Matrix Embedding for Large Payloads

Jessica Fridrich[a] and David Soukal[b]

[a]Department of Electrical and Computer Engineering;
[b] Department of Computer Science;
SUNY Binghamton, Binghamton, NY 13902-6000, USA

ABSTRACT

Matrix embedding is a general coding method that can be applied to most steganographic schemes to improve their embedding efficiency—the number of message bits embedded per one embedding change. Because smaller number of embedding changes is less likely to disrupt statistic properties of the cover object, schemes that employ matrix embedding generally have better steganographic security. This gain is more important for long messages than for shorter ones because longer messages are easier to detect. Previously introduced approaches to matrix embedding based on Hamming codes are, however, not efficient for long messages. In this paper, we present novel matrix embedding schemes that are efficient for embedding messages close to the embedding capacity. One is based on a family of codes constructed from simplex codes and the second one on random linear codes of small dimension. The embedding efficiency of the proposed methods is evaluated with respect to theoretically achievable bounds.

Keywords: steganography, covering codes, matrix embedding, simplex codes

1. INTRODUCTION

Statistical undetectability is the main requirement for a steganographic scheme. By undetectability, we understand the inability of an attacker to distinguish between stego and cover objects with success rate better than random guessing, given the knowledge of the embedding algorithm and the source of cover media. There are four main factors that influence the steganographic security

1. Type of cover media
2. Method for selection of places within the cover that might be modified
3. The embedding operation
4. The number of embedding changes

If two different embedding schemes share 1)–3), the one that introduces fewer embedding changes will be less detectable because it is less likely to disturb the statistics of the cover to trigger detection.

Matrix embedding improves embedding efficiency—the expected number of random message bits embedded with one embedding change. Matrix embedding was discovered by Crandall[1] in 1998 and analyzed by Bierbrauer.[2] It was also independently re-discovered by Willems et al.[3] and Galand et al.[4] Westfeld[5] was the first one to incorporate matrix embedding in his F5 algorithm. Intuitively, it is clear that the gain in embedding efficiency is larger for short messages than for longer ones. However, improving the embedding efficiency for increasingly shorter messages becomes progressively less important for the overall security because short messages are more difficult to detect than longer ones. Matrix embedding based on binary Hamming codes[5] is, however, far from theoretically achievable bounds for payloads larger than 67% of embedding capacity.

In this paper, we attempt to remedy this situation and propose two coding methods that enable efficient matrix embedding for long messages. The first method uses simplex codes and codes derived from them, while

Further author information:
J.F.: E-mail: fridrich@binghamton.edu, Telephone: +1 607 777 6177

the second method uses codes of small dimension with random generator matrix. Section 2 introduces the necessary basic concepts from coding theory. The embedding mechanism of matrix embedding based on binary Hamming codes is reviewed in Section 3. Relating covering codes with steganography enables us to derive upper bounds on achievable embedding efficiency in Section 4. In Section 5, matrix embedding based on simplex codes and random linear codes with small dimension is explained. The code designs and coding algorithms are supplied with pseudo-codes to ease the code implementation for practitioners. The paper is concluded in Section 6.

2. BASIC CONCEPTS

In this section, we introduce a few elementary concepts from coding theory and some simple facts that will be needed in the rest of the paper. A good introductory text to this subject is, for example, the book by Sloane et al.[6] Throughout the text, boldface symbols denote vectors or matrices while the caligraphiccalligraphic font is reserved for sets.

The space of all n-bit column vectors $\mathbf{x} = (x_1, \dots, x_n)^t$, where $()^t$ denotes the matrix transpose, will be denoted $\mathbb{F}_2^n$. A binary code $\mathcal{C}$ is any subset of $\mathbb{F}_2^n$. The vectors in $\mathcal{C}$ are called codewords. The set $\mathbb{F}_2^n$ is a linear vector space if we define the sum of two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$ and a multiplication of a vector by scalar $a \in \{0, 1\}$ using the usual arithmetics in the finite field $\mathrm{GF}(2) = \{0, 1\}$. Note that in binary arithmetics, sum is the same as difference. The Hamming weight $\mathrm{w}(\mathbf{x})$ of a vector $\mathbf{x}$ is defined as the number of ones in $\mathbf{x}$, i.e., $\mathrm{w}(\mathbf{x}) = x_1 + \cdots + x_n$. The distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ is defined as $d(\mathbf{x}, \mathbf{y}) = \mathrm{w}(\mathbf{x} - \mathbf{y})$. We denote as $B(\mathbf{x}, r)$ the ball with center $\mathbf{x} \in \mathbb{F}_2^n$ and radius r,

$$B(\mathbf{x}, r) = \{\mathbf{y} \in \mathbb{F}_2^n | d(\mathbf{x}, \mathbf{y}) \le r\}.$$

The distance between $\mathbf{x}$ and subset $\mathcal{C} \subset \mathbb{F}_2^n$ is defined as $d(\mathbf{x}, \mathcal{C}) = \min_{\mathbf{c} \in \mathcal{C}} d(\mathbf{x}, \mathbf{c}) = d(\mathbf{c}, \mathbf{c}')$ for some $\mathbf{c}' \in \mathcal{C}$. The covering radius R of $\mathcal{C}$ is defined as

$$R = \max_{\mathbf{x} \in \mathbb{F}_2^n} d(\mathbf{x}, \mathcal{C}).$$

The covering radius is determined by the vector most distant from $\mathcal{C}$. In this text, we will need one more concept called the "average distance to code,"

$$R_a = 2^{-n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} d(\mathbf{x}, \mathcal{C}),$$

which is the average distance between a randomly selected vector from $\mathbb{F}_2^n$ and $\mathcal{C}$. It follows directly from the definitions that $R_a \le R$.

For any subset $\mathcal{C}$ and vector $\mathbf{x}$, $\mathbf{x} + \mathcal{C} = \{\mathbf{y} \in \mathbb{F}_2^n | \mathbf{y} = \mathbf{x} + \mathbf{c},\ \mathbf{c} \in \mathcal{C}\}$. The redundancy r of a code $\mathcal{C}$ is defined as $r = \log_2 \frac{2^n}{|\mathcal{C}|}$, where $|\mathcal{C}|$ is the cardinality of $\mathcal{C}$.

Codes that form a linear vector subspace of $\mathbb{F}_2^n$ are called linear codes. If the vector subspace $\mathcal{C}$ has dimension k, we say that $\mathcal{C}$ is a linear code of length n and dimension k (and codimension $n - k$). We can also say that $\mathcal{C}$ is an $[n, k]$ code. Since there are 2^k codewords in an $[n, k]$ code, the redundancy of a linear code is equal to its codimension $r = n - k$. Each $[n, k]$ code has a basis consisting of k vectors. By writing the basis vectors as rows of an $k \times n$ matrix $\mathbf{G}$, we obtain a generator matrix of $\mathcal{C}$. Each codeword can be written as a unique linear combination of rows from $\mathbf{G}$.

Given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$, their dot product is defined as $\mathbf{x} \cdot \mathbf{y} = x_1 y_1 + \cdots + x_n y_n$, all operations in $\mathrm{GF}(2)$. The vectors $\mathbf{x}$ and $\mathbf{y}$ are orthogonal if $\mathbf{x} \cdot \mathbf{y} = 0$. The orthogonal complement of $\mathcal{C}$ is defined as $\mathcal{C}^\perp = \{\mathbf{x} \in \mathbb{F}_2^n | \mathbf{x} \cdot \mathbf{c} = 0 \text{ for all } \mathbf{c} \in \mathcal{C}\}$, which is an $[n, n-k]$ code. It is called the dual code to $\mathcal{C}$ and its generator matrix $\mathbf{H}$ has $n - k$ rows and n columns. From orthogonality, $\mathbf{H}\mathbf{x} = \mathbf{0}$ for each $\mathbf{x} \in \mathcal{C}$. The matrix $\mathbf{H}$ is called the parity check matrix of $\mathcal{C}$.

For any $\mathbf{x} \in \mathbb{F}_2^n$, the vector $\mathbf{s} = \mathbf{H}\mathbf{x} \in \mathbb{F}_2^n$ is called the syndrome of $\mathbf{x}$. For each syndrome $\mathbf{s} \in \mathbb{F}_2^{n-k}$, the set $\mathcal{C}(\mathbf{s}) = \{\mathbf{x} \in \mathbb{F}_2^n | \mathbf{H}\mathbf{x} = \mathbf{s}\}$ is called a coset. Note that $\mathcal{C}(\mathbf{0}) = \mathcal{C}$. It should be clear that cosets associated with different syndromes are disjoint. From elementary linear algebra, every coset can be written as $\mathcal{C}(\mathbf{s}) = \mathbf{x} + \mathcal{C}$, where $\mathbf{x} \in \mathcal{C}(\mathbf{s})$ is arbitrary. Therefore, there are total of 2^{n-k} disjoint cosets, each consisting of 2^k vectors. Any member of the coset $\mathcal{C}(\mathbf{s})$ with the smallest Hamming weight is called a coset leader and will be denoted as $\mathbf{e}_L(\mathbf{s})$.

The following two simple lemmas will be needed in the text.

LEMMA 2.1. *Given a coset $\mathcal{C}(\mathbf{s})$, for any $\mathbf{x} \in \mathcal{C}(\mathbf{s})$, $d(\mathbf{x}, \mathcal{C}) = w(\mathbf{e}_L(\mathbf{s}))$. Moreover, if $d(\mathbf{x}, \mathcal{C}) = d(\mathbf{x}, \mathbf{c}')$ for some $\mathbf{c}' \in \mathcal{C}$, the vector $\mathbf{x} - \mathbf{c}'$ is a coset leader.*

Proof. $d(\mathbf{x}, \mathcal{C}) = \min_{\mathbf{c} \in \mathcal{C}} \mathrm{w}(\mathbf{x} - \mathbf{c}) = \min_{\mathbf{y} \in \mathcal{C}(\mathbf{s})} \mathrm{w}(\mathbf{y}) = \mathrm{w}(\mathbf{e}_L(\mathbf{s}))$. The second equality follows from the fact that if $\mathbf{c}$ runs through $\mathcal{C}$, $\mathbf{x} - \mathbf{c}$ goes through all members of the coset $\mathcal{C}(\mathbf{s})$. $\square$

LEMMA 2.2. *If $\mathcal{C}$ is $[n, k]$ with an $(n-k) \times n$ parity check matrix $\mathbf{H}$ and covering radius R, then any syndrome $\mathbf{s} \in \mathbb{F}_2^{n-k}$ can be written as a sum of at most R columns of $\mathbf{H}$ and R is the smallest such number. Thus, the covering radius can also be defined as the maximal weight of all coset leaders while the average distance to code is equal to the average weight of coset leaders.*

Proof. Any $\mathbf{x} \in \mathbb{F}_2^n$ belongs to exactly one coset $\mathcal{C}(\mathbf{s})$. We know from Lemma 2.1 that $d(\mathbf{x}, \mathcal{C}) = \mathrm{w}(\mathbf{e}_L(\mathbf{s}))$. But the weight $\mathrm{w}(\mathbf{e}_L(\mathbf{s}))$ is the smallest number of columns in $\mathbf{H}$ that must be added to obtain $\mathbf{s}$. $\square$

LEMMA 2.3. *(Sphere-covering bound) For any code $\mathcal{C} \subset \mathbb{F}_2^n$ with covering radius R*

$$|\mathcal{C}| \geq \frac{2^n}{V(n, R)}, \tag{1}$$

where $V(n, R)$ is the volume of a ball of radius R in $\mathbb{F}_2^n$, $V(n, R) = \sum_{i=0}^{R} \binom{n}{i}$. Moreover, for $R < n/2$,

$$\log_2 V(n, R) \leq nH(R/n), \tag{2}$$

where $H(x) = -x \log_2 x - (1-x) \log_2 (1-x)$ is the binary entropy function.

Proof. Each ball with radius R covers $V(n, R)$ vectors. The balls with centers at codewords cover the whole space but they may have non-empty intersection. Thus, we must have $|\mathcal{C}|V(n, R) \geq 2^n$. The upper bound (2) is a frequently used inequality in coding and its proof is not essential for understanding the rest of this paper. The reader is referred to Lemma 2.4.4 in Ref. 7. $\square$

3. MATRIX EMBEDDING USING BINARY HAMMING CODES

In this section, we describe binary Hamming codes and explain how they can be used for matrix embedding. Binary Hamming codes are $[2^p - 1, 2^p - 1 - p]$ linear codes with parity check matrix $\mathbf{H}$ of dimensions $p \times (2^p - 1)$ whose columns are binary expansions of numbers $1, \ldots, 2^p - 1$. For example, the parity check matrix $\mathbf{H}$ for $p = 3$ is

$$\mathbf{H} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

For any syndrome $\mathbf{s} \in \mathbb{F}_2^p$, let $\mathrm{dec}(\mathbf{s})$ be the integer whose binary expansion is $\mathbf{s}$. It is easy to see that for any non-zero syndrome $\mathbf{s}$, the vector $\mathbf{e}_L(\mathbf{s}) = (0, \ldots, 0, 1, 0, \ldots, 0)^t$ with 1 at the $\mathrm{dec}(\mathbf{s})$-th place is the leader of the coset $\mathcal{C}(\mathbf{s})$ because $\mathbf{H}\mathbf{e}_L(\mathbf{s}) = \mathbf{s}$.

Let us assume that the cover object is an image consisting of N pixels. Most steganographic schemes assign a bit to each possible pixel value, for example, as the LSB of the grayscale value. The embedding then usually proceeds by changing the pixel values to match their assigned bits to the desired message bits. To do so, one might for example flip the LSB of the pixel grayscale value. Assuming the embedded message is a random bit-stream, the probability that each pixel will have to be changed is 1/2. Thus, on average we embed 2 bits per embedding change. We can also say that the scheme has embedding efficiency of 2.

To improve the embedding efficiency using matrix embedding, we divide the cover image into N/n subsets, each consisting of n pixels, where n is the length of an appropriately chosen code. For matrix embedding using the binary Hamming code, $n = 2^p - 1$. We now show that we can embed p message bits in each subset by making at most one embedding change.

Let $\mathbf{x}$ be the bits assigned to a subset of n cover image pixels. To embed p message bits, consider the bits as a syndrome $\mathbf{m} \in \mathbb{F}_2^p$ and replace $\mathbf{x}$ with $\mathbf{y} = \mathbf{x} + \mathbf{e}_L(\mathbf{m} - \mathbf{Hx})$. The receiver extracts the message from $\mathbf{y}$ by calculating $\mathbf{Hy} = \mathbf{Hx} + \mathbf{He}_L = \mathbf{Hx} + \mathbf{m} - \mathbf{Hx} = \mathbf{m}$. The block length n is either shared between the sender and the recipient or communicated in the stego object, for example as in F5.[5]

When $\mathbf{m} - \mathbf{Hx} \neq \mathbf{0}$, only one embedding change is needed because $\mathrm{w}(\mathbf{e}_L(\mathbf{m} - \mathbf{Hx})) = 1$. The changed pixel is the $\mathrm{dec}(\mathbf{m} - \mathbf{Hx})$-th pixel. When $\mathbf{m} - \mathbf{Hx} = \mathbf{0}$ no embedding change is necessary. Thus, assuming $\mathbf{m}$ is a random bit-stream, we make on average $1 - 2^{-p}$ embedding changes per block during embedding. Therefore, the embedding efficiency e_p, defined as the number of random bits embedded per one embedding change is

$$e_p = \frac{p}{1 - 2^{-p}}. \tag{3}$$

Since we are embedding p bits into $2^p - 1$ pixels, the relative message length is $\frac{p}{2^p-1}$. Table 1 shows the relative message length and embedding efficiency for $p = 1, \ldots, 9$.

Table 1. Relative message length and embedding efficiency for matrix embedding using binary Hamming codes $[2^p - 1, 2^p - 1 - p]$.

p	Relative message length	Embedding efficiency
1	1.000	2.000
2	0.667	2.667
3	0.429	3.429
4	0.267	4.267
5	0.161	5.161
6	0.093	6.093
7	0.055	7.055
8	0.031	8.031
9	0.018	9.018

From this table we see that Hamming codes do not improve embedding efficiency for messages whose relative length is above 2/3. We can also see that embedding efficiency increases as the message becomes shorter. For short messages this gain becomes less important because short messages are more difficult to detect anyway. The range where the relative message length is large would benefit the most.

Hamming codes can be used for message lengths larger than 2/3 using a construction called the direct sum.[7] We can divide the message into two or more segments and embed them in disjoint parts of the cover using Hamming codes with different parameters. For instance, a message of relative length 0.8 may be divided into two halves and embed the first half in $0.4 \times n$ pixels and the second half in $0.6 \times n$ pixels. In the first part, we do not use matrix embedding at all and embed with efficiency 2. In the second part, we can use matrix embedding with $p = 2$ because the relative message length is $0.4/0.6 = 2/3$. This will give us embedding efficiency of $0.8/(0.4/2 + 0.4/e_2) = 16/7 \doteq 2.286$, which is larger than 2—the efficiency we would obtain if we embedded without using matrix embedding at all. A markedly better performance can be obtained using the codes described in Section 5.

Before we describe these improved matrix embedding methods, in the next section we derive achievability bounds on how good a performance one can theoretically expect from applying codes to embedding. The efficiency of the proposed methods can then be measured against these bounds.

4. BOUNDS ON EMBEDDING EFFICIENCY

In this section, we derive bounds on the theoretically achievable embedding efficiency of steganographic schemes. We start by reformulating the embedding efficiency in the language of codes. This will enable us to apply the apparatus developed in coding theory to derive the bounds.

An embedding scheme on $\mathbb{F}_2^n$ is a pair of embedding and extraction functions Emb and Ext,

$$\begin{aligned} Emb :& \mathbb{F}_2^n \times \mathcal{M} \to \mathbb{F}_2^n \\ Ext :& \mathbb{F}_2^n \to \mathcal{M}, \end{aligned}$$

such that $Ext(Emb(\mathbf{x}, \mathbf{m})) = \mathbf{m}$ for all $\mathbf{m} \in \mathcal{M}$ and $\mathbf{x} \in \mathbb{F}_2^n$. Here, $\mathcal{M}$ is the set of all messages that can be communicated. Let us further assume that we can embed every message $\mathbf{m} \in \mathcal{M}$ with at most R changes

$$d(\mathbf{x}, Emb(\mathbf{x}, \mathbf{m})) \leq R \text{ for all } \mathbf{m} \in \mathcal{M} \text{ and } \mathbf{x} \in \mathbb{F}_2^n.$$

The value $h(n, R) = \log_2 |\mathcal{M}|$ is called the embedding capacity of the scheme and $\underline{e} = \frac{\log_2 |\mathcal{M}|}{R}$ its lower embedding efficiency. Let R_a be the *expected number* of embedding changes over uniformly distributed covers $\mathbf{x} \in \mathbb{F}_2^n$ and messages $\mathbf{m} \in \mathcal{M}$. The embedding efficiency e is defined as $e = \frac{\log_2 |\mathcal{M}|}{R_a}$. Note that since R is the upper bound on the number of embedding changes, for any embedding scheme $\underline{e} \leq e$.

In Section 3, we saw a matrix embedding scheme with $n = 2^p - 1$, $R = 1$, and $h(n, R) = p$ realized using binary Hamming codes. We now generalize this construction and derive upper bounds on h and e. The two propositions below are due to Galland[4] and also appeared in the unpublished work by Bierbrauer.[2]

Proposition 4.1. *An $[n, k]$ code $\mathcal{C}$ with covering radius R can be used to construct an embedding scheme capable of communicating $n - k$ bits using at most R changes.*

Proof. The proof of this proposition is constructive. In fact, we can simply follow the steps as in embedding using Hamming codes. Let $\mathbf{H}$ be the parity check matrix of the code $\mathcal{C}$. We define the embedding function as $Emb(\mathbf{x}, \mathbf{m}) = \mathbf{x} + \mathbf{e}_L = \mathbf{y}$, where $\mathbf{e}_L$ is a coset leader of the coset $\mathcal{C}(\mathbf{m} - \mathbf{Hx})$ and $\mathbf{m} \in \mathbb{F}_2^{n-k}$ is a segment of $n - k$ message bits. Since $\mathcal{C}$ has covering radius R, we know that $d(\mathbf{x}, \mathbf{y}) = \mathrm{w}(\mathbf{e}_L) \leq R$. The extraction function Ext is defined as $Ext(\mathbf{y}) = \mathbf{Hy} = \mathbf{Hx} + \mathbf{He}_L = \mathbf{Hx} + \mathbf{m} - \mathbf{Hx} = \mathbf{m}$. □

The proof of this proposition also explains why we used the symbols R and R_a to denote the maximal number of embedding changes and the average number of embedding changes. These symbols already have meaning of the covering radius and average distance to code defined in Section 2. For embedding schemes realized using linear codes as in Proposition 4, these concepts coincide. Note that, indeed, the average number of embedding changes for randomly chosen messages (syndromes $\mathbf{m}$) is equal to the average weight of a coset leader.

Proposition 4.2. *Conversely, any embedding scheme on $\mathbb{F}_2^n$ with threshold R and embedding capacity $\log_2 |\mathcal{M}|$ defines a code (not necessarily linear) with covering radius R. Moreover, let $\mathcal{C}^\star$ be the smallest code with radius R and length n. Then, $\log_2 |\mathcal{M}| \leq n - \log_2 |\mathcal{C}^\star|$, which means that the embedding capacity is upper bounded by the redundancy of the smallest code with covering radius R.*

Proof. For each message $\mathbf{m} \in \mathcal{M}$, the set $Ext^{-1}(\mathbf{m})$ is a code with covering radius R. This is, indeed, easy to see because for an arbitrary $\mathbf{x} \in \mathbb{F}_2^n$, $\mathbf{y} = Emb(\mathbf{x}, \mathbf{m}) \in Ext^{-1}(\mathbf{m})$ and $d(\mathbf{x}, \mathbf{y}) \leq R$. To prove the rest of the claim, let $\mathbf{m}_0$ be the message that produces a covering of the smallest cardinality $|Ext^{-1}(\mathbf{m}_0)|$. Because $Ext^{-1}(\mathbf{m})$ are mutually disjoint for different messages, we have $|\mathcal{M}| \leq \frac{2^n}{|Ext^{-1}(\mathbf{m}_0)|} \leq \frac{2^n}{|\mathcal{C}^\star|}$ for the smallest code $\mathcal{C}^\star$. □

After establishing this formal relationship between codes and embedding schemes, we now return to our original problem to obtain bounds for the embedding efficiency. Let $h_{\max}(n, R)$ be the maximal number of bits that can be embedded in an n element cover object (or subset) by making at most R changes. From Proposition 4, we know that $h_{\max}(n, R)$ is upper bounded by the redundancy of the smallest code with covering radius R and length n, i.e., $h_{\max}(n, R) \leq \log_2 \frac{2^n}{|\mathcal{C}^\star|}$. Moreover, let $r_L(n, R)$ be the largest codimension $n - k$ among all $[n, k]$ codes with covering radius R and length n. From Proposition 4, we know that $r_L(n, R) \leq h_{\max}(n, R)$. Thus,

$$r_L(n, R) \leq h_{\max}(n, R) \leq \log_2 \frac{2^n}{|\mathcal{C}^\star|}. \tag{4}$$

We now combine (4), (1), and (2) to derive an upper bound on the maximal relative message length $h_{\max}(n, R)/n$ embeddable using R changes in an n-bit cover object

$$\frac{h_{\max}(n, R)}{n} \leq H(R/n). \tag{5}$$

This inequality enables us to derive an upper bound on the lower embedding efficiency $\underline{e} = \frac{h(n,R)}{R}$ for a given relative message length $\alpha = \frac{h(n,R)}{n}$. From (5)

$$\begin{aligned} \alpha &\leq H(R/n) \\ H^{-1}(\alpha) &\leq R/n \\ \underline{e} = \frac{h(n,R)}{R} &\leq \frac{\alpha}{H^{-1}(\alpha)}. \end{aligned} \tag{6}$$

The same asymptotic upper bound holds for the embedding efficiency e

$$e = \frac{h(n,R)}{R_a} \leq \frac{\alpha}{H^{-1}(\alpha)}. \tag{7}$$

The proof of this can be found the journal version of the paper.[8]

We note that the upper bounds in (5) and (7) are asymptotically achievable using linear codes because the redundancy $r_L(n,R)$ of almost all random linear codes with covering radius R asymptotically achieves $nH(R/n)$ for R/n fixed as $n \to \infty$ (Theorem 12.3.5 in[7]). Thus, there exist asymptotically optimal embedding schemes that are based on linear coverings.

Let us assume that we are interested in matrix embedding schemes on $\mathbb{F}_2^n$ realized using linear codes of length n capable of embedding messages of relative length α (i.e., schemes that embed αn bits in an n-element cover object). Such schemes will be realized using $[n, n(1-\alpha)]$ codes (from Proposition 4). It would be useful to have an upper bound on the embedding efficiency of codes from this class. In other words, we need a lower bound on R_a for $[n, n(1-\alpha)]$ codes.

From Lemma 2.2, the average distance to code R_a can be calculated from the parity check matrix $\mathbf{H}$ by counting how many different syndromes can be obtained by adding the columns of $\mathbf{H}$. Clearly, in the most optimistic case, all $\binom{n}{i}$ possible sums of i columns will lead to different new syndromes for $i = 1, \dots, R$, where R is the covering radius. This is indeed the case for perfect codes, such as the Hamming codes. In general, this will give us a lower bound on R_a.

For any integer n, let R_n be an integer and $0 \leq \xi_n < 1$, a real number, such that

$$\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{R_n - 1} + \xi_n \binom{n}{R_n} = 2^{\alpha n}. \tag{8}$$

We remind that there are $2^{\alpha n}$ syndromes (messages). From (8), we obtain a lower bound on the average number of embedding changes R_a and finally an upper bound on the embedding efficiency e

$$R_a \geq \frac{\sum_{i=1}^{R_n - 1} i\binom{n}{i} + R_n \xi_n \binom{n}{R_n}}{2^{\alpha n}}. \tag{9}$$

$$e = \frac{n\alpha}{R_a} \leq \frac{n\alpha 2^{\alpha n}}{\sum_{i=1}^{R_n - 1} i\binom{n}{i} + R_n \xi_n \binom{n}{R_n}}. \tag{10}$$

5. MATRIX EMBEDDING FOR LARGE PAYLOADS

5.1. Random linear codes

We have seen in Section 3 that random linear codes are asymptotically optimal. Thus, we may attempt to construct good codes randomly. The only problematic part in using random codes is that they lack structure needed to develop fast encoding and decoding algorithms. Fortunately, for large payloads $n-k$ the code dimension k will be small enough to enable coding by exhaustive search.

We first describe the embedding process in words and then give a pseudo-code summarized in Algorithm 1. We remind that the sender has a vector of bits $\mathbf{x} \in \mathbb{F}_2^n$ from the cover object (e.g., LSBs of a subset of pixels) and a parity check matrix $\mathbf{H}$ of dimensions $(n-k) \times n$. Our goal is to change the $n-k$ bit syndrome $\mathbf{Hx}$ to a

Algorithm 1 Matrix embedding using random linear codes of small dimension

```
1. Read the next $n$ bits $\mathbf{x}$ from the cover object (along a pseudo-random path generated from the stego key)
   and read the next message segment $\mathbf{m}$ of length $n-k$.
2. Calculate the syndrome $\mathbf{Hx}$.
3. Find any $\mathbf{e}$ that solves $\mathbf{He} = \mathbf{m} - \mathbf{Hx}$.
4. In the list of all $2^k$ codewords, find the closest codeword to $\mathbf{e}$, denote it $\mathbf{c}(\mathbf{e})$.
5. [Embedding modifications] Modify the cover object so that $\mathbf{y} = \mathbf{x} - \mathbf{e} + \mathbf{c}(\mathbf{e})$.
6. If there are no more message bits to be embedded, stop, otherwise go to 1.
7. [Extraction step] The message bits are extracted by following the same embedding path and calculating
   $n-k$ bits $\mathbf{m}$ from each segment of $n$ bits $\mathbf{y}$ of the stego object $\mathbf{m} = \mathbf{Hy}$.
```

desired message segment $\mathbf{m} = \mathbf{Hy}$, where $\mathbf{y}$ is as close to $\mathbf{x}$ as possible. Because $\mathbf{H}(\mathbf{x}-\mathbf{y}) = \mathbf{Hx} - \mathbf{m}$, $d(\mathbf{x}, \mathbf{y})$ will be minimal if and only if $\mathbf{x} - \mathbf{y} = \mathbf{e}_L(\mathbf{Hx} - \mathbf{m})$, a coset leader of the coset $\mathcal{C}(\mathbf{Hx} - \mathbf{m})$. To find the coset leader, we first find an arbitrary vector $\mathbf{e}$ satisfying $\mathbf{He} = \mathbf{Hx} - \mathbf{m}$. If $\mathbf{c}(\mathbf{e})$ is the closest codeword to $\mathbf{e}$, from Lemma 2.1, we have that $\mathbf{e} - \mathbf{c}(\mathbf{e})$ is a coset leader of $\mathcal{C}(\mathbf{Hx} - \mathbf{m})$.

We note that if $\mathbf{H}$ is generated randomly but already in a systematic form*, finding such a vector $\mathbf{e}$ will be trivial. Therefore, the most time consuming part of encoding is finding the closest codeword $\mathbf{c}(\mathbf{e})$. Because there are total of 2^k codewords of length n, keeping the table of all codewords in memory requires $n2^k$ bits. Finding the closest codeword requires the same order of computations $O(n2^k)$. Due to the exponentially increasing complexity and memory requirements, the code dimension k should be small, e.g., $k \leq 14$. Note that for a fixed k, the relative message length α_n embeddable using $[n, k]$ codes is $\alpha_n = \frac{n-k}{n}$.

We are now ready to describe a pseudo-code for steganographic communication using matrix embedding with random linear codes suitable for large lpayloadsayloads. Let us assume that the sender wants to embed K bits in an N element cover object. This means that the relative message length $\alpha = K/N$. The sender first finds n such that

$$\alpha_{n-1} = \frac{n-1-k}{n-1} \leq \alpha < \frac{n-k}{n} = \alpha_n.$$

The embedding and extraction process will use a random $[n, n(1-\alpha_n)]$ code. Similar to matrix embedding using Hamming codes, the code length n is communicated to the recipient in the stego image itself. This can be arranged in a variety of ways, depending on the steganographic scheme. The sender and the recipient also need to share a set of matrices $\mathbf{H}$ for different values of n or at least a procedure that generates them. As in the original F5 algorithm, the matrices and the parameter k can be a public knowledge. Security should come from a shared secret stego key that determines a pseudo-random order in which the individual elements of the cover object are visited.

In Figure 1, we show the embedding efficiency of random linear codes with dimension $k = 10$ and $k = 14$ for code length $n \leq 165$. A useful practical property of random codes is that they provide an almost continuously changing family of codes with the same coding algorithm, allowing the sender to choose the code length n to closely match $\alpha_n = (n-k)/n$ to the relative payload length and thus efficiently use the available embedding space in the cover object.

In order to see how much the coding improves the embedding efficiency, let us inspect two messages with relative message lengths $\alpha = 0.9$ and 0.8. We can read out the embedding efficiency from Figure 1. For random linear codes of dimension 14, the embedding efficiency improves from 2 (no coding) to approximately 2.7 and 3, respectively. Thus, the coding reduces the impact of embedding the two messages as if we were embedding without any coding messages of relative length $\frac{0.9\times 2}{2.7} = 0.67$ and $\frac{0.8\times 2}{3} = 0.53$, respectively. This appears to be

*$\mathbf{H} = [\mathbf{I}_{n-k}, \mathbf{D}]$, where $\mathbf{I}_{n-k}$ is a square $(n-k)\times(n-k)$ identity matrix.

Table 2. Embedding time for a 1280×1024 grayscale image with block length $n = 100$ and dimension $k = 10, 12$, and 14.

Dimension k	Embedding time in seconds
10	0.82
12	2.42
14	8.65

a substantial improvement because the performance of current steganalyzers for some embedding methods may be quite sensitive to the relative message length in this range.[9, 10]

Also, note that the embedding efficiency of random codes is fairly close to the upper bound (9) on embedding efficiency of $[n, n(1-\alpha)]$ codes of the same length n. We point out that the little "ripples" in the upper bound are not a computing artifact but a real phenomenon.[11]

Figure 1 also demonstrates how the embedding efficiency increases by comparing codes with dimension $k = 10$ and 14. However, the code dimension cannot be increased much without severe complexity increase (recall that the complexity of coding is $O(n2^{n(1-\alpha)})$. To give the reader an idea of the typical embedding speed achievable on a PC, we simulated embedding into an image with $N = 1280 \times 1024$ pixels using a random code with length $n = 100$. We measured the embedding time for code dimensions $k = 10, 12$, and 14. The test was run on Pentium IV running at 3.4 GHz with 1 GB RAM. The algorithm was implemented in C++ and compiled under Linux with GCC 3.4.3. The results are summarized in Table 2.

5.2. Simplex codes

In the previous section, we described an algorithm for matrix embedding using random linear codes suitable for large payloads. In this section, we study a well-known class of structured codes, the $[2^q - 1, q]$ simplex codes. Because simplex codes are duals of Hamming codes, their generator matrix $\mathbf{G}$ is equal to the parity check matrix of binary Hamming codes. For instance, for $q = 3$,

$$\mathbf{G} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}. \tag{11}$$

Simplex codes are constant-weight codes, which means that all non-zero codewords have the same weight, equal to 2^{q-1}, which is also the distance between any two codewords.

There are practical and fast decoders of simplex codes that are quantizers, meaning that for any $\mathbf{x} \in \mathbb{F}_2^{2^q-1}$ the decoding process always finds the closest codeword to $\mathbf{x}$. This means that we can use the *decoding* algorithm for simplex codes to implement the embedding function. Below, we state the decoding algorithm for simplex codes without proof. The proof can be found, for example, in.[6] In the following text, we use function bin : $\{0, 1, \ldots, 2^p - 1\} \to \mathbb{F}_2^p$ which is the inverse function to dec(). It maps an integer i to its binary representation bin(i) written as a column vector with the most significant bit as the first element.

Let $\mathcal{C}$ be a $[2^q - 1, q]$ simplex code with generator matrix $\mathbf{G}$. We denote by $\mathbf{v}_i$ the i-th row of $\mathbf{G}$. Let $\mathbf{x}$ be a noisy codeword (an arbitrary vector $\mathbf{x} \in \mathbb{F}_2^{2^q-1}$) and $\hat{\mathbf{x}} = (0, x_1, x_2, \ldots, x_{2^q-1})^t \in \mathbb{F}_2^{2^q}$ be $\mathbf{x}$ prepended with a zero. Then, the closest codeword $\mathbf{c}(\mathbf{x})$ to $\mathbf{x}$ is $\mathbf{c}(\mathbf{x}) = \sum_{i=1}^{q} u_i \mathbf{v}_i^t$, where $\mathbf{u} = (u_1, \ldots, u_q) = \mathrm{bin}(i_0 - 1)$ and $i_0 = \arg\max_i (\mathbf{1} - 2\hat{\mathbf{x}})^t \mathbf{H}_{2^q}$. We stress that the product is carried out in *regular integer arithmetic* and the arg max is taken over 2^q elements of the vector $(\mathbf{1} - 2\hat{\mathbf{x}})^t \mathbf{H}_{2^q}$. The symbol $\mathbf{1}$ is a column vector of 2^q ones and $\mathbf{H}_{2^q}$ is the Hadamard (Sylvester) matrix of order 2^q. This matrix can be obtained, for example, using the Kronecker product as $\mathbf{H}_{2^q} = \mathbf{H}_2 \otimes \cdots \otimes \mathbf{H}_2$, where there are q Kronecker products of

$$\mathbf{H}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Algorithm 2 Matrix embedding using simplex codes

```
1. Read the next $p = 2^q - 1$ bits $\mathbf{x}$ from the cover object and the next message segment $\mathbf{m}$ of length $2^q - q - 1$ (following a pseudo-random path determined by a stego key).
2. Calculate the syndrome $\mathbf{Hx}$.
3. Find an arbitrary vector $\mathbf{e}$ that solves $\mathbf{He} = \mathbf{m} - \mathbf{Hx}$.
4. Form $\hat{\mathbf{e}} = (0, e_1, \dots, e_{2^q-1})^t$ and calculate $\mathbf{E} = (\mathbf{1} - 2\hat{\mathbf{e}})^t \mathbf{H}_{2^q}$ using the fast Hadamard transform (12)
    (a) $\mathbf{E}^{(0)} = (\mathbf{1} - 2\hat{\mathbf{e}})^t$, for $i = 1$ to $q$, $\mathbf{E}^{(i)} = \mathbf{E}^{(i-1)} \mathbf{M}^{(i)}_{2^q}$.
    (b) $\mathbf{E} = (E_1, \dots, E_{2^q}) = \mathbf{E}^{(q)}$.
    (c) Find the largest number $E_{i_0}$ among $E_1, \dots, E_{2^q}$.
    (d) $\mathbf{u} = \mathrm{bin}(i_0 - 1)$.
    (e) The closest codeword to $\mathbf{e}$ is $\mathbf{c}(\mathbf{e}) = \sum_{i=1}^{q} u_i \mathbf{v}_i^t$, where $\mathbf{v}_i$ is the $i$-th row of $\mathbf{G}$.
    (f) [Embedding modifications] The modified block of the stego object is $\mathbf{y} = \mathbf{x} + \mathbf{e} - \mathbf{c}(\mathbf{e})$.
5. If there are no more message bits to be embedded, stop, otherwise go to 1.
```

Moreover, $\mathbf{H}_{2^q}$, is a square $2^q \times 2^q$ matrix consisting of 1's and -1's whose rows are orthogonal to each other $\mathbf{H}_{2^q}\mathbf{H}^t_{2^q} = 2^q \mathbf{I}_{2^q}$ (orthogonality with respect to the usual dot product in Euclidean space).

The complexity of calculating the product $(\mathbf{1} - 2\hat{\mathbf{x}})^t \mathbf{H}_{2^q}$ is $O(2^{2q})$. Also, keeping the matrix in memory requires the same amount of space. Fortunately, there is a faster algorithm to obtain the product that also requires much less memory. It is known as the fast Hadamard transform.

The transform $\mathbb{Z}^n \to \mathbb{Z}^n$ defined as $\mathbf{x} \mapsto \mathbf{x}^t \mathbf{H}_n$, where $\mathbb{Z}$ the set of all integers, is called the Hadamard transform. The fast transform uses the following fact[6]

$$\mathbf{H}_{2^q} = \mathbf{M}^{(1)}_{2^q} \mathbf{M}^{(2)}_{2^q} \cdots \mathbf{M}^{(q)}_{2^q}, \tag{12}$$

where

$$\mathbf{M}^{(i)}_{2^q} = \mathbf{I}_{2^{q-i}} \otimes \mathbf{H}_2 \otimes \mathbf{I}_{2^{i-1}}. \tag{13}$$

Each $\mathbf{M}^{(i)}_{2^q}$ is sparse with only *two* non-zero elements in each row and column, which translates to drastically reduced memory requirements $O(q2^q)$ for storing all q matrices $\mathbf{M}^{(i)}_{2^q}$. Calculating one single product $\mathbf{x}^t \mathbf{M}^{(i)}_{2^q}$ only takes 3×2^q operations. Thus, evaluating the whole Hadamard transform using (12) and (13) takes $O(q2^q)$ operations, which is significantly better than the direct implementation (which has complexity of $O(2^{2q})$).

We now describe a pseudo-code for matrix embedding using $[2^q - 1, q]$ simplex codes. Let $\mathbf{G}$ be the generator matrix (11) with rows $\mathbf{v}_1, \dots, \mathbf{v}_q$. We will be embedding $2^q - 1 - q$ bits in $2^q - 1$ pixels, which corresponds to relative message length $\alpha_q = \frac{2^q - 1 - q}{2^q - 1}$. Thus, if the sender plans to communicate K bits in an N element cover object, he needs to select the parameter q so that

$$\alpha_{q-1} \le \frac{K}{N} < \alpha_q.$$

Again, the value q should be communicated in the stego object. To finish the embedding process, the sender follows the pseudo-code 2. We note that the parity check matrix $\mathbf{H}$ should be pre-calculated in the systematic form to speed up Step 2 and 3.

The embedding efficiency of simplex codes for $q = 3, \dots, 11$, is shown in Figure 1 using the $\times$ symbol. Note that their performance is not as good as that of random linear codes. Also, they do not cover the range of relative message length α as densely as random codes ($\alpha_q = \frac{2^q - 1 - q}{2^q - 1}$). On the other hand, they easily reach into the range of relative message length $\alpha > 0.95$ and they do so with low computational complexity. In terms of code length $n = 2^q - 1$, the complexity is $O(n \log n)$.

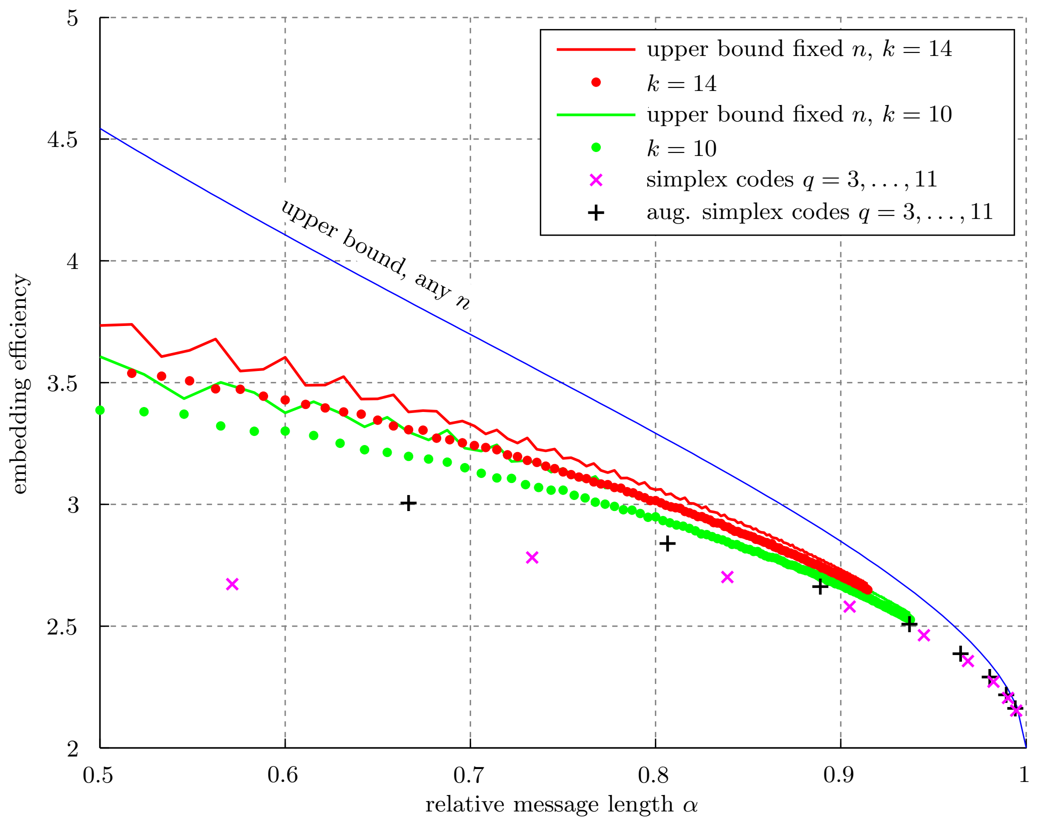

Figure 1. Embedding efficiency vs. relative capacity (large payload case).

5.3. Augmented simplex codes

There are other codes derived from simplex codes using common operations on codes, such as lengthening (increasing length by one) or augmenting (adding a codeword to the generator matrix), that also give good performance and can be decoded using a simple modification of the decoding algorithm for simplex codes. By augmenting the simplex code with an all-one vector $\mathbf{1}$, we obtain a linear $[2^q - 1, q+1]$ code which coincides with the punctured first-order Reed-Muller code.[6] The generator matrix for the augmented simplex code for $q = 3$ is

$$\mathbf{G} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

This code can be decoded using a simple modification of Step 4 of Algorithm 2. Step 4 is now run with e prepended with both '0' and '1': $\hat{\mathbf{e}}_0 = (0, e_1, \ldots, e_{2^q-1})^t$ and $\hat{\mathbf{e}}_1 = (1, e_1, \ldots, e_{2^q-1})^t$, now obtaining two vectors $\mathbf{c}_0$ and $\mathbf{c}_1$ in Step 4e. The vector closer to $\mathbf{e}$ is taken as the output. To avoid calculating the Hadamard transform twice, we note that $(\mathbf{1} - 2\hat{\mathbf{e}}_1)^t \mathbf{H}_{2^q} = (\mathbf{1} - 2\hat{\mathbf{e}}_0)^t \mathbf{H}_{2^q} - 2\mathbf{h}_1$, where $\mathbf{h}_1$ is the first row of $\mathbf{H}_{2^q}$.

The performance of the augmented simplex code is better than for the simplex code (see Figure 1) but, again, not as good as for the random linear codes. The augmented simplex codes also populate the range for α more sparsely with $\alpha_q = \frac{2^q-2-q}{2^q-1}$. On the other hand, both structured codes can easily reach into the range of $\alpha > 0.95$ with low computational complexity.

To give an example of the improvement obtained from these codes, for a message of relative length 0.94, embedding using the augmented simplex code leaves the same impact as an uncoded embedding of a message with relative length $\frac{0.9 \times 2}{2.7} = 0.75$, which is an improvement of about 20%.

6. CONCLUSIONS

Matrix embedding is a previously introduced method for improving embedding efficiency of steganographic schemes. It involves a coding procedure can be applied to most steganographic schemes without any other changes to their embedding mechanism to increase the number of bits embedded using one embedding change. Embedding schemes that impose fewer embedding changes are more secure because they are less likely to disturb the statistics of the cover object to trigger detection.

We proposed two new approaches to matrix embedding that are suitable when the embedded message length is close to the embedding capacity. The first approach is based on random linear codes of small dimension. Random linear codes provide good embedding efficiency that is fairly close to the theoretical upper bound for the class of codes of fixed length. Also, their relative embedding capacity densely covers the range of large payloads, which makes these codes suitable for practical applications.

The second approach to matrix embedding for large payloads proposed in this paper is based on a family of structured codes—the simplex codes. Structured codes are more computationally efficient and can be used even for relative payloads above 0.9. Their performance for shorter payloads is however not as good as for the random codes.

Acknowledgements

The work on this paper was supported by Air Force Research Laboratory, Air Force Material Command, USAF, under the research grants number FA8750-04-1-0112 and F30602-02-2-0093. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation there on. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of Air Force Research Laboratory, or the U.S. Government. Special thanks belong to Jürgen Bierbrauer for directing our attention to simplex codes and to Petr Lisoněk and Miroslav Goljan for many useful discussions.

REFERENCES

1. R. Crandall, "Some notes on steganography." Posted on Steganography Mailing List. `http://os.inf.tu-dresden.de/~westfeld/crandall.pdf`, 1998.
2. J. Bierbrauer, "On Crandall's problem." Personal communication, 1998.
3. M. van Dijk and F. M. J. Willems, "Embedding information in grayscale images," in *Proceedings of the 22nd Symposium on Information and Communication Theory in the Benelux*, pp. 147–154, (Enschede, The Netherlands), May 15–16 2001.
4. F. Galand and G. Kabatiansky, "Information hiding by coverings," in *Proc. ITW2003*, pp. 151–154, (Paris, France), 2003.
5. A. Westfeld, "High capacity despite better steganalysis (F5—A steganographic algorithm)," in *Proceedings, Information Hiding: 4th International Workshop, IHW 2001*, I. S. Moskowitz, ed., *Lecture Notes in Computer Science* **2137**, pp. 289–302, Springer-Verlag, (Pittsburgh, PA, USA), Apr. 25–27 2001.
6. F. J. M. Williams and N. J. Sloane, *The Theory of Error-correcting Codes*, North-Holland, Amsterdam, 1977.
7. G. D. Cohen, I. Honkala, S. Litsyn, and A. Lobstein, *Covering Codes*, vol. 54, Elsevier, North-Holland Mathematical Library, 1997.
8. J. Fridrich and D. Soukal, "Matrix embedding for large payloads," in *submitted to IEEE Transactions on Information Security and Forensics*, 2005.
9. T. Holotyak, J. Fridrich, and S. Voloshynovskiy, "Blind statistical steganalysis of additive steganography using wavelet higher order statistics," in *Submitted to 9th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security*, (Salzburg, Austria), Sep. 19–21 2005.
10. A. Ker, "Resampling and the detection of LSB matching in color bitmaps," in *Security, Steganography and Watermarking of Multimedia Contents VII*, E. J. Delp III and P. W. Wong, eds., *Proceedings of SPIE* **5681**, pp. 1–15, SPIE and IS&T, (San Jose, California, USA), Jan. 16–20 2005.

11. J. Fridrich, M. Goljan, and D. Soukal, “Wet paper codes with improved embedding efficiency,” in *(to appear in) IEEE Transactions on Information Security and Forensics*, 2006.