Approaching Maximum Embedding Efficiency on Small Covers
Using Staircase-Generator Codes

Simona Samardjiska\textsuperscript{1,2} and Danilo Gligoroski\textsuperscript{1}

Department of Telematics, Norwegian University of Science and Technology, Trondheim, Norway\textsuperscript{1}
FCSE, “Ss Cyril and Methodius” University, Skopje, Republic of Macedonia\textsuperscript{2}
\{simonas,danilog\}@item.ntnu.no

Abstract. We introduce a new family of binary linear codes suitable for steganographic matrix embedding. The main characteristic of the codes is the staircase random block structure of the generator matrix. We propose an efficient list decoding algorithm for the codes that finds a close codeword to a given random word. We provide both theoretical analysis of the performance and stability of the decoding algorithm, as well as practical results. Used for matrix embedding, these codes achieve almost the upper theoretical bound of the embedding efficiency for covers in the range of 1000 - 1500 bits, which is at least an order of magnitude smaller than the values reported in related works.

Keywords. Steganography, Matrix embedding, Embedding efficiency, Stego-codes.

1 Introduction

A widely accepted security model for steganographic systems was given in [1]. It is modelled as security in a form of visual and statistical undetectability. How much data can safely be embedded in a cover without being detected is given in [2].

One method for achieving undetectability is matrix embedding. It has been first informally introduced in [3] and more formally in [4] and [5]. It is a steganographic method that uses \((n,k)\) linear binary codes \(C\) to transmit messages of length \(n-k\) bits, embedded into arbitrary covers of \(n\) bits using as small as possible number of changes. Matrix embedding addresses two important design goals in the steganography schemes: 1. To achieve as high ratio as possible between the size of the embedded message and the size of the cover message (so called payload); 2. To achieve as high security as possible in a form of visual and statistical undetectability.

The so called cover in practice can come from different sources such as binary images, textual or binary files, line drawings, three-dimensional models, animation parameters, audio or video files, executable code, integrated circuits and many other sources of digital content [6, Ch.1].

In [5] the theoretical bound that is achievable with matrix embedding was given and it was shown that random linear codes asymptotically achieve the theoretical bound.

1.1 Related Work

Having just a theoretical result that long random linear codes can achieve the embedding capacity, is far from satisfactory in practice. Thus in a series of works, different practical matrix embedding algorithms have been proposed. In [7], two practical schemes for matrix embedding based on random linear codes and simplex codes were proposed. These schemes use relatively small values of \(k \leq 14\), one reason being that the embedding algorithm takes \(O(n2^k)\) operations. Modifications of the schemes from [7] with improved efficiency were proposed in a series of papers such as [8][9][10][11][12]. These schemes, although efficient, do not offer embedding efficiency close to the upper theoretical bound. In [13], the authors propose to use low-density generator matrix (LDGM) codes (defined in [14]) for matrix embedding. Because of the efficient decoding algorithms for LDGM codes, the proposed schemes are quite fast and achieve very good embedding efficiency close to the theoretical bound, but for large \(n\) in the range \(10^5 – 10^6\).

1.2 Our Contribution

The contributions of this paper are severalfold: 1. We define a new family of binary linear codes with a generator matrix \(G\) that has a specifically designed staircase random block structure. 2. We propose...
Definition 1. A steganographic scheme on \( \mathbb{F}_q^n \) with a distortion bound \( R \) is a pair of embedding and extraction mappings \( \text{Emb}: \mathbb{F}_q^n \times \mathcal{M} \rightarrow \mathbb{F}_q^n, \text{Ext}: \mathbb{F}_q^n \rightarrow \mathcal{M} \), such that

\[
\text{Ext}(\text{Emb}(y, m)) = m, \forall y \in \mathbb{F}_q^n, \forall m \in \mathcal{M}, \text{ and } d(y, \text{Emb}(y, m)) \leq R.
\]

The embedding rate, or relative message length, is the value \( \alpha = (n - k)/n \), and the lower embedding efficiency is the value \( e = (n - k)/R \). For the average absolute distortion (average number of changes) \( R_a \), we have the average embedding efficiency \( e_a = (n - k)/R_a \).

Let \( C \) be a binary \((n,k)\) code with a generator matrix \( G \) and a parity check matrix \( H \), both given in a systematic form. We assume that the sender and the recipient share the matrix \( H \) (and thus the matrix \( G \) as well).

Algorithm\[ A \] taken from \[ B \] describes a general matrix embedding scheme for any linear code \( C \).

### Algorithm 1: Matrix embedding

**Emb\((y, m)\):**

1. Set \( z = yH^\top + m \).
2. Let \( c_0 \) be any such that \( c_0H^\top = z \).
3. Find the closest codeword \( xG \) to \( c_0 \) using some efficient algorithm.
4. Set \( e = c_0 + xG \).
5. Embed the message as \( y' = y + e \).

**Output:** A stego object \( y' \).

**Ext\((y')\):**

1. Extract the message as \( m = y'H^\top \).

**Output:** The extracted message \( m \).

Fig. 1. Algorithm for matrix embedding using code \( C \)

The crucial step in Algorithm 1 is Step 3, i.e., finding the closest codeword \( xG \) to \( c_0 \). Thus, the performance of such a scheme is determined by the covering radius \( R \) of the code \( C \) which guarantees
a lower bound on the embedding efficiency, but also on the average distance to the code \( R_a \) which coincides with the average distortion. Therefore, the art of designing a practical steganographic scheme lies in finding codes of small average distance \( R_a \) for which efficient algorithms for finding a codeword within \( R_a \) exist. However, both problems are known to be particularly difficult and challenging.

It is known [7] that when \( n \to \infty \), random codes asymptotically achieve the upper bound for the embedding efficiency \( e \) for a given embedding rate \( \alpha \):

\[
e \leq \frac{\alpha}{H^{-1}(\alpha)}, \quad 0 \leq \alpha \leq 1
\]  

where \( H \) is the binary entropy function.

The authors of [13] report several LDGM codes of dimension \( n = 10000 \) and \( n = 100000 \) with extremely good embedding efficiency. However, there are no known (to the authors’ knowledge) codes, of smaller dimension, close to the bound (1), or to the codes from [13]. Furthermore, to the authors’ knowledge, the codes reported in [13] have currently the best performance regarding embedding efficiency.

3 Staircase-Generator Codes

We consider a binary \((n,k)\) code \( C \) with the following generator matrix in standard form:

\[
G = \begin{pmatrix}
I_k & B_1 & B_2' & \cdots & B_v'
\end{pmatrix}
\]  

Each \( B_i \) is a binary matrix of dimension \( k_i \times n_i \) whose structure will be discussed shortly, and each \( B_i' \) is a random binary matrix of dimension \( \sum_{j=1}^{i-1} k_j \times n_i \), so that \( k = k_1 + k_2 + \cdots + k_v \) and \( n = k + n_1 + n_2 + \cdots + n_v \). Further we set \( K_i = k_1 + \cdots + k_i \) and \( N_i = n_1 + \cdots + n_i \). We will call these codes - Staircase-Generator codes.

Note that a code with generator matrix of the form (2) can be considered as a generalization of at least two well known constructions, taking for example the extended direct sum (EDS) and the amalgamated direct sum (ADS) construction [15]. Indeed, it can be seen that the codes (2) are a generalization of the EDS construction of the codes with generator matrices \( [I_{K_1}B_1], \ldots, [I_{K_v}B_v] \), since the matrices \( B_i' \) are chosen at random. On the other hand, (2) can be seen as an amalgamation of the codes with generator matrices

\[
\begin{pmatrix}
B_1 & B_2' & \cdots & B_{i-1}'
0 & B_2 & \cdots & B_{i-1}'
I_{K_{i-1}} & \cdots & 0
\end{pmatrix}
\]  

and

\[
\begin{pmatrix}
0 & B_2 & \cdots & B_{i-1}'
0 & 0 & \cdots & B_{i-1}'
\end{pmatrix}
\]  

for each \( i \in \{2, \ldots, v\} \).

We should emphasize that, we do not impose any condition on the normality of the codes being amalgamated. In the standard ADS construction, such conditions are necessary in order to prove the improvement on the covering radius. However, in the general case where more than one coordinate is amalgamated, it is much more difficult to theoretically estimate the improvement, and some attempts have not given the desired results [15].

3.1 An Algorithm for Matrix Embedding Using Staircase-Generator Codes

We describe a general list decoding algorithm (Algorithm 2) for the code \( C \), that can be used for finding a codeword close to a given random \( c_0 \). Under the condition that Algorithm 2 is efficient, we immediately
obtain an efficient variant of Algorithm 1 for matrix embedding. Thus, two important questions about Algorithm 2 need to be answered: How efficient it is and what is the expected weight of the obtained e. In order to answer these questions, we discuss several different design choices.

First, we need to fix some notations. Let $G_i$ denote the submatrix of the generator matrix $G$ of size $K_i \times (K_i + N_i)$ consisting of an identity matrix $I_{K_i}$ concatenated with the matrix of the entries from the first $K_i$ rows and the columns $k + 1, \ldots, k + n_i$ of $G$. Further, let $w_0$ denote a small constant that we will refer to as round weight limit.

**Algorithm 2: Decoding**

**Input:** A vector $e_0 \in \mathbb{F}_2^n$, a generator matrix $G$ of the form (2), and a starting weight limit $w_1$.

**Output:** A vector $e \in \mathbb{F}_2^n$ of small weight, such that $xG = e + e_0$, for some $x \in \mathbb{F}_2^n$.

**Procedure:**

Let $x_i$ represent the first $K_i$ bits of the unknown vector $x$. During decoding, we will maintain lists $L_1, L_2, \ldots, L_v$ of triples $(x_i, e_i, b_i)$ where $|x_i| = K_i$, $|e_i| = K_i + N_i$, $|b_i| = K_i + N_i$, that satisfy $\exists \ G_i = e_i + b_{i-1}$, and $wt(e_i) \leq w_i$.

**Step 0:** Let $b_0 = c_0[1 \ldots k_i][k + 1 \ldots k + n_i]$ and $x_0$ and $e_0$ be vectors of dimension 0. Set a starting list $L_0 = ([x_0, e_0, b_0])$.

**Step 1:**

For each $(x_i, e_i, b_i) \in L_{i-1}$, add all $(x_i, e_i, b_i)$ to $L_i$, that satisfy (3) and $wt(e_i) \leq w_0$, where $x_i = x_{i-1} || x_i$, $e_i = e_{i-1} || e_i$ and $x_i$ and $e_i$ are unknown parts of $x_i$, $e_i$. Further, set $b_i = b_{i-1} || x_i || B_i'$.

If $|L_i| < L$ then $w_{i+1} = w_i + 1$, otherwise $w_{i+1} = w_i$.

**Return:** $(x_i, e_i, b_i) \in L_v$ with minimal $wt(e_i)$.

**Fig. 2.** Algorithm for finding a codeword close to a given random word

### 3.2 Choosing the Matrices $B_i$

Note first, that at each step $Step \ i$, effectively, we work with the codes $C_i$ with generator matrices $G_i = [I_{k_i}, B_i]$. In particular, we find all the codewords $x_iG_i$ such that $x_iG_i$ is within a radius $w_i$ of a word $b_{i-1}$. Thus, it is desirable that the codes $C_i$ have as small as possible covering radius. For efficiency reasons of the algorithm, these codes also need to be relatively small. Luckily, for small codes, it is not hard to find exactly the least possible covering radius. The authors of [16] provide a nice classification of the least covering radius of small codes. For our purposes, for $i > 1$, we use small codes of covering radius $R = 1$, and some of the possible choices are given in Table [1]. For $i = 1$, we typically choose the code $C_1$ to be of very high rate. Thus we don’t need to choose it with particular properties (although it is possible), since it is expected that it has low covering radius.

| $(n, k)R$ | matrix $B$ |
|-----------|------------|
| (2, 1)1   | (1)        |
| (3, 1)1   | (1 1)      |
| (3, 2)1   | (1)        |
| (4, 3)1   | (1 1)      |

| $(n, k)R$ | matrix $B$ |
|-----------|------------|
| (5, 3)1   | (1 1 1)    |
| (5, 4)1   | (1 1 1)    |

**Table 1.** Concrete codes $(n, k)R$ of covering radius $R = 1$, with generator matrix $[I_B]$
3.3 Average Distortion Estimates

In order to keep the complexity of Algorithm 2 low, not only need the codes \( G_i \) be small, but also the size of the maintained lists needs to be low as well. Our algorithm chooses the size of the lists to be bounded by some constant \( L \). In this case we can quite accurately estimate the average distortion achievable by Algorithm 2, that we will denote by \( R_{\text{alg}} \). The following theorem provides an estimate of \( R_{\text{alg}} \).

**Theorem 1.** Let \( C \) be a \((n, k)\) code with a generator matrix \( G \) of the form (2). Then, in Algorithm 2 with round weight limit \( w_b \), we have that at each step \( i \), \( 1 \leq i \leq v \),

1. The expected number of vectors \( e_i \in \mathbb{F}_2^{K_i+N_i} \) such that \( wt(e_i) = j \), is \( V_i(j) \) where:
   
   \[
   V_i(j) = \frac{{k_i+n_i \choose j}}{2^{n_i}}, \quad \text{for} \quad 0 \leq j \leq w_i, \\
   V_i(0) = \frac{1}{2^{n_i}}V_{i-1}(0), \quad \text{if} \quad j > w_i \quad \text{and} \\
   V_i(j) = \sum_{\ell=0}^{\min(j,w_i)} \frac{{k_i+n_i \choose \ell}}{2^{n_i}} \cdot V_{i-1}(j-\ell) \quad \text{for} \quad 0 < j \leq w_i.
   
2. The expected size of the list \( L_i \) is \( \sum_{j=0}^{w_i} V_i(j) \).

3. \( R_{\text{alg}} \) is the smallest \( j \) such that \( V_v(j) \geq 1 \).

**Proof.** 1. Since there are \( \binom{k_i+n_i}{j} \) different ways to make exactly \( j \) changes in \( k_i + n_i \) bits, and there are \( 2^k \) codewords in \( C_i \) (the code with generator matrix \([I \ B_i]\)) \(), it can be expected that among all codewords, \( 2^k \binom{k_i+n_i}{j} \frac{1}{2^{n_i}} \) will be at distance exactly \( j \) from a random word in \( \mathbb{F}_2^{k_i+n_i} \).

   Further, if at Step \( i \), \( V_{i-1}(0) \) is the expected number of vectors \( e_{i-1} \) of weight \( 0 \), in the next step, this number reduces to \( 2^k \binom{k_i+n_i}{j} \frac{1}{2^{n_i}} V_{i-1}(0) \).

   The value of \( V_i(j) \) can be obtained as follows. At each Step \( i \) we test each \( e_i \) of weight \( wt(e_i) = \ell \leq w_b \) and each \( e_{i-1} \) in the list \( L_{i-1} \) for consistency with (3). If consistent, \( e_i \) belongs to \( V_i(j) \) only if \( wt(e_{i-1})|e_i| = j \), i.e., only if \( e_{i-1} \) belongs to \( V_{i-1}(j-\ell) \). Again, on average, \( 1/2^{n_i} \) of the tested \( e_i \) satisfy (3). From here, we immediately obtain the claimed value for \( V_i(j) \).

2. Since the list \( L_i \) contains all vectors \( e_i \) of weight \( \leq w_i \), we can estimate its size by taking the sum of all \( V_i(j) \), \( 0 \leq j \leq w_i \).

3. In the last Step \( v \), if \( V_v(j) \geq 1 \), then we can expect that in the list \( L_v \), there is an element \( e'_v \) of weight \( j \). Taking the smallest such \( j \) determines the expected average distortion.

As mentioned in Subsection 3.2 for better results, in practice, we use matrices \( B_i \) as in Table 1 that guarantee that the codes \( C_i, i > 1 \) with generator matrix \([I_k, B_i]\) have covering radius 1. Thus we can restrict the choice of the round weight limit \( w_b \) to small values, typically, 1 or 2. Calculating the average distortion for concrete parameters using Theorem 1 shows that the choice of \( w_b = 2 \) gives better results (cf. Section 4).

3.4 List Estimates

Theorem 1 not only provides the average distortion obtained using Algorithm 2 it also gives the size of the lists \( L_i \) at each step of Algorithm 2. For the efficiency of the algorithm it is important to know at what conditions the lists grow or decrease from one step to another.

Let \( B_i(j) \) denote the ball of radius \( j \) around the zero vector of length \( K_i + N_i \), containing the vectors \( e_i \) with \( wt(e_i) \leq j \) obtained in Algorithm 2. Further, we denote by \( |B_i(j)| \) the expected size of \( B_i(j) \). Then, using the notation from Theorem 1, \( |B_i(j)| = \sum_{s=0}^{j} V_i(s) \), and \( |L_i| = |B_i(w_i)| \). Directly from Theorem 1 we have the following lemma.
Lemma 1. Let $w_b = 2$. Then, when $w_{i+1} = w_i$

$$|L_{i+1}| = |B_{i+1}(w_{i+1})| = \frac{1}{2^{n_{i+1}}} |L_i| + \frac{(k_{i+1} + n_{i+1})}{2^{n_{i+1}}} |B_i(w_{i-1})| + \frac{(k_{i+1} + n_{i+1})}{2^{n_{i+1}}} |B_i(w_i - 2)|,$$

(4)

and when $w_{i+1} = w_i + 1$,

$$|L_{i+1}| = |B_{i+1}(w_{i+1})| = \frac{(k_{i+1} + n_{i+1} + 1)}{2^{n_{i+1}}} |L_i| + \frac{(k_{i+1} + n_{i+1})}{2^{n_{i+1}}} |B_i(w_i - 1)|.$$

(5)

\[\square\]

Before continuing, for clarity of the exposition, we make two simplifications.

First, without loss of generality, we can assume that at a given step $i$ only $|B_i(w_i)|, |B_i(w_i - 1)|, \ldots, |B_i(w_i - t)| > \frac{1}{2^n}$, while $|B_i(0)|, \ldots, |B_i(w_i - t - 1)| \leq \frac{1}{2^n}$ for some integer $s$. Thus for big enough $s$, we will take $|B_i(0)| = 0, \ldots, |B_i(w_i - t - 1)| = 0$. (Indeed, this can be safely assumed for $|B_i(0)|$ since $|B_i(0)| \to 0$ as $i$ grows.) Second, we assume that $k_i = k_2, n_i = n_2$ for all $i \geq 2$. In practice, we will typically choose such parameters.

Proposition 1. Let $w_b = 2$. Let at step $i$, $|B_i(j)| > 0$, for $j \geq w_i - t$, and $|B_i(j)| = 0$, for $j < w_i - t$. Then, if $w_{i+1} = w_i$, we have that:

$$|B_i(w_i - t)| > |B_{i+1}(w_i - t)|,$$

(6)

$$|B_i(w_i - t + 1)| > |B_{i+1}(w_i - t + 1)| \text{ if and only if } \frac{k_2 + n_2}{2^n - 1} |B_i(w_i - t)| < |B_i(w_i - t + 1)|,$$

(7)

$$|B_i(j)| > |B_{i+1}(j)|, w_i - t + 1 < j \leq w_i, \text{ if and only if } \frac{(k_2 + n_2)|B_i(j - 1)| + (k_2 + n_2)|B_i(j - 2)|}{2^n - 1} < |B_i(j)|.$$  

(8)

Proof. The proof is rather straightforward. We have:

$$|B_{i+1}(w_i - t)| = \frac{1}{2^n} |B_i(w_i - t)| < |B_i(w_i - t)|,$$

since $|B_i(w_i - t - 1)| = |B_i(w_i - t - 2)| = 0$ from the condition. Further,

$$|B_{i+1}(w_i - t + 1)| = \frac{|B_i(w_i - t + 1)| + (k_2 + n_2)|B_i(w_i - t)|}{2^n}.$$  

Now, directly, it can be seen that $\frac{k_2 + n_2}{2^n - 1} |B_i(w_i - t)| < |B_i(w_i - t + 1)|$ is equivalent to $|B_{i+1}(w_i - t + 1)| < |B_i(w_i - t + 1)|$.

Very similarly, the last claim follows directly from the expression for $|B_{i+1}(j)|$. \[\square\]

The previous simple proposition implies that if at step $i$, (7) and (8) hold, in the next step $i+1$, for all $j$, $|B_{i+1}(j)| < |B_i(j)|$. When $j = w_i$, this guarantees that at Step $i+1$ the size of the list $L_{i+1}$ will be smaller than in the previous step. However, even if (7) or (8) for some weight $j$ is not satisfied at Step $i$, at a later step $i+s$, if $w_{i+s} = w_i$, an equivalent expressions to (8) (for $i+s$ instead of $i$) will hold.

Proposition 2. Let $w_b = 2$. There exists an integer $s$, s.t. if $w_i = w_i+1 = \cdots = w_i+s$ then $|L_i+s+1| < |L_i+s|$.

Proof. We will show that (8) will eventually be satisfied. From here, the claim will follow immediately.

Suppose $|B_i(w_i - t + 1)| = p_2 \frac{k_2 + n_2}{2^n - 1} |B_i(w_i - t)|$, where $0 < p < 1$. Then $|B_{i+1}(w_i - t)| = \frac{1}{2^n} |B_i(w_i - t)|$, and it can easily be verified that $|B_i(w_i - t + 1)| = \frac{1}{2^n} |B_i(w_i - t)(p + 1 + 2^n)|$, and since $(p + 1 + 2^n) > 1$ for every $n \geq 1$, we get that $|B_{i+1}(w_i - t + 1)| > \frac{k_2 + n_2}{2^n} |B_i(w_i - t)|$.

Next, suppose, (7) and (8) hold for $j < s$, but $|B_i(s)| = p_1 \frac{(k_2 + n_2)|B_i(s-1)| + (k_2 + n_2)|B_i(s-2)|}{2^n - 1}$, where $0 < p_1 < 1$. Similarly, let $|B_{i+1}(s)| = p_2 \frac{(k_2 + n_2)|B_{i+1}(s-1)| + (k_2 + n_2)|B_{i+1}(s-2)|}{2^n - 1}$, for some $p_2$. Then, it can be verified
that \(|B_{i+1}(s)| = \frac{p_1 + 2n_2 - 1}{2^n_2} \cdot \frac{(k_2 + n_2)|B_i(s-1)| + (\frac{k_2 + n_2}{2})|B_i(s-2)|}{2^n_2 - 1}\). From the assumption that \(|B_i(s-1)| > |B_{i+1}(s-1)|, |B_i(s-2)| > |B_{i+1}(s-2)|\), we get that
\[
p_2 > \frac{p_1 + 2n_2 - 1}{2^n_2}, \text{ i.e., } 1 - p_1 > 2^{n_2}(1 - p_2).
\]
This means that either \(p_2 > 1\), in which case either \(\frac{k_2}{2}\) is satisfied for step \(i + 1\), or if \(p_2 \leq 1\), then it is \(2n_2\) times closer to 1 than \(p_1\). Thus if we repeat the process for \(i + 2, i + 3, \ldots\), the sequence \(p_1, p_2, p_3, \ldots\) will either surpass 1 or will exponentially fast approach 1. When \(p_2 < 1\) is very close to 1, (depending on the parameters) we can expect that \(\frac{k_2}{2}\) will be satisfied in the next step. \(\square\)

For the practical parameters that will be given in Section 4, it was observed that \(|L_{i+s+1}| < |L_{i+s}|\) becomes true in just a couple of steps.

The previous discussion was concerned with Steps \(i\), when the weight \(w_i\), does not change throughout the steps. Since in this case, the size of the lists \(|L_i|\) will eventually (after a few steps) start decreasing, at one point the condition \(|L_i| < L\), will be satisfied, and Algorithm 2 will increase the weight to \(w_{i+1} = w_i\). The goal of this step, is to increase the list again. A direct application of Lemma 1 yields:

**Proposition 3.** Let \(w_0 = 2\). Then, when \(w_{i+1} = w_i + 1\),
\[
|L_{i+1}| > |L_i| \text{ if and only if } |B_i| > \frac{2n_2 - k_2 - n_2 - 1}{(k_2 + n_2)^2} |L_i|.
\]
\(\square\)

For the chosen practical parameters from Section 4 the right hand side in (9) will always be negative, and thus increasing the weight \(w_i\) will always increase the size of the list.

### 3.5 Complexity of Algorithm 2

We will first take a look at the first round since it is different from the rest. Filling up the list \(L_1\) would require testing all \(k_1 + n_1\) bit words \(e_i\) of weight at most \(w_i\) for consistency with \(L_1\), and finding the appropriate vectors \(b_1\). The complexity of this part is \(\mathcal{O}((k_1 + n_1)^{w_i} k_1 n_2)\). This immediately implies that \(w_1\) needs to be chosen very small, typically \(\leq 3\).

Next, at each Step \(i\), we consider only \(e_i\) of weight \(wt(e_i) \leq 2\). Testing all the elements from \(L_{i-1}\), would thus take \(|L_{i-1}|(k_2 + n_2)^2\), and forming the list \(L_i\) would take additional \(|L_i|\mathcal{O}((k_1 + (i-1) k_2) n_2) = |L_i|\mathcal{O}(k)\). Since we restrict the size of the lists to \(\mathcal{O}(L)\), we have that the complexity of one round is \(\mathcal{O}(Lk)\). In total there are \(v = \mathcal{O}(n)\) rounds, so the total complexity of the algorithm amounts to \(\mathcal{O}(Lkn)\). Note that this is only a rough estimate of the complexity, without taking into account any implementation optimizations.

### 4 Practical Parameters and Experimental Results

Following the design choices justified in the previous sections, we have formed several codes suitable for practical use. For all codes, \(w_1 = 2\) and \(w_0 = 2\). The other parameters are summarized in Table 2.

| \(n_1\) | \(k_1\) | \(n_2\) | \(k_2\) | \(C_i\) | \(\mathcal{C} (n \approx 1000)\) | \(\mathcal{C} (n \approx 1500)\) |
|---|---|---|---|---|---|---|
| 2 | 14 | 2 | 1 | (3,1) | (1000,343) | (1501,509) |
| 1 | 14 | 1 | 1 | (2,1) | (1001,507) | (1501,757) |
| 2 | 14 | 2 | 3 | (5,3) | (999,603) | (1499,903) |
| 1 | 14 | 1 | 2 | (3,2) | (1002,672) | (1500,1004) |
| 1 | 14 | 1 | 3 | (4,3) | (1003,755) | (1503,1130) |
| 1 | 14 | 1 | 4 | (5,4) | (1000,802) | (1500,1202) |
We have performed an extensive set of experiments to test the performance of our codes in terms of their embedding efficiency $e(\alpha)$. The experiments were performed using an initial implementation in Magma [17]. All the results presented in this section are obtained as an average over 50 experiments for each code.

Figure 3 provides a comparison of the St-Gen codes of length $n \approx 1500$ to other previously known codes: the binary Hamming codes [18], the BDS codes [18], the LDGM codes [13], as well as to two theoretically derived stego-code families (SCF) of Golay codes and LDGM codes [19]. It can be seen from the figure that our codes have approximately the same embedding efficiency as the LDGM codes, with one important difference: The presented performance of the LDGM codes is achieved for much bigger lengths, namely for $n = 10000$.

For lengths $1000 \leq 1500$, the authors of [13] provide information only for $\alpha = 1/2$. Plotted together with our codes in Figure 4, we see that the embedding efficiency of the LDGM codes is quite smaller than that of the St-Gen codes. Our assumption is that the behaviour is similar for other embedding rates as well.

![Figure 3. Comparison of the embedding efficiency of St-Gen codes with other known codes for matrix embedding](image)

![Figure 4. Comparison of the embedding efficiency of St-Gen codes for different lengths of the codes.](image)
We have also tested how well the experimental results fit the theoretical estimates from Theorem 1. Figure 5 shows a small offset between the two, and it remains an open problem to investigate whether the offset has a true significance. The last Figure 6 shows the difference in performance, obtained experimentally, depending on the choice of the round weight limit $w_b$.

5 Conclusions

We introduced a new family of binary linear codes called Staircase-Generated codes, whose structure is suitable for applying list decoding techniques. We proposed an algorithm that searches for a close codeword, and as our theoretical analysis and experiments confirm, it shows very good performance results: Our codes achieve approximately the same embedding efficiency as the currently best codes for matrix embedding, but for lengths at least an order of magnitude smaller.

Having a proof of concept, our future work will be, firstly, directed towards making a fast, optimized implementation, for ex. in C, and applying different techniques for reducing the running time of the algorithm. On the theoretical side, we plan to extend the analysis to estimating the covering radius and the average distance of the codes for different and more general scenarios.

Acknowledgements

We thank the anonymous referees for their comments which helped improve this work. The first author of the paper has been partially supported by FCSE, UKIM, Macedonia and the COINS Research School of Computer and Information Security, Norway.

References

1. C. Cachin, “An information-theoretic model for steganography,” in Information Hiding. Springer, 1998, pp. 306–318.
2. J. J. Harmsen and W. A. Pearlman, “Capacity of steganographic channels,” in Proceedings of the 7th workshop on Multimedia and security. ACM, 2005, pp. 11–24.
3. R. Crandall, “Some notes on steganography,” Steganography Mailing List, Tech. Rep., 1998. [Online]. Available: http://dde.binghamton.edu/download/Crandall_matrix.pdf
4. M. van Dijik and F. Willems, “Embedding information in grayscale images,” in Proceedings of the 22nd Symposium on Information and Communication Theory in the Benelux, Enschede, The Netherlands, 2001, pp. 147–154.
5. F. Galand and G. Kabatiansky, “Information hiding by coverings,” in Information Theory Workshop, 2003. Proceedings, 2003 IEEE. IEEE, 2003, pp. 151–154.
6. I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, Digital Watermarking and Steganography, 2nd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2008.
7. J. Fridrich and D. Soukal, “Matrix embedding for large payloads,” in Electronic Imaging 2006. International Society for Optics and Photonics, 2006, pp. 60 721W–60 721W.
8. Y. Gao, X. Li, T. Zeng, and B. Yang, “Improving embedding efficiency via matrix embedding: a case study,” in Image Processing (ICIP), 2009 16th IEEE International Conference on. IEEE, 2009, pp. 109–112.
9. J. Chen, Y. Zhu, Y. Shen, and W. Zhang, “Efficient matrix embedding based on random linear codes,” in Multimedia Information Networking and Security (MINES), 2010 International Conference on, IEEE, 2010, pp. 879–883.

10. C. Wang, W. Zhang, J. Liu, and N. Yu, “Fast matrix embedding by matrix extending,” Information Forensics and Security, IEEE Transactions on, vol. 7, no. 1, 2012, pp. 346–350.

11. G. Liu, W. Liu, Y. Dai, and S. Lian, “Adaptive steganography based on block complexity and matrix embedding,” Multimedia Systems, vol. 20, no. 2, 2014, pp. 227–238.

12. Q. Mao, “A fast algorithm for matrix embedding steganography,” Digital Signal Processing, vol. 25, 2014, pp. 248–254.

13. J. Fridrich and T. Filler, “Practical methods for minimizing embedding impact in steganography,” in Electronic Imaging 2007. International Society for Optics and Photonics, 2007, pp. 650502–650502.

14. M. J. Wainwright and E. Maneva, “Lossy source encoding via message-passing and decimation over generalized codewords of ldgm codes,” in Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on. IEEE, 2005, pp. 1493–1497.

15. R. L. Graham and N. J. A. Sloane, “On the covering radius of codes,” IEEE Transactions on Information Theory, vol. 31, no. 3, 1985, pp. 385–401.

16. T. S. Baicheva and I. Bouyukliev, “On the least covering radius of binary linear codes of dimension 6,” Adv. in Math. of Comm., vol. 4, no. 3, 2010, pp. 399–404.

17. W. Bosma, J. Cannon, and C. Playoust, “The Magma Algebra System. I. The User Language,” J. Symbolic Comput., vol. 24, no. 3-4, 1997, pp. 235–265, computational algebra and number theory (London, 1993).

18. J. Bierbrauer and J. Fridrich, “Constructing good covering codes for applications in steganography,” T. Data Hiding and Multimedia Security, vol. 3, 2008, pp. 1–22.

19. W. Zhang, X. Zhang, and S. Wang, “Near-optimal codes for information embedding in gray-scale signals,” IEEE Transactions on Information Theory, vol. 56, no. 3, 2010, pp. 1262–1270.