Wet Paper Codes with Improved Embedding Efficiency

Jessica Fridrich[a], Miroslav Goljan[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

Construction of steganographic schemes in which the sender and the receiver do not share the knowledge about the location of embedding changes requires wet paper codes. Steganography with non-shared selection channels empowers the sender as now he is able to embed secret data by utilizing arbitrary side information, including a high-resolution version of the cover object (perturbed quantization steganography), local properties of the cover (adaptive steganography), and even pure randomness, e.g., coin flipping, for public key steganography. In this paper, we propose a new approach to wet paper codes using random linear codes of small codimension that at the same time improves the embedding efficiency—the number of message bits embedded per embedding change. We describe a practical algorithm, test its performance experimentally, and compare the results to theoretically achievable bounds. We point out an interesting ripple phenomenon that should be taken into account by practitioners. The proposed coding method can be modularly combined with most steganographic schemes to allow them to use non-shared selection channels and, at the same time, improve their security by decreasing the number of embedding changes.

Keywords: Steganography, covering codes, embedding efficiency, wet paper codes, matrix embedding, selection channel

1. INTRODUCTION

The detectability of hidden data in a stego object is mainly influenced by four basic factors: 1) the cover object, 2) the selection rule that is used to choose individual elements of the cover that might be modified during embedding, 3) the character of the embedding operation that modifies the cover elements, and 4) the number of embedding changes (proportional to the message length).

A selection channel is the process of choosing elements of the cover image that will by used for embedding. In adaptive steganography, the selection channel is typically constrained to areas of the cover image that are highly textured and avoids flat and homogeneous regions. This approach is motivated by the idea that flat and less textured areas of the image may be better modeled by the attacker than areas with high activity; therefore restricting the embedding to textured complex regions of the image will improve security. There is a fundamental problem that every adaptive embedding scheme must solve: the act of embedding modifies the cover image, which may disturb the receiver's ability to read to message. This problem can be solved by calculating the selection channel from certain quantities that are invariant to embedding (e.g., the seven most significant bits in LSB steganography). Recently, some researchers[1] have begun to question whether this paradigm indeed improves security. If the selection channel is public or "weakly" public the attacker may use this knowledge to calibrate his model on parts of the image where no embedding took place, or he may narrow down his attention to regions of the image with higher embedding density. To prevent the attacker from doing so, the selection channel should not be publicly available even in any "partial form." One possible remedy is to select it based on some side information that is *in principle* unavailable to the attacker (e.g., purely random) or that cannot be well estimated from the stego image itself, for example a high resolution (unquantized) version of the cover object.[2]

Further author information:
J.F.: E-mail: fridrich@binghamton.edu, Telephone: +1 607 777 6177
M.G.: E-mail: mgoljan@binghamton.edu, Telephone: +1 607 777 5793

The information-theoretic model for non-shared selection channels is the memory with defective cells.[3] This channel is also known in steganography as writing on wet paper.[4] Practical wet paper codes were described in Ref 5.

Given two embedding schemes that share the same source of cover images, selection channel, and embedding operation, the one that incurs a smaller number of embedding changes will be less detectable as it decreases the chance that any statistics used by an attacker will be disturbed enough to mount a successful steganalysis attack. Thus, it is important to develop techniques that allow embedding data while making as few changes to the cover image as possible.

The problem of minimizing the number of embedding changes can be formulated in terms of covering codes (or a more common term used in steganography—matrix embedding). This fact was discovered by Crandall in 1998,[6] later analyzed in an unpublished article by Bierbrauer,[7] and recently independently rediscovered by Galand et al.[8]

In this paper, we provide a new tool for steganography—a coding method that empowers the steganographer with the ability to use *arbitrary* selection channels while substantially decreasing the number of embedding changes, assuming the embedded message length is shorter than 70% of the embedding capacity. This method is general and flexible and can be easily incorporated as a module into majority of existing steganographic methods.

The paper is organized as follows. In Section 2, we review a few basic concepts from coding theory that will be needed in the rest of the paper. A previously proposed approach to wet paper codes based on syndrome coding using random linear codes is briefly described in Section 3. In the same section, we derive theoretical bounds on achievable embedding efficiency for linear codes. Section 4 explains the proposed coding method in detail. Its embedding efficiency is calculated in Section 5. Experimental results, their analysis, and interpretation appear in Section 6. In Section 7, we discuss an application of the proposed technique for data embedding in binary images. Finally, the paper is concluded in Section 8.

2. BASIC CONCEPTS OF CODING THEORY

In this section, we review some elementary concepts from coding theory that are relevant for our study. An excellent introductory text for this subject is, for example, Ref. 9. We employ the following notation in the rest of the paper: boldface symbols denote vectors or matrices and the calligraphic font is used for sets.

A binary code $\mathcal{C}$ is any non-empty subset of the space of all n-bit column vectors $\mathbf{x} = (x_1, \dots, x_n)^t \in \{0,1\}^n$. The vectors in $\mathcal{C}$ are called codewords. The set $\{0,1\}^n$ endowed with the operations of addition of two vectors and multiplication of a vector by a scalar from $\{0,1\}$, defined using the usual arithmetics in the finite field GF(2), forms a linear vector space. We will denote the field GF(2) by $\mathbb{F}_2$. For any sets $\mathcal{C}, \mathcal{D} \subset \mathbb{F}_2^n$ and any vector $\mathbf{x}$, we define $\mathcal{C} + \mathcal{D} = \{\mathbf{y} \in \mathbb{F}_2^n \,|\, \mathbf{y} = \mathbf{c} + \mathbf{d},\, \mathbf{c} \in \mathcal{C},\, \mathbf{d} \in \mathcal{D}\}$, $\mathbf{x} + \mathcal{C} = \{\mathbf{y} \in \mathbb{F}_2^n \,|\, \mathbf{y} = \mathbf{x} + \mathbf{c},\, \mathbf{c} \in \mathcal{C}\}$.

The Hamming weight w of a vector $\mathbf{x}$ is defined as the number of ones in $\mathbf{x}$. The distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ is the Hamming weight of their difference $d(\mathbf{x},\mathbf{y}) = w(\mathbf{x}-\mathbf{y})$. For any $\mathbf{x} \in \mathcal{C}$ and a positive real number r, we denote as $\mathcal{B}(\mathbf{x}, r)$ the ball with center $\mathbf{x}$ and radius r, $\mathcal{B}(\mathbf{x}, r) = \{\mathbf{y} \in \mathbb{F}_2^n | d(\mathbf{x},\mathbf{y}) \le r\}$. We further define the distance between $\mathbf{x}$ and set $\mathcal{C} \subset \mathbb{F}_2^n$ as the distance of the closest vector from $\mathcal{C}$ to $\mathbf{x}$, formally $d(\mathbf{x},\mathcal{C}) = \min_{\mathbf{c}\in\mathcal{C}} d(\mathbf{x},\mathbf{c})$. The covering radius R of $\mathcal{C}$ is defined as $R = \max_{\mathbf{x}\in\mathbb{F}_2^n} d(\mathbf{x},\mathcal{C})$, which is the distance of the most distant vector of the space $\mathbb{F}_2^n$ from the code $\mathcal{C}$. The average distance to code $\mathcal{C}$, defined as $R_a = 2^{-n}\sum_{\mathbf{x}\in\mathbb{F}_2^n} d(\mathbf{x},\mathcal{C})$, is the average distance between a randomly selected vector from $\mathbb{F}_2^n$ and the code $\mathcal{C}$. Clearly, $R_a \le R$.

A code $\mathcal{C}$ is linear if it is a vector subspace of $\mathbb{F}_2^n$. If $\mathcal{C}$ has dimension k, we call $\mathcal{C}$ a linear code of length n and dimension k (and codimension or redundancy $n-k$), we will also say that $\mathcal{C}$ is an $[n,k]$ code. A linear code $\mathcal{C}$ of dimension k has a basis consisting of k vectors. Writing the basis vectors as rows of a $k \times n$ matrix $\mathbf{G}$, we obtain a generator matrix of $\mathcal{C}$. Each codeword can be then written as a linear combination of rows from $\mathbf{G}$. There are 2^k codewords in an $[n,k]$ code.

Given $\mathbf{x},\mathbf{y} \in \mathbb{F}_2^n$, we define their dot product $\mathbf{x}\cdot\mathbf{y} = x_1y_1 + \cdots + x_ny_n$, all operations in GF(2). If $\mathbf{x}\cdot\mathbf{y} = 0$, we say that $\mathbf{x}$ and $\mathbf{y}$ are orthogonal. Given a code $\mathcal{C}$, the dual code of $\mathcal{C}$, denoted as $\mathcal{C}^\perp$, is the set of all vectors $\mathbf{x}$ that are orthogonal to all vectors in $\mathcal{C}$. The dual code of an $[n,k]$ code is an $[n,n-k]$ code and any of its

$(n-k) \times n$ generator matrices $\mathbf{H}$ has the property that $\mathbf{Hx} = \mathbf{0}$ for each $\mathbf{x} \in \mathcal{C}$. The matrix $\mathbf{H}$ is called the parity check matrix of $\mathcal{C}$. Thus a parity check matrix of a code $\mathcal{C}$ is a generator matrix of $\mathcal{C}^\perp$ and vice versa, which explains the term "dual."

For any $\mathbf{x} \in \mathbb{F}_2^n$, the vector $\mathbf{s} = \mathbf{Hx} \in \mathbb{F}_2^{n-k}$ is called the syndrome of $\mathbf{x}$. For each syndrome $\mathbf{s} \in \mathbb{F}_2^{n-k}$, the set $\mathcal{C}(\mathbf{s}) = \{\mathbf{x} \in \mathbb{F}_2^n | \mathbf{Hx} = \mathbf{s}\}$ is called a coset, note that $\mathcal{C}(\mathbf{0}) = \mathcal{C}$. Cosets associated with different syndromes are disjoint. Also, from elementary linear algebra we know that every coset can be written as $\mathcal{C}(\mathbf{s}) = \mathbf{x} + \mathcal{C}$, where $\mathbf{x} \in \mathcal{C}(\mathbf{s})$ arbitrary. Therefore, there are 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 $\mathbf{e}_L(\mathbf{s})$.

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

LEMMA 2.2. *If $\mathcal{C}$ is an $[n, k]$ code with a $(n-k) \times n$ 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, we can also define the covering radius as the maximal weight of all coset leaders.*

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

3. WET PAPER CODES AND COVERING CODES

In the following text, we for simplicity assume that the cover image $\mathbf{x}$ consists of n pixels x_i, $x_i \in \{0, 1, \ldots, 255\}$. The sender selects k changeable pixels x_j, $j \in \mathcal{J} \subset \{1, 2, \ldots, n\}$, $|\mathcal{J}| = k$, which is the selection channel. The sender can modify the changeable pixels at will in order to communicate a secret message to the recipient. The remaining (non-changeable) pixels cannot be modified during embedding. We repeat that the selection channel is not shared with the recipient.

We further assume that the communicating parties agree on a mapping $b : \{0, 1, \ldots, 255\} \to \{0, 1\}$. For example, they can use $b(x) =$ the LSB of x (Least Significant Bit). During embedding, the sender either leaves the changeable pixels x_j, $j \in \mathcal{J}$, unmodified or replaces x_j with y_j in order to modify its bit from $b(x_j)$ to $b(y_j)$. After embedding, the vector of cover image bits $\mathbf{b_x} = (b(x_1), \ldots, b(x_n))^t$ changes to $\mathbf{b_y} = (b(y_1), \ldots, b(y_n))^t$, where $\mathbf{x}^t$ denotes the transpose of $\mathbf{x}$. In order to communicate m bits $\mathbf{m} \in \mathbb{F}_2^m$, the sender modifies some changeable pixels x_j, $j \in \mathcal{J}$, so that

$$\mathbf{Db_y} = \mathbf{m}, \tag{1}$$

where $\mathbf{D}$ is an $m \times n$ binary matrix shared by the sender and the recipient. The equation (1) can be further rewritten to

$$\mathbf{Dv} = \mathbf{m} - \mathbf{Db_x} \tag{2}$$

using the variable $\mathbf{v} = \mathbf{b_y} - \mathbf{b_x}$ with non-zero elements corresponding to the pixels that must be changed in order to satisfy (1). In (2), there are k unknowns v_j, $j \in \mathcal{J}$, while the remaining $n-k$ values v_i, $i \notin \mathcal{J}$, are zeros. The sender can therefore remove from $\mathbf{D}$ all $n-k$ columns $\mathbf{d}_i$, $i \notin \mathcal{J}$, and also remove from $\mathbf{v}$ all $n-k$ elements v_i with $i \notin \mathcal{J}$. Keeping the same symbol for $\mathbf{v}$, (2) now becomes

$$\mathbf{Hv} = \mathbf{s}, \tag{3}$$

where $\mathbf{H}$ is an $m \times k$ matrix consisting of those columns of $\mathbf{D}$ corresponding to indices $\mathcal{J}$, $\mathbf{v}$ is an unknown $k \times 1$ vector, and $\mathbf{s} = \mathbf{m} - \mathbf{Db_x}$ is the $m \times 1$ right hand side. Thus, the sender needs to solve a system of m linear equations with k unknowns in GF(2). The problem of solvability of (3) has been investigated in great detail in Ref. 4 and is briefly reviewed below.

Using the terminology of coding theory, interpreting $\mathbf{H}$ as a parity check matrix of some $[k, k-m]$ linear code, solving (3) amounts to decoding the noisy codeword $\mathbf{v}$ from its syndrome $\mathbf{s}$. The minimization of the number of embedding changes then equals to finding such a solution $\mathbf{v}$ to (3) (possibly out of many) with the minimal weight—a coset leader.

The matrix $\mathbf{H}$ is obtained from $\mathbf{D}$ as a column sub-matrix as defined by the selection channel. Since the selection channel can be arbitrary, e.g., even random or dependent on the cover, it is difficult to impose structure on $\mathbf{D}$ that would carry over to $\mathbf{H}$ and that would help us solve (3). Furthermore, we need a whole class of good codes for various values of n, k, and m.

The results from Ref. 4 show that random linear codes asymptotically enable communication of k bits and that with increasing code length n they also achieve the best possible embedding efficiency for a fixed relative message length $\alpha = m/k$ and fixed rate $r = k/n$. This makes these codes ideal for steganographic applications provided there exist efficient coding algorithms (Section 4.1). Assuming that the sender will always try to embed as many bits as possible by adding rows to $\mathbf{D}$ while (3) still has a solution then for random binary matrices whose elements are i.i.d. realizations of a random variable uniformly distributed in $\{0, 1\}$, the expected value of the maximum message length $m_{\max}$ that can be communicated in this manner is[4]

$$m_{\max} = k + O(2^{-k/4}) \tag{4}$$

as $k \to \infty$, $k < n$. Therefore, these variable-rate random linear codes asymptotically achieve the maximal embedding capacity.

We now address the embedding efficiency of the syndrome coding approach above. Let R be the covering radius of the linear code with parity check matrix $\mathbf{H}$. Lemma 2.2 tells us that every syndrome can be generated by adding at most R columns, where R is the covering radius. Since there are $\binom{k}{i}$ different sums of i columns of $\mathbf{H}$, we obtain the sphere-covering bound

$$2^m \leq \sum_{i=0}^{R} \binom{k}{i} = V(k, R) \leq 2^{kH(R/k)}, \tag{5}$$

where $V(k, R)$ is the volume of a ball of radius R in $\mathbb{F}_2^k$ and $H(x) = -x \log_2 x - (1 - x) \log_2(1 - x)$ is the binary entropy function. The second inequality is a frequently used bound in coding (e.g., Lemma 2.4.3 in Ref. 10).

Since we are embedding m bits using k changeable pixels, the relative message length is $\alpha = m/k$. We define the lower embedding efficiency $\underline{e}$ as the ratio $\underline{e} = m/R$, which is the number of embedded bits per embedding change in the worst case when we have to make all R changes. For practical purposes, however, steganographers may be more interested in the average case rather than the worst case. Therefore, we further define the embedding efficiency as $e = m/R_a$, where R_a is the average distance to code with parity check matrix $\mathbf{H}$. It is obvious that $\underline{e} \leq e$, which is the reason why $\underline{e}$ is called the lower embedding efficiency. The inequality (5) enables us to derive an upper bound on $\underline{e}$ and eventually on e. From (5),

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

where $H^{-1}(x)$ is the inverse of $H(x)$ on $[0, 1/2]$.

Thus, we obtained an upper bound on the lower embedding efficiency for a fixed relative message length α. It is possible to show[11] that $\frac{\alpha}{H^{-1}(\alpha)}$ is also an asymptotic upper bound on e. Furthermore, it is known that the upper bound is asymptotically achievable using almost all random linear codes $[k, k - \alpha k]$ with $k \to \infty$ (see Theorem 12.3.5. in Ref. 10, page 325).

We conclude this section by remarking that random linear codes are good candidates for wet paper codes with minimal number of embedding changes. We introduce a practical embedding method and study its performance in the following section.

4. MATRIX EMBEDDING WITH WET PAPER CODES

The sender's goal is to find a solution to (3), $\mathbf{Hv} = \mathbf{s}$, with the smallest weight $w(\mathbf{v})$. This problem can be viewed equivalently as finding a coset leader of the coset $\mathcal{C}(\mathbf{s})$. This problem is, in general, an NP complete problem.[12] One often used approach to overcome the complexity issue is to use structured codes. In our situation, however, we cannot use this approach because $\mathbf{H}$ is obtained from $\mathbf{D}$ through a selection process over which the sender may not have control. On the other hand, it is possible to impose some stochastic structure on $\mathbf{D}$ that would be transferred to $\mathbf{H}$, for example we may use specific distribution of weights of columns. This is the basic idea used in the implementation of efficient wet paper codes in Ref. 5. The column weights of $\mathbf{D}$ were required to follow the robust soliton distribution from LT codes.[13] While these low density parity check matrices work very well and can be used to quickly solve (3), attempts to modify the process of decoding the LT codes to improve the embedding efficiency were only moderately successful (see Section 4.2 in Ref. 5). Still, it is possible that some other stochastic properties may be imposed on $\mathbf{D}$ that would allow us find efficiently solutions of (3) with small weight. This topic will be the subject of our future efforts.

Another frequently used paradigm in situations when facing an NP complete problem is to use brute force. This approach can be effective as long as the search space is managablemanageable. In our situation, an obvious simple measure is to use codes of small codimension where an exhaustive search is computationally feasible. However, we must answer the question of how much is lost on optimality of coding as the results in Section 3 are only asymptotic.

4.1. Random linear codes of small codimension

Let us assume that the cover image has n pixels and k changeable pixels and that we wish to communicate m message bits. The sender and receiver agree on a small integer p (e.g., $p < 20$) and using the stego key divide the cover image into $n_B = m/p$ disjoint pseudo-random blocks of cardinality $n/n_B = pn/m$ (for simplicity we assume the quantities above are all integers). Each block will contain on average $k/n \times pn/m = pk/m = p/\alpha$ changeable pixels, where $\alpha = m/k$, $0 \le \alpha \le 1$, is the relative message length. The sender will use a pseudo-random binary $p \times pn/m$ matrix $\mathbf{D}$ for embedding up to p bits. The matrix $\mathbf{D}$ can be the same for each block, publicly available, or also generated from a secret stego key. Note that since duplicates and zero columns in $\mathbf{D}$ do not help, as long as* $n/n_B = pn/m < 2^p$, we can generate $\mathbf{D}$ so that its columns are non-zero and mutually different.

As described in Section 3, in each block the sender forms a binary sub-matrix $\mathbf{H}$ of $\mathbf{D}$ and the syndrome $\mathbf{s}$ from the set of all changeable pixels in that block. The matrix $\mathbf{H}$ will have exactly p rows and, on average, p/α columns. Let $\mathcal{C}_1 \subset \mathbb{F}_2^p$ be the set of all columns of $\mathbf{H}$, and $\mathcal{C}_{i+1} = \mathcal{C}_1 + \mathcal{C}_i - (\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_i) - \{\mathbf{0}\}$, for $i = 1, \ldots, p$. Note that $\mathcal{C}_i = \emptyset$ for $i > R$, where R is the covering radius of $\mathbf{H}$. Also note that $\mathcal{C}_i$ is the set of syndromes that can be obtained by adding i columns of $\mathbf{H}$ but no less than i (equivalently, $\mathcal{C}_i$ is the set of all coset leaders of weight i).

Let $\mathbf{s} = \mathbf{h}_{j_1} + \cdots + \mathbf{h}_{j_r}$, where r is the minimal number of columns of $\mathbf{H}$ adding up to $\mathbf{s}$, $r \le R$. Then, $\mathbf{s} + \mathbf{h}_{j_1} + \cdots + \mathbf{h}_{j_{\lfloor r/2 \rfloor}} = \mathbf{h}_{j_{\lfloor r/2 \rfloor + 1}} + \cdots + \mathbf{h}_{j_r}$, which implies $(\mathbf{s} + \mathcal{C}_{\lfloor r/2 \rfloor}) \cap \mathcal{C}_{r - \lfloor r/2 \rfloor} \neq \emptyset$ and $\mathbf{v}$ with zeros everywhere except for indices $j_1, \ldots, j_r$ solves (3). This leads to the Algorithm 1 for finding the coset leader.

After the solution $\mathbf{v}$ is found, the sender modifies the pixels in the block accordingly—the non-zero elements of $\mathbf{v}$ determine pixels x_i within the block where embedding changes must take place. The modified block of pixels in the stego image is denoted $\mathbf{y}$.

The extraction of the message is very simple: the receiver knows n from the stego image and knows p because it is an agreed-upon (or publicly shared) parameter. The message length m is used in dividing the image into blocks, therefore it needs to be communicated in the stego image as well. This can be arranged in many different ways, for example, by isolating a small subset (using the stego key) from the image and embedding $\log_2 m$ bits in it using standard wet paper code from Section 3. Knowing m, the recipient uses the secret stego key and partitions the rest of the stego image into the same disjoint blocks as the sender and extracts p message bits $\mathbf{m}$ from each block of pixels $\mathbf{y}$ as $\mathbf{m} = \mathbf{Dy}$.

*This will be satisfied for embedding in typical digital media files because we use $p \approx 20$ (see below).

Algorithm 1 Meet-in-the-middle algorithm for finding coset leaders

if $\mathbf{s} \in \mathcal{C}_1$ **then**
  $v_{j_1} \leftarrow 1$
  for all $j \neq j_1$ **do** $v_j \leftarrow 0$
  return ▷ *the solution is one of the original columns, we are done*
end if
$l \leftarrow r \leftarrow 1$
while $(\mathbf{s} + \mathcal{C}_l) \cap \mathcal{C}_r = \emptyset$ **do**
  if $l = r$ **then**
    $r \leftarrow r + 1$
    if $\mathcal{C}_r$ not constructed **then** construct $\mathcal{C}_r$
  else
    $l \leftarrow l + 1$
    if $\mathcal{C}_l$ not constructed **then** construct $\mathcal{C}_l$
  end if
end while
▷ *there is a solution* $\mathbf{v}$ *of weight* $l + r$ *determined by any vector from the intersection*

4.2. Complexity

The Algorithm 1 will, in the worst case, need to calculate all sets $\mathcal{C}_1, \ldots, \mathcal{C}_{\lceil R/2 \rceil}$. The cardinalities of $\mathcal{C}_i$ increase with i, achieve a maximum for $i \approx R_a$, and then quickly fall off to zero for $i > R_a$. The covering radius R_a asymptotically approaches R with increasing length of the code (or increasing p). This means that the algorithm avoids computation of the largest of the sets $\mathcal{C}_i$.

The space complexity of the algorithm is driven by the need to keep the sets $\mathcal{C}_i, i = 1, \ldots, \lceil R/2 \rceil$ and the indices $j_1, \ldots, j_i$ for each element of $\mathcal{C}_i$ in memory. The average value of $\mathcal{C}_i$ is upper bounded by $\binom{p/\alpha}{i}$ because on average $|\mathcal{C}_1| = p/\alpha$. This means that the total memory requirements are bounded by $O(R/2 \cdot \binom{p/\alpha}{R/2}) \approx O(p \cdot 2^{p/\alpha H(R\alpha/2p)}) \approx O(p2^{\beta p})$, where $\beta = \frac{H(H^{-1}(\alpha)/2)}{\alpha} < 1$, because $R \approx p/\alpha H^{-1}(\alpha)$ for large p from (6). For example, for $\alpha = 1/2, \beta = 0.61$. To obtain a bound on the computational complexity, note that we need to compute $\mathcal{C}_1 + \mathcal{C}_i$ for $i = 1, \ldots, R/2$. Thus, the computational complexity is bounded by $O(R/2 \cdot p/\alpha \cdot \binom{p/\alpha}{R/2}) \approx O(p^2 2^{\beta p})$.

We also studied other approaches for finding coset leaders, such as the method based on non-expurgated syndrome trellis proposed by Wadayama.[14] However, because the computational complexity of Wadayama's method is $O(p2^p)$, it is asymptotically slower than the meet-in-the-middle method.

4.3. Implementation issues

In this subsection, we now discuss the solvability of (3). The equation $\mathbf{Hv} = \mathbf{s}$ will have a solution for all $\mathbf{s} \in \mathbb{F}_2^p$ if and only if $\text{rank}(\mathbf{H}) = p$. The probability of this is $1 - O(2^{p(1-k/m)})$, because this is the probability that a random binary matrix with dimension $p \times p/\alpha$, $\alpha = m/k$, will have full rank.[15] We see, that this probability quickly approaches 1 as we decrease message length m or increase p (for fixed m and k) because $k > m$.

When k/m is close to 1 ($m \sim k$), the probability that $\text{rank}(\mathbf{H}) < p$ may become large enough to encounter a failure to embed all p bits in some blocks. For example, for $p = 18$ and $k/m = 2$, $n = 10^6$, $k = 50,000$, the probability of failure is about 0.0043. The fact that the number of columns in $\mathbf{H}$ varies from block to block also contributes to failures. However, this is a problem only for large payloads because the probability of failure very quickly decreases with increasing k/m (decreasing message length).

In order for this method to be applicable to as wide range of the parameters k, n, and m as possible, we need to communicate the number of bits embedded in each block. Let us assume k, n, and m are fixed. For the i-th block, let p_i be the largest integer for which the first p_i rows of $\mathbf{H}$ form a matrix of rank p_i. Furthermore, let $f(q), q = 0, \ldots, p-1, p$, be the probability distribution of p_i over the blocks and random matrices $\mathbf{H}$. The

Table 1. Loss of embedding capacity caused by non-solvability of (3) as a function of the relative message length $\alpha = m/k$. The values were obtained experimentally for a cover image with $n = 10^6$ pixels, $k = 50,000$ randomly selected changeable pixels, and $p = 18$.

m/k	0.3	0.4	0.5	0.6	0.7	0.8	0.9
m'/k	0.3	0.4	0.5	0.591	0.660	0.696	0.698

Table 2. Embedding speed in seconds for different relative message length $\alpha = m/k$ for various values of p. The values were obtained experimentally for a cover image with $n = 10^6$ pixels and $k = 50,000$ changeable pixels.

m/k	0.1	0.2	0.25	0.33	0.5
$p = 17$	0.73	2.29	2.30	2.00	2.24
$p = 18$	1.17	4.58	4.19	3.58	3.80
$p = 19$	5.28	10.35	7.99	6.59	9.05
$p = 20$	15.74	17.37	12.82	10.19	18.68

information necessary to communicate p_i is $H(f)$, the entropy of f. Denoting by $E\{f\}$ the mean value of the distribution f, the average number of bits that can be encoded per block is thus $E\{f\} - H(f) \leq p$. Thus, the pure payload $m' = m(E\{f\} - H(f))/p$ that can be embedded is slightly smaller than m. In Table 1, we show the embedding capacity loss for some typical values of m/k. We observe that while this loss is negligible for $m/k \leq 0.6$, it imposes a limit on the maximal relative message length that can be embedded using this method to $\alpha_{\max} = m'/k < 0.698$. In other words, payloads longer than roughly 70% of the maximal embeddable message cannot be embedded using this approach.

From the practical point of view, the sequence p_i should be compressed[†] and then embedded, for example, one bit per block, as the first bit in each block. The decoder first extracts p bits from each block, decompresses the bit sequence formed by the first bits from each block, reads p_i for all blocks, and then discards $p - p_i$ bits from the end of each block message chunk together with the first bit.

In general, the embedding efficiency improves[‡] with the increasing value of p. However, there is a practical limit on the largest usable p imposed by the exponentially increasing complexity and memory requirements. Table 2 shows the embedding time for a one-mega-pixel image (image with $n = 10^6$ pixels), for $k = 50,000$ changeable pixels, for some values of p on a PC equipped with a 3.4 GHz Intel Pentium IV processor. We recommend $p \leq 19$ to in order to keep the embedding time of the order of seconds.

5. EMBEDDING EFFICIENCY

In Section 3 we saw that as we increase the code length, random linear codes asymptotically achieve the theoretical upper bound (6) on the embedding efficiency. However, in practice, we are limited by the computational complexity of the proposed method. We now derive an approximate but sufficiently accurate expression for the embedding efficiency of the method.

Given two integers p and n, let $\mathcal{H}(p, n)$ be the ensemble of all binary matrices of dimension $p \times n$ with n different non-zero columns. The average number of embedding changes for a given matrix $\mathbf{H} \in \mathcal{H}(p, n)$ is the average distance R_a to the code represented by $\mathbf{H}$ (here calculated in the syndrome space using the sets $\mathcal{C}_i$ defined in Section 4.1)

$$R_a = 2^{-p}(|\mathcal{C}_1| + 2|\mathcal{C}_2| + \cdots + R|\mathcal{C}_R|). \tag{7}$$

[†]In practice, the compressed bit-stream will be slightly larger than $H(f)$. Since f is not known to the decoder beforehand, adaptive coders, such as adaptive arithmetic coder, can be used.

[‡]Detailed analysis of how the embedding efficiency depends on p is in Section 5.

Let $c_i(p,n)$, $i=1,\dots,p$, be the expected value of $|\mathcal{C}_i|/2^p$ over matrices $\mathbf{H}$ drawn uniformly from $\mathcal{H}(p,n)$. The expected value of R_a over matrices $\mathbf{H}$ drawn uniformly from $\mathcal{H}(p,n)$ is denoted $r_a(p,n)=\sum_{i=1}^{p} ic_i(p,n)$.

We know that $c_1=n$. In the journal version of this paper, we show that the value of c_2 is (Lemma 3 in Ref. 16)

$$c_2=(1-2^{-p})\left(1-\prod_{i=1}^{n-1}\frac{2^p-2i}{2^p-i}\right).$$

The remaining c_i for $i>2$ will be obtained using an approximate recurrent formula. Let $\mathcal{U}_i$ be the set of all vectors in $\mathbb{F}_2^p$ that can be generated by adding i or fewer columns of a given matrix $\mathbf{H}\in\mathcal{H}(p,n)$. Then, $\mathcal{C}_i=\mathcal{U}_i-\mathcal{U}_{i-1}=\mathcal{C}_i^\star-\mathcal{U}_{i-1}$, where $\mathcal{C}_i^\star$ is the set of all vectors obtainable by adding exactly i different columns of $\mathbf{H}$. There are up to $\binom{n}{i}$ vectors in $\mathcal{C}_i^\star$. We now make a simplifying assumption that $\mathcal{C}_i^\star$ is obtained by random sampling of $\binom{n}{i}$ elements with replacement from the set $\{0,1,\dots,2^p\}$. Under this assumption, $E\{|\mathcal{C}_i^\star|\}=\text{ball}\left(\binom{n}{i},2^p\right)$, where $\text{ball}(k,N)$ denotes the expected number of occupied bins after throwing k balls in N bins, $\text{ball}(k,N)=N-N(1-1/N)^k$. From this assumption, it follows that among all $|\mathcal{C}_i^\star|$ vectors there will be on average $|\mathcal{C}_i^\star|\times|\mathcal{U}_{i-1}|/2^p$ vectors in $\mathcal{U}_{i-1}$. Denoting with lower case letters the expected values of cardinalities of corresponding sets divided by 2^p, we write the following recurrent formula for $c_i(p,n)$, $u_i(p,n)=E\{2^{-p}|\mathcal{U}_i|\}$, and $c_i^\star(p,n)=E\{2^{-p}|\mathcal{C}_i^\star|\}$ for $i=3,\dots,p$ (we leave out the arguments p,n for better readability)

$$\begin{aligned}
c_i^\star &= \text{ball}\left(\binom{n}{i},2^p\right)\\
u_i &= c_i^\star+u_{i-1}-c_i^\star u_{i-1}\\
c_i &\approx u_i-u_{i-1},
\end{aligned}\tag{8}$$

supplied with the initial conditions

$$\begin{aligned}
c_1 &= n2^{-p}\\
u_1 &= c_1\\
c_2 &= (1-2^{-p})\left(1-\prod_{i=1}^{n-1}\frac{2^p-2i}{2^p-i}\right)\\
u_2 &= c_1+c_2.
\end{aligned}$$

Having obtained $c_i(p,n)$, we can now calculate the average number of embedding modifications of the algorithm from Section 4.1. The number of changeable pixels in each block is a random variable κ that follows hyper-geometric distribution

$$\Pr(\kappa=j)=\frac{\binom{k}{j}\binom{n-k}{n/n_B-j}}{\binom{n}{n/n_B}}\text{ for }j=0,\dots,n/n_B.\tag{9}$$

Thus, the average number of embedding changes is

$$R_a(p)=E\{r_a(p,\kappa)\}=E\left\{\sum_{i=1}^{p}ic_i(p,\kappa)\right\}=\sum_{j=1}^{n/n_B}\sum_{i=1}^{p}ic_i(p,j)\Pr(\kappa=j).\tag{10}$$

Finally, the average embedding efficiency is

$$e(p)=p/R_a(p).\tag{11}$$

We have verified the accuracy of (10) using computer experiments in the range $5\le p\le 18$ and $2\le k/m\le 15$. In this range, it was possible to calculate $|\mathcal{C}_i|$ in each block, averaging over $10,000$ blocks (for $n=10^6$ and $k=50,000$). We found out that the difference Δ between $R_a(p)$ obtained using simulations and using (10) is the largest for small values of p and k/m. However, the accuracy quickly improves with increasing p and becomes less than 10^{-2} for $p>10$ across the whole tested range of the relative message length $\alpha=m/k$. We will use this formula to explore the properties of the proposed embedding algorithm for a wider range of p.

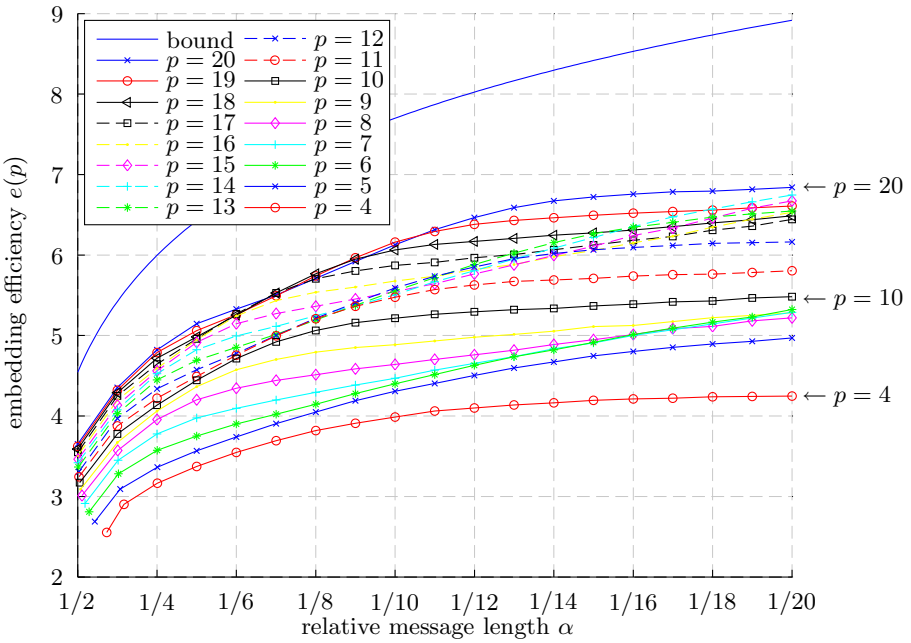

Figure 1. Plot of embedding efficiency $e(p)$ versus relative message length α for $p = 4, \ldots, 20$.

6. EXPERIMENTAL RESULTS

Figure 1 shows the embedding efficiency as a function of the ratio $\alpha^{-1} = k/m$ for a cover image with $n = 10^6$ pixels and $k = 50,000$ changeable pixels for $p = 4, \ldots, 20$. We obtained it by averaging over 100 embeddings of a random message bit-stream into the same cover image with the same parameters k, n, and m. The solid curve is the asymptotic upper bound (6).

The efficiency increases with shorter messages for a fixed p. Once the number of changeable pixels in each set exceeds 2^p, the embedding efficiency starts saturating at $p/(1-2^{-p})$, which is the value that all curves in Figure 1 reach asymptotically with decreasing α. This is because the p/α columns of $\mathbf{H}$ eventually cover the whole space $\mathbb{F}_2^p$ and thus we embed every non-zero syndrome $\mathbf{s}$ using one embedding change (when $\mathbf{s} = \mathbf{0}$ no embedding changes are necessary).

Note that for fixed α, the embedding efficiency increases in a curious non-monotone manner with increasing p. To see this interesting phenomenon more clearly, we plot e as a function of p for various fixed relative message lengths α. The result is shown in Figure 2. The diagram shows the expected value of embedding efficiency obtained from (11) as a function of $p = 4, \ldots, 80$. Each curve corresponds to a different value of $\alpha = 1/2, 1/3, \ldots, 1/200$. The diagram was generated using the approximate formula (8) because it is not computationally feasible to obtain accurate estimates of c_i by direct calculation for such large values of p.

We see from Figure 2 that with increasing value of p the embedding efficiency increases and reaches the asymptotic value given by the bound (6). However, this increase is not monotone. In fact, it is not always true that increasing p will improve the embedding efficiency. For $p = 19$ and $\alpha = 1/10$ (embedding at 10% of embedding capacity), we obtain an improvement only after we increase p beyond 24. Without this knowledge, we may increase p from 19 to 22 hoping to improve the performance because, in general, increasing p improves embedding efficiency. In this case, however, we only increase the embedding time while the embedding efficiency, in fact, decreases! A quantitative explanation of this curious phenomenon is in the journal version of this paper Ref. 16.

7. APPLICATION FOR DATA HIDING IN BINARY IMAGES

Non-shared selection channels arise quite frequently in steganography. For example, the sender may wish to use side information only available to him, such as a high-resolution version of the cover, to determine the

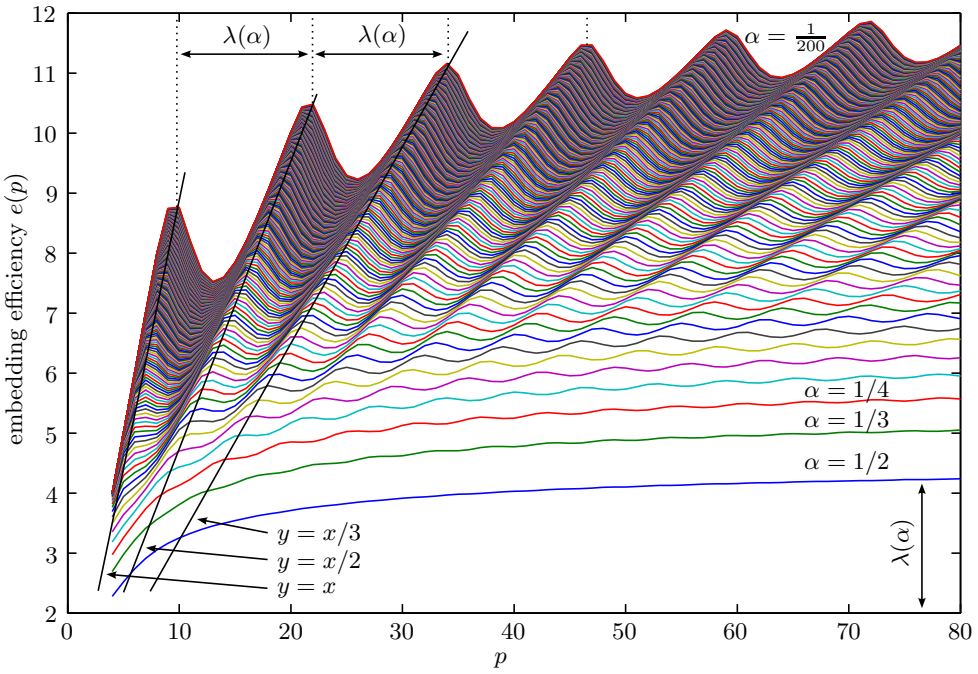

Figure 2. Plot of embedding efficiency $e(p)$ as a function of p for relative message length$\alpha = 1/2, 1/3, \ldots, 1/200$.

Figure 3. Binary 48×288 image "Clinton's signature."

selection channel. The sender may also place the embedding changes based on local context, which is, however, inevitably changed by the embedding process itself and thus the recipient may not be able to recover the same selection channel. This problem is especially pronounced for data hiding in binary images, which we selected to demonstrate the usefulness of the embedding method proposed in this paper.

In a binary image, pixels can only have two colors—black and white. To prevent the embedding process from introducing visible artifacts, the embedding changes should be confined to the boundary between both colors. For example, Wu et al.[17] assign a "flippability score" to each pixel to evaluate the visual impact of changing its color. This measure is determined from a local neighborhood of the pixel. Wu et al. advise to only modify pixels with the highest flippability score to avoid introducing visible artifacts.

Figure 3 is a digitized 48×288 signature of the former president Bill Clinton. This image contains 250 pixels with flippability score 0.625, 32 pixels with score 0.375, 3 pixels with 0.25, 86 with score 0.125, 382 with score 0.1, 662 with 0.05, 71 with 0.01, and the remaining 12338 pixels have score 0. Let us suppose that for authentication purposes we want to embed in the image a 64-bit Message Authentication Code (MAC) and a 16 bit header, making the total payload $m = 80$ bits. To minimize the visual impact of embedding, we would like to use only pixels with the highest flippability score of 0.625. Thus, we have $k = 250$ changeable pixels and the total of $n = 48 \times 288 = 13824$ pixels. The relative message length $\alpha = m/k = 80/250 = 0.32$.

Because the act of embedding may change the flippability score of many pixels, this application calls for wet

paper codes. We can read out from Figure 1 that for $p = 18$ and $\alpha = 0.32$ we can embed with efficiency of roughly $e(18) = 4.4$. Therefore, we can embed the payload of 80 bits with the expected number of hanges equal to $80/4.4 \doteq 18$. This should be contrasted with regular wet paper codes that would introduce about $80/2 = 40$ changes.

The improved embedding efficiency can also be used in a different manner—instead of decreasing the number of embedding changes, we can now embed a larger payload of up to $4.4/2 \times 80 = 176$ bits with the same embedding distortion as the one due to embedding 80 bits using regular wet paper codes. This allows us to trade the decrease of the embedding distortion for the improvement of the robustness of embedded data by applying strong error correction code to the 80 bit payload. Therefore, the improved embedding efficiency can be utilized either for decreasing the visual impact of embedding or to improve the robustness of the embedded data to channel noise or a balance of both.

8. CONCLUSIONS

This paper combines wet paper codes with matrix embedding. As a result, we obtain a general tool for constructing steganographic schemes with non-shared selection channels and improved embedding efficiency. We describe a method that uses random linear codes of small codimension, where the coding can be done using efficient exhaustive searches (the meet-in-the-middle method). We evaluate the performance of the proposed method experimentally and constrast the results with theoretically achievable bounds.

While analyzing the performance of the proposed method, we have discovered a curious transient phenomenon. While the embedding efficiency globally increases with increasing code block length and eventually reaches the theoretical upper bound, it does so in a non-monotone manner. Understanding this transient "ripple" phenomenon is important for practical implementations.

We show an application of the proposed techniques to data hiding in binary images, where the improved embedding efficiency can be used to decrease the embedding distortion or to increase robustness to channel noise. This is because instead of decreasing the distortion, we can embed a larger amount of bits and apply stronger error correction algorithms to the embedded payload.

In our future effort, reflecting on our previous work on application of LT codes to wet paper codes, we plan to investigate low density parity check codes and their iterative decoding algorithms with the intention to obtain good quantizers suitable for steganography with non-shared selection channels with improved embedding efficiency.

9. 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 Petr Lisoněk and Alexander Barg for many useful discussions.

REFERENCES

1. 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.
2. J. Fridrich, M. Goljan, and D. Soukal, "Perturbed Quantization steganography using wet paper codes," in *MM&Sec '04: Proceedings of the 2004 multimedia and security workshop on Multimedia and security*, J. Dittmann and J. Fridrich, eds., *Proceedings of ACM*, ACM Press, (New York, NY, USA), Dec. 6 2004.
3. A. Kuznetsov and B. Tsybakov, "Coding in a memory with defective cells," **10**, pp. 132–138, 1974.
4. J. Fridrich, M. Goljan, P. Lisoněk, and D. Soukal, "Writing on wet paper," in *IEEE Trans. on Sig. Proc., Third Supplement on Secure Media*, **53**, pp. 3923–3935, Oct. 2005.

5. J. Fridrich, M. Goljan, and D. Soukal, "Efficient Wet Paper Codes," in *Proceedings, Information Hiding: 7th International Workshop, IHW 2005*, *Lecture Notes in Computer Science*, Springer-Verlag, (Barcelona, Spain), 2005.
6. R. Crandall, "Some notes on steganography." Posted on Steganography Mailing List. `http://os.inf.tu-dresden.de/~westfeld/crandall.pdf`, 1998.
7. J. Bierbrauer, "On Crandall's problem." Personal communication, 1998.
8. F. Galand and G. Kabatiansky, "Information hiding by coverings," in *Proc. ITW2003*, pp. 151–154, (Paris, France), 2003.
9. F. J. M. Williams and N. J. Sloane, *The Theory of Error-correcting Codes*, North-Holland, Amsterdam, 1977.
10. G. D. Cohen, I. Honkala, S. Litsyn, and A. Lobstein, *Covering Codes*, vol. 54, Elsevier, North-Holland Mathematical Library, 1997.
11. J. Fridrich and D. Soukal, "Matrix embedding for large payloads," in *submitted to IEEE Transactions on Information Security and Forensics*, 2005.
12. E. R. Berlekamp, R. J. McEliece, and H. C. van Tilborg, "On the inherent intractability of certain coding problems," *IEEE Trans. Inf. Theory* **24**, pp. 384–386, May 1978.
13. M. Luby, "LT codes," in *Proc. The 43rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 271–282, Nov. 16–19 2002.
14. T. Wadayama, "An algorithm for calculating the exact bit error probability of a binary linear code over the binary symmetric channel," in *IEEE Trans. Inform. Theory*, **50**, pp. 331–337, Feb. 2004.
15. R. Brent, S. Gao, and A. Lauder, "Random Krylov spaces over finite fields," in *SIAM J. Discrete Math.*, **16(2)**, pp. 276–287, 2003.
16. J. Fridrich, M. Goljan, and D. Soukal, "Wet paper codes with improved embedding efficiency," in *submitted to IEEE Transactions on Information Security and Forensics*, July 2005.
17. M. Wu and B. Liu, "Data hiding for binary image for authentication and annotation," in *IEEE Trans. on Multimedia*, **6**, pp. 528–538, Aug. 2004.