information Theory 30(6), 800–804 (1984)

[Sim84] Simmons, G.J.: The prisoners' problem and the subliminal channel. In: Advances in Cryptology – CRYPTO 1983, pp. 51–67. Plenum Press (1984)

[Sud97] Sudan, M.: Decoding of Reed-Solomon codes beyond the error-correction bound. Journal of Complexity 13(1), 180–193 (1997)

[SW06] Schönfeld, D., Winkler, A.: Embedding with syndrome coding based on BCH codes. In: Proc. of the ACM Multimedia and Security Workshop 2006, pp. 214–223. ACM Press, New York (2006)

[Var97] Vardy, A.: The intractability of computing the minimum distance of a code. IEEE Transactions on Information Theory 43(6), 1757–1766 (1997)

[Wes01] Westfeld, A.: F5 – a steganographic algorithm: high capacity despite better steganalysis. In: Moskowitz, I.S. (ed.) Information Hiding. LNCS, vol. 2137, pp. 289–302. Springer, Heidelberg (2001)