Writing on Wet Paper

[a]Jessica Fridrich[*], [a]Miroslav Goljan, [b]Petr Lisoněk, and [c]David Soukal
[a]Department of Electrical and Computer Engineering, [c]Department of Computer Science
SUNY Binghamton, Binghamton, NY 13902-6000, USA
[b]Dept. of Mathematics, Simon Fraser University, Burnaby, British Columbia V5A 1S6, Canada

ABSTRACT

In this paper, we show that the communication channel known as writing in memory with defective cells[1,2] is a relevant information-theoretical model for a specific case of passive warden steganography when the sender embeds a secret message into a subset C of the cover object X without sharing the selection channel C with the recipient. The set C could be arbitrary, determined by the sender from the cover object using a deterministic, pseudo-random, or a truly random process. We call this steganography "writing on wet paper" and realize it using low-density random linear codes with the encoding step based on the LT process. The importance of writing on wet paper for covert communication is discussed within the context of adaptive steganography and perturbed quantization steganography[3]. Heuristic arguments supported by tests using blind steganalysis[4] indicate that the wet paper steganography provides improved steganographic security for embedding in JPEG images and is less vulnerable to attacks when compared to existing methods with shared selection channels.

Keywords: Wet paper code, steganography, adaptive, selection channel, memory with defective cells

1. INTRODUCTION

The importance of the informed coder channel of Gelfand-Pinsker[5] for steganography and watermarking has been widely recognized by many researchers. A special case of the informed sender channel that is highly relevant for robust watermarking and active warden steganography is Costa's writing on dirty paper[6] and its extensions[7,8]. In this paper, we point out the importance of another special case of the informed sender channel that is known as memory with defective cells[1,2]. When used in passive warden steganography, we coin a new term for this channel and call it "writing on wet paper", which intentionally evokes analogy with Costa's work. Indeed, similar to the result of Costa that the channel capacity is independent of the Gaussian noise known to the encoder but not to the decoder, in the wet paper scenario, quite similarly the channel capacity does not depend on the recipient's knowledge of the selection channel.

To explain the metaphor "writing on wet paper", imagine that the cover object X is an image that was exposed to rain and the sender can only slightly modify the dry spots of X (the set C) but not the wet spots. During transmission, the stego image Y dries out and thus the recipient does not know which pixels were used by the sender. We note that the rain can be random, pseudo-random, completely determined by the sender, or an arbitrary mixture of all. In this communication setup, the sender has a complete freedom in choosing the dry pixels that will be used for embedding because the recipient does not need to determine the dry pixels from the stego image in order to read the message. In particular, the sender may formulate selection rules based on side information that is *in principle unavailable* to the recipient and thus to any attacker. This is likely to provide better security[9–13] than steganographic schemes with public selection rules[14,15].

The "wet paper" channel is highly relevant to steganography and arises in numerous different situations. One of them is adaptive steganography, where the sender selects the location of pixels that will carry message bits based on pixels' neighborhood in the cover image. A fundamental problem with adaptive schemes is that the requirement that the recipient be able to recover the same message-carrying pixels from the stego image undermines the security of the algorithm because it gives an attacker a starting point for mounting an attack[17]. Another potential problem is that the

[*] fridrich@binghamton.edu; phone 1 607 777-2577; fax 1607 777-4464; http://www.ws.binghamton.edu/fridrich

recipient may not be able to recover the same set of message carrying pixels from the stego image, which is modified by the embedding act itself. This problem is usually solved either by increasing the message redundancy using error correction to recover from random bit losses and inserts or by employing some artificial measures, such as special embedding operations matched to the selection rules[14,15]. These measures, however, usually limit the capacity[18], complicate the embedding algorithm, and do not give the sender the ability to fully utilize his side-information – the cover image. Moreover, the pixel selection rule is often ad hoc and it is not always possible to justify it from the point of view of steganographic security. In fact, ideally, the sender should be able to use any available side information in an unrestricted manner and perform embedding while focusing on important steganographic design principles and necessary security considerations rather than the receiver's ability to read the message.

A different case of the wet paper scenario occurs when the cover image is processed before embedding using an information-reducing operation, such as A/D conversion during image acquisition, lossy compression, dithering, recoloring, downsizing, etc. Most today's steganographic algorithms[‡] disregard this side information (the raw cover image before processing) and work with the processed image only. *However, there is no reason why the steganographer could not utilize, for example, the knowledge of the raw, uncompressed cover image when embedding into its JPEG compressed form.* In fact, the sender will have access to the unquantized values of the processed image (e.g., non-rounded DCT coefficients during compression or non-rounded pixel values after downsampling) and may utilize this information for coefficient/pixel selection (Section 4). This side information is largely ignored by current steganographic algorithms because of the seemingly insurmountable obstacle that the receiver would not be able to read the message due to the fact that the coefficient/pixel selection is based on information that is practically completely removed during quantization. On the contrary, we view the fact that one can create an embedding scheme where the information about dry pixels is unavailable to the recipient (and any attacker) as a good property that could substantially improve the steganographic security and remove the above mentioned problem of adaptive steganography.

In Section 2, we show that writing on wet paper can be realized using random linear codes with sparse matrices with an average capacity that reaches the channel capacity. In Section 3, we construct an efficient encoder and decoder using the LT process[20]. In Section 4, we describe several practical steganographic schemes that use wet paper codes, most notably the Perturbed Quantization[3], and briefly evaluate their steganographic security using heuristic arguments and blind steganalysis. The paper is concluded in Section 5 where we outline future research directions and list other applications of wet paper codes.

2. WET PAPER CODES

2.1 Basic concepts

Let us assume that the sender has a cover object X consisting of n elements $\{x_i\}_{i=1}^{n}$, $x_i \in J$, where J is the range of discrete values for x_i. For example, for an 8-bit grayscale image represented in the spatial domain, $J = \{0, 1, \ldots, 255\}$ and n is the number of pixels in the cover image X. The sender uses a Selection Rule (SR) to select k *changeable elements* x_j, $j \in C \subset \{0, 1, \ldots, n-1\}$. The changeable elements may be used and modified by the sender to communicate a secret message to the recipient. The remaining object elements are not modified during embedding.

We further assume that the sender and the recipient agree on a public parity function P, which is a mapping $P: J \to \{0,1\}$. For example, $P(x)$ might be defined as the least significant bit of x, $P(x) = \mathrm{LSB}(x)$. Although we do not consider it in this paper, this mapping could in principle depend on the element position in X and a secret stego key K shared by the sender and the recipient.

During embedding, the sender either leaves the changeable elements x_j, $j \in C$, unmodified or replaces x_j with y_j such that $P(x_j) = 1 - P(y_j)$. The stego object Y will consist, again, of n elements $\{y_i\}_{i=1}^{n}$. The recipient will decode message bits from the bit-stream of parities of stego object elements $\{P(y_i)\}_{i=1}^{n}$.

[‡] A few notable exceptions include the fragile authentication by Marvel et al.[19] and the embedding-while dithering method[15].

Obviously, if the recipient could somehow determine the changeable elements from the stego object, the sender would be able to communicate up to $k = |C|$ bits, one parity bit per each changeable element. However, as discussed in the introduction, there are two problems with this scenario. First, the requirement that the recipient be able to determine the same set of changeable elements imposes a limitation on the SR and the embedding modification. Second, the fact that the message-carrying elements can be determined from the stego object may help an attacker to mount an attack. Thus, we propose a different approach that solves both problems at once – the recipient can read the correct message but does not need to know the set of changeable elements (or even the SR or the embedding operation $x_j \rightarrow y_j$) because the message bits are not communicated directly as element parities. All the recipient needs to share with the sender is a secret key and the public parity function P.

2.2 Writing on wet paper as memory with defective cells

Let $b_i = P(x_i)$ be the sequence of parities of all n elements from the cover object X. The bits b_i are known to the sender. The sender can modify all k bits b_j, $j \in C$, but cannot modify the remaining $n - k$ bits. The recipient does not know the set C. This is an example of a channel known as an n-bit memory containing $n - k$ defective cells, which are stuck either at 0 or 1, introduced by Tsybakov and Kuznetsov[1].

A simple random binning argument[21] can show that asymptotically the capacity of this channel is k. It is also known that the capacity of this channel is k (Ref 2, 22, 23). For alphabet with q symbols, this capacity can be achieved, for example, using MDS erasure codes, such as Reed-Solomon codes. This is easily seen as follows[21]. Each coset of a $[n, n–k, k+1]$ linear MDS code contains all symbol patterns of any $n–k$ stuck cells (this follows directly from the MDS property). Since this code contains q^k cosets, they can be indexed with all possible messages consisting of k symbols. One can then communicate k message symbols by first selecting an appropriate coset and finding in this coset a word with the same pattern of stuck $n–k$ cells as in the memory. Since this word is compatible with the memory defects, the exact same word can be read from the memory later and the k message symbols extracted from the index of the coset to which the word belongs. This approach, however, would be inefficient for our application. By grouping the bits into q-ary symbols, the number of stuck symbols would increase (this is especially a problem for large $n–k$).

In steganographic applications, both the number of defective cells (wet elements) and the dry elements may vary significantly and may be quite large. For example, in the double compression embedding (Perturbed Quantization[3]) briefly described in Section 4.2, for a typical JPEG image, $n \sim 10^6$ and $k \sim 10^4$. Due to the large variability of k from image to image, one cannot assume a reasonable upper bound on the number of defective cells without sacrificing the capacity. Thus, considering these specifics of our application, in the next section we describe a simple random linear code that also enables the sender to communicate on average k bits. In Section 3, we show that the encoding and decoding processes can be efficiently implemented using the LT process[20].

2.3 Wet paper codes

In Sections 2 and 3, we use capital non-bold letters to denote the cover and stego object and their subsets, small bold letters for vectors, and bold capital letters for matrices. The proposed code can be viewed as a generalization of the selection channel[10] where one message bit is communicated as the parity of a group of individual elements. In the selection channel, at most one elements value must be changed in order to match the parity of a group of elements to the message bit. The parity of the group is a sum modulo 2 of the individual element parities. Now, if there are q changeable elements in the group, one can attempt to embed q message bits by forming q linearly independent linear combinations of element parities instead of just one sum. This suggests the following approach to the wet paper code.

Encoder: We repeat that the sender has a binary column vector $\boldsymbol{b} = \{b_i\}_{i=1}^{n}$ and a set of indices $C \subset \{0, 1, \ldots, n–1\}$, $|C| = k$, of those bits that can be modified to embed a message. The sender wants to communicate q bits $\boldsymbol{m} = \{m_1, \ldots, m_q\}^{\mathsf{T}}$, where "$^{\mathsf{T}}$" denotes transposition. For a moment, let us assume that the recipient knows q. We later show how to relax this assumption. The sender and recipient use a shared secret stego key to generate a pseudo-random binary matrix $\boldsymbol{D}$ of dimensions $q \times n$. The sender will modify b_j, $j \in C$, so that the modified binary column vector $\boldsymbol{b}' = \{b'_i\}_{i=1}^{n}$ satisfies

$$\boldsymbol{D}\boldsymbol{b}' = \boldsymbol{m} . \qquad (1)$$

Thus, the sender needs to solve a system of linear equations in GF(2). The question of solvability of (1) is discussed below. Note that the selection channel[10] is a special case of (1) with $\boldsymbol{D} = [1, \ldots, 1]$.

Decoder: The recipient forms the vector $b'_i = P(y_i)$ from the stego object $Y = \{y_i\}_{i=1}^{n}$ and then obtains the message $\boldsymbol{m} = \boldsymbol{D}\boldsymbol{b}'$ using the shared matrix $\boldsymbol{D}$. The biggest computational load is on the sender's side who needs to solve (1).

At this point, we explain how to relax the assumption that the recipient knows q. The sender and recipient can generate the matrix $\boldsymbol{D}$ in a row-by-row manner rather than generating it as a two-dimensional array of $q\times n$ bits. In this way, the sender can reserve the first $\lceil \log_2 n \rceil$ bits of the message $\boldsymbol{m}$ for a header to inform the recipient of the number of rows in $\boldsymbol{D}$ – the message length q. The symbol $\lceil x \rceil$ is the smallest integer larger than or equal to x. The recipient starts by generating the first $\lceil \log_2 n \rceil$ rows of $\boldsymbol{D}$, multiplies them by the received vector $\boldsymbol{b}'$, and reads the header (the message length q). Then, he generates the rest of $\boldsymbol{D}$ and reads the message $\boldsymbol{m} = \boldsymbol{D}\boldsymbol{b}'$.

Note that the decoding mechanism is similar to that of matrix embedding[24], where the recipient also extracts the message bits by multiplying the parity vector by an appropriate code matrix. The difference is that in matrix embedding the sender's goal is to maximize the embedding rate utilizing the *positions of the changes* to convey information. While in matrix embedding any element can be modified, in writing on wet paper the set of elements that can be modified is pre-determined and is different for different cover objects.

We now investigate the issue of solvability of (1) and the average number of bits that the sender can communicate. Obviously, for small q, (1) will have a solution with a very high probability and this probability decreases with increasing q. We rewrite (1) to

$$\boldsymbol{D}\boldsymbol{v} = \boldsymbol{m} - \boldsymbol{D}\boldsymbol{b} \tag{2}$$

using the variable $\boldsymbol{v} = \boldsymbol{b}' - \boldsymbol{b}$ with non-zero elements corresponding to the bits the encoder must change to satisfy (1). In the system (2), there are k unknowns v_j, $j\in C$, while the remaining $n - k$ values v_i, $i\notin C$, are zeros. Thus, on the left hand side, we can remove from $\boldsymbol{D}$ all $n - k$ columns i, $i\notin C$, and also remove from $\boldsymbol{v}$ all $n - k$ elements v_i with $i\notin C$. Keeping the same symbol for $\boldsymbol{v}$, (2) now becomes

$$\boldsymbol{H}\boldsymbol{v} = \boldsymbol{m} - \boldsymbol{D}\boldsymbol{b}, \tag{3}$$

where $\boldsymbol{H}$ is a binary $q\times k$ matrix consisting of those columns of $\boldsymbol{D}$ corresponding to indices C, and $\boldsymbol{v}$ is an unknown $k\times 1$ binary vector. This system has a solution for an arbitrary message $\boldsymbol{m}$ as long as $rank(\boldsymbol{H}) = q$. The probability $P_{q,k}(s)$ that the rank of a random[§] $q\times k$ binary matrix is s, $s \le \min(q, k)$, is (Ref. 25, Lemma 4)

$$P_{q,k}(s) = 2^{s(q+k-s)-qk} \prod_{i=0}^{s-1} \frac{(1-2^{i-q})(1-2^{i-k})}{(1-2^{i-s})}. \tag{4}$$

It can be shown that for a large fixed k, $P_{q,k}(q)$ very quickly approaches 1 with decreasing $q < k$. This suggests that the sender can on average communicate close to k bits to the recipient. In fact, the expected maximum number $q_{\max}$ of bits that can be communicated for a given fixed message $\boldsymbol{m}$ using k changeable bits is[26]

$$q_{\max}(k) = k + O(2^{-k/4}). \tag{5}$$

3. PRACTICAL ENCODER/DECODER IMPLEMENTATION

We would like to stress that the computational requirements are less of an obstacle in our application, which is steganography, because the coding and embedding is usually performed off-line, as opposed to coding for a communication channel where real time performance is essential. Nevertheless, our goal is to develop a practical scheme and this means a scheme with a low complexity that scales well with the cover object size and the message length.

[§] With elements that are iid realizations of a random variable uniformly distributed in {0,1}

The main complexity of this communication is on the sender's side, who needs to solve q linear equations for k unknows in binary arithmetic. Assuming that the maximal length message $q = k$ is sent, the complexity of Gaussian elimination for (3) is $O(k^3)$, which would lead to impractical performance for large payloads, such as $k > 10^5$. In our previous work[3], we proposed to divide the cover object into n/n_B disjoint random subsets (determined from the shared stego key) of a fixed, predetermined size n_B and then perform the embedding for each subset separately. The complexity of embedding was thus linear in the number of cover object elements, albeit with a large multiplicative constant. By selecting n_B such that there are 200–500 changeable elements in each subset, a fast software implementation of Gaussian elimination gave us good performance even for $k \approx 10^5$.

Ideally, we would like to avoid having to solve the system of equations (3) and speed up as well as simplify the coding process. This is indeed possible by making $\boldsymbol{D}$ sparse and imposing a special stochastic structure on its columns. In the next section, we describe a fast and elegant implementation of wet paper codes using the LT process[20] for erasure correcting LT codes.

3.1 LT codes

LT codes were introduced in 2002 by Luby et al.[20] as an example of universal erasure codes with low encoding and decoding complexity that are asymptotically optimal (approaching the Shannon capacity of the erasure channel).

LT encoding: The encoding process of LT codes is best described using a bipartite graph (Fig. 1). The w message symbols are on the left, while the W encoding symbols are on the right. Although the codes can work without any modification with l-bit symbols (packets), in this paper, we only use binary symbols. Each encoding symbol is obtained as an XOR of (on average) $O(\ln(w/\delta))$ randomly selected input symbols (message bits) that are connected to it in the graph. The graph is generated randomly so that the degrees of encoding nodes follow so-called robust soliton distribution. The probability that an encoding symbol node has degree d, is $(\rho(d)+\tau(d))/\beta$, where

$$\rho(i) = \begin{cases} 1/w & \text{for } i = 1 \\ 1/i(i-1) & \text{for } i = 2,\dots,w \end{cases}, \quad \tau(i) = \begin{cases} R/(iw) & \text{for } i = 1,\dots,w/R-1 \\ R\ln(R/\delta)/w & \text{for } i = w/R \\ 0 & \text{for } i = w/R+1,\dots,w \end{cases}, \text{ and } \beta = \sum_{i=1}^{w}(\rho(i)+\tau(i)), \quad (6)$$

and $R = c\ln(w/\delta)\sqrt{w}$ for δ and c suitably chosen constants. According to Luby[20], an arbitrary set of W encoding symbols

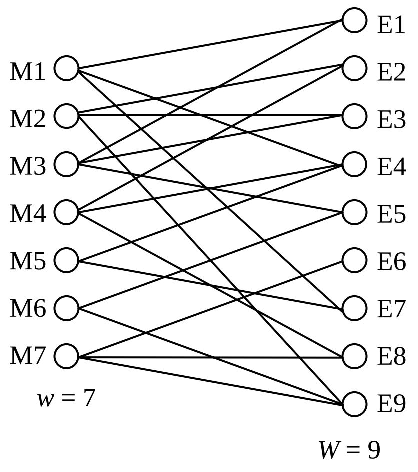
$$W > \beta w = w + O(\sqrt{w}\ln^2(w/\delta)), \quad (7)$$

are needed to uniquely determine all w input symbols with probability better than $1-\delta$. The decoding requires on average $O(w\ln(w/\delta))$ operations.

Note that the vector of encoding symbols can be obtained from the input symbols using matrix multiplication in GF(2) with the bi-adjacency binary matrix $\boldsymbol{A}$ (Fig. 1). The decoding can be obviously done by solving a system of W linear equations (represented with matrix $\boldsymbol{A}$) with w unknowns – the message symbols. As long as this over-determined system has a unique solution, the message is also unique. The beauty of LT codes lies in the decoding step that can be performed with high probability without having to solve the equations. The degree distribution (6) guarantees with probability better than $1-\delta$ that the decoding can be done by iteratively repeating the following simple operation.

LT decoding: Find an encoding symbol that has only one edge (encoding symbol E6 in Fig. 1). In this case, the associated message symbol (M7 in Fig. 1) must be equal to this encoding symbol. Because the message symbol is now known, we can XOR it with all encoding symbols that are connected to it (E8 and E9) and remove it and all its edges from the graph. This step may create some new encoding nodes of degree one (E8). Repeat this process again till all message symbols are recovered.

The decoding process fails if, at some point, there are no encoding symbols of degree 1 while some message symbols still remain undetermined. The encoding degree distribution (6) guarantees that the decoding process will successfully recover all message symbols with probability better than $1-\delta$.



$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Fig. 1. Left: Bipartite graph with 7 message symbols and 9 encoding symbols. Right: its bi-adjacency matrix.

3.2 Using LT process for wet paper codes

The LT process essentially provides a method for a fast solution of an over-determined system of equations $\boldsymbol{Ax} = \boldsymbol{y}$ as long as the number of ones in each row of $\boldsymbol{A}$ follows the robust soliton distribution. We cannot, however, directly apply it to solve (3) because this system is under-determined and we are seeking one solution, possibly out of many solutions. Despite this dissimilarity, the LT process can be used to quickly bring $\boldsymbol{H}$ to the upper triangular form.

It is easy to see that the LT process on the bipartite graph induces the following row/column swapping process on its bi-adjacency matrix. The LT process finds a row in this matrix that has exactly one 1 (say, on the j_1-th place). We now exchange this row with the first row and then exchange the 1st and the j_1-th unknowns (this corresponds to swapping the 1st and j_1-th column). At this point in the LT process, the value of the unknown No. 1 is known – determined from the first equation. In this matrix version of the LT process, however, we do not evaluate the unknowns because we are only interested in bringing $\boldsymbol{A}$ to a lower-triangular form. Continuing the LT process, we now search for another row (besides the first row) that has exactly one 1 among the columns 2, …, w (say, the 1 is in the j_2-th column). If the LT process proceeds successfully, we will be able to do so. Again, we swap this row with the second row and swap the 2nd and j_2-th columns). We continue in this way, now looking for another row besides the first two rows with exactly one 1 in positions 3, …, w, etc. At the end of this process, the matrix $\boldsymbol{A}$ will have zeros above the main diagonal.

Coming back to the wet paper code, our task is to solve the system $\boldsymbol{Hv} = \boldsymbol{m}-\boldsymbol{Db}$ of q equations for k unknowns, $q < k$. By applying the above matrix version of the LT process to $\boldsymbol{H}^{\mathrm{T}}$, we bring $\boldsymbol{H}$ to the upper triangular form with zeros below the main diagonal. In fact, we can work directly with $\boldsymbol{H}$ if we replace in the LT process row swapping with column swapping and vice versa. Of course, in order for the LT process to work, the number of ones in each *column* of $\boldsymbol{H}$ must follow the robust soliton distribution and the message length q must satisfy (from (7))

$$k > \beta q = q + O(\sqrt{q}\ln^2(q/\delta)) . \tag{8}$$

In other words, there is a small price to pay (a slight capacity loss) to enjoy the possibility to solve (3) quickly using the LT process. Depending on the choice of the free parameters c and δ, this loss can be made as small as 5% for $q \geq 10000$ or 10% for $q \geq 1000$.

3.3 Practical implementation issues

Assuming the maximal message is sent ($q \approx k$), the density of 1's in $\boldsymbol{H}$ (and thus in $\boldsymbol{D}$) is $O(\ln(k/\delta)/k)$. Thus, the encoding complexity of the wet paper code implemented using the LT process is $O(n \ln(k/\delta)+k \ln(k/\delta)) = O(n \ln(k/\delta))$. The first term arises from having to calculate the product $\boldsymbol{Db}$, while the second term is the complexity of the LT process. This is a significant savings compared to solving (3) directly using Gaussian elimination. The decoding complexity is $O(n \ln(k/\delta))$, which corresponds to evaluating the product $\boldsymbol{Db}$.

Note that for the LT process, the columns of $\boldsymbol{H}$ (and thus $\boldsymbol{D}$) must comply with the robust soliton distribution that depends on the parameter q, which is unavailable to the decoder and now cannot be communicated in the same way as in Section 2.3. Below, we give a simple solution to this problem, although other more efficient (and more involved) approaches are possible.

Let us assume that the parameter q can be encoded using h bits (in practice, h=20 or so should be sufficient). The sender may use his stego key and divide the cover object X into two pseudo-random disjoint subsets X_h and $X–X_h$ and communicate q using elements from X_h and the main message using elements from $X–X_h$. We need to guarantee, however, that X_h will contain at least h changeable elements, which can be arranged for by requesting that $|X_h|$ be a few percent larger than h/r_{min}, where r_{min} is the minimal value of the ratio k/n that can be encountered in a specific steganographic scheme. Then, using the stego key the sender generates a pseudo-random $h\times|X_h|$ binary matrix $\boldsymbol{D}_h$ with density of 1's equal to ½. The sender now embeds q in X_h by solving the wet paper code equations on X_h using a simple Gaussian elimination. Because $\boldsymbol{D}_h$ has a small number of rows, the Gaussian elimination will be fast. The main message is hidden in $X–X_h$ using the LT process with the parameter q.

The decoder first uses his stego key (and the knowledge of h and r_{min}) to determine the subset X_h and the matrix $\boldsymbol{D}_h$. Then, the decoder extracts q by multiplying $\boldsymbol{D}_h$ with the vector of parities of elements from X_h. Knowing q, the decoder now generates $\boldsymbol{D}$ and extracts the message bits by multiplying $\boldsymbol{D}$ with the parity vector obtained from $X–X_h$.

Finally, we discuss the choice of the parameters c and δ. Because the bounds in Luby's analysis are rather loose, we experimented with a larger range for δ, ignoring for a while its probabilistic interpretation. We found that it was advantageous to set δ to a much larger number (e.g., $\delta = 5$) and, if necessary, repeat the encoding process with matrix $\boldsymbol{D}$ till a successful pass through the LT process is obtained (this requires a few more bits to be encoded with q). For $c = 0.1$, the capacity loss in (8) is about 10% of k for k=1500 with probability of successful encoding about 0.5. This probability increases and capacity loss decreases with increasing k. For example, for k =10000 the capacity loss is only about 5%.

4. APPLICATIONS IN STEGANOGRAPHY

The proposed writing on wet paper enables constructions of new, more secure, steganographic schemes that were not possible before. We show examples of schemes that fit the wet paper communication setup and briefly discuss their steganographic security in comparison to existing schemes. Although we will be explaining the concepts on the example of digital images, the considerations are clearly general enough to apply to other digital objects that allow insertion of steganographic content.

4.1 Adaptive steganography

A typical steganographic algorithm for digital media objects (images, audio, video) embeds one bit per object element (pixel, DCT coefficient, index) by applying an embedding operation to the element. This embedding operation is applied if the sender needs to adjust the "parity" of the element to match the embedded message bit. The elements are selected either sequentially, randomly using a shared secret key, or adaptively based on the cover content. In adaptive steganography, the embedding distortion and selection of message-carrying elements in the cover object is in some way related to the cover content. This often undermines the security of the steganographic system because the set of adaptively selected elements may become available to the attacker[17]. Also, since the act of embedding itself modifies the cover object, care needs to be taken to make sure that the recipient correctly recovers the message. This is usually solved by employing some artificial ad hoc measures whose only purpose is to guarantee the message readability. These measures limit the sender in his choice of the embedding operation and element selection and often severely limit the capacity. Next, we give a few examples of adaptive systems previously proposed in the literature.

Example 1 (Adaptive Least Significant Bit Embedding for images). The sender chooses some local complexity measure σ that is calculated from the 7 Most Significant Bits (MSBs) of pixels located in a small neighborhood $N(x)$ of a given pixel x. The message bits are embedded in Least Significant Bits (LSBs) of those pixels x for which $\sigma(N(x)) > \sigma_0$, where σ_0 is an appropriately chosen threshold shared by the sender and the recipient. In other words, the sender is placing the message bits in the LSBs of pixels located in those parts of the cover image that have a certain minimal level of structure or noise. The recipient will be able to determine the same set of message-carrying pixels as the sender and thus read the message. This is because σ is a function of the 7 MSBs, which are *invariant* with respect to the embedding operation. Note that in this scenario, an attacker also has access to the message-carrying pixels.

Example 2 (Statistics-Preserving LSB Embedding for images). Franz[14] proposed to use the LSB embedding method only for pixels with colors c_1 and c_2 (differing only in their LSBs) that are statistically spatially independent. This independency is evaluated using the chi-square test for statistical independency of the values c_1 and c_2 occurring as spatially neighboring pixels in the cover image (this is done for several orientations of the neighboring pair). The LSBs of all pixels with spatially independent color pairs (c_1, c_2) are replaced with message bits pre-biased to match the relative counts of each color. This embedding mechanism is intended to prevent histogram-based steganalytic attacks[27] and it also guarantees that the recipient will determine the same message-carrying pixels because the chi-square statistics used for determining the color pairs is invariant with respect to embedding changes. Again, the public selection channel gives a starting point to the attacker[17].

Example 3 (Block Parity Embedding). This technique was proposed for color palette images[15]. The image is divided into disjoint blocks B (for example 3×3 blocks) completely covering the image. In each block B, at most one bit will be embedded as the parity of the whole block (e.g., XOR of LSBs of all pixels in B) by changing one pixel in B. As in Example 1, a local block complexity measure σ is selected together with a threshold σ_0. If $\sigma(B) > \sigma_0$, the sender embeds the message bit, obtaining the modified block B', and immediately verifies that $\sigma(B') > \sigma_0$. If the embedding change leads to $\sigma(B') \le \sigma_0$, the sender makes the change anyway and re-embeds the same bit in the next block. The recipient will thus correctly read all message bits from blocks B satisfying $\sigma(B) > \sigma_0$. This method also suffers from the public selection channel. In addition, the necessity to use non-overlapping pixel blocks leads to a significant capacity decrease.

Note that in the examples above the encoder is forced to choose such combinations of the embedding operation and the selection rule that satisfy the requirement of message readability by the receiver, who does not know the cover object. Ideally, the sender should fully focus on the impact of embedding changes on detectability and choose the embedding operation and selection rule accordingly, rather than paying attention to the readability. This is exactly, however, what writing on wet paper enables the sender to do. The selection rule can be completely arbitrary (it can, in fact, contain an element of *true* randomness) and does not have to be shared with the recipient. *The wet paper codes thus solve one of the fundamental problems of adaptive steganography and improve the steganographic security because less information is now available to the attacker.*

4.2 Perturbed quantization

In this section, we give a short description and analysis of an embedding method called Perturbed Quantization that was previously proposed[3] by the authors of this paper. Let us assume that before embedding the sender processes a digital cover image using some information-reducing process F, such as A/D conversion, lossy compression, downsizing, color quantization, etc. The process F typically consists of a real-valued transformation T and an integer quantizer Q. The sender has access to all numerical values before quantization occurs. The largest quantization errors occur for those values that are close to the middle of the quantization intervals of Q. Due to the noise that is commonly present in digital images, the quantization of these values is dominated by the noise and thus closely resembles a random process. The sender may designate such elements (pixels, DCT coefficients) as changeable and use them, together with the wet paper code, for steganography. The remaining elements will be quantized without any changes (those are the wet pixels or "defects" in the "memory"). This method is called Perturbed Quantization (PQ) because the sender slightly perturbs the quantization process in order to embed message bits.

Because the downgrading process is *information-reducing*, an attacker cannot easily recover those fine details of the original image that would enable him to find statistical evidence that some of the elements in the stego image were quantized “incorrectly” (imagine, for example, obtaining a good-enough approximation to the uncompressed cover image from its JPEG compressed form). This is difficult because the sender used side information (the unquantized values) that is essentially removed during quantization and is unavailable to the attacker. Also, the sender can accept additional coefficient selection rule(s) to further decrease the probability of introducing detectable artifacts and thus improve the security. For example, the sender may avoid changing coefficients in those areas of the cover image where the attacker could more easily predict the original coefficient values (e.g., smooth areas or segments with a smooth gradient).

4.2.1 Information-reducing operations

We proceed with providing a more formal description of the embedding process. Let us assume that the cover image X is represented with a vector $\boldsymbol{x}\in I^m$, where I is the range of its pixel/coefficient/color/index values depending on the format of X. For example, for an 8-bit grayscale image, $I = \{0, \ldots, 255\}$. The downgrading process F will be modeled as a transformation

$$F = Q \circ T\colon I^m \to J^n, \tag{9}$$

where J is the integer dynamic range of the downgraded image $Y = F(X)$ represented with an n-dimensional integer vector $\boldsymbol{y}\in J^n$, $m \ge n$. The transform T: $I^m \to \mathbf{R}^n$ is a real-valued transformation and Q: $\mathbf{R}^n \to J^n$ is a quantizer. The intermediate “image” $T(X)$ will be represented with an n-dimensional vector $\boldsymbol{t}\in\mathbf{R}^n$. We give several examples of image downgrading operations F that could be used for steganography based on Perturbed Quantization.

Example 1 (Resizing). For grayscale images, the transformation T maps a square $m_1\times m_2$ matrix of integers x_{ij}, $i = 0, \ldots, m_1-1$, $j = 0, \ldots, m_2-1$ into an $n_1\times n_2$ matrix of real numbers u_{rs}, $n_1 < m_1$, $n_2 < m_2$ using a resampling algorithm. The quantizer Q is a uniform scalar quantizer (rounding to integers), applied to the vector $\boldsymbol{t}$ by coordinates

$$Q(z) = round(z),\ z\in \mathbf{R}. \tag{10}$$

Example 2 (Decreasing the color depth by *d* bits). The transformation T maps a square $m_1\times m_2$ matrix of integers x_{ij} in the range $I = \{0, \ldots, 2^b-1\}$, $i = 0, \ldots, m_1-1$, $j = 0, \ldots, m_2-1$ into a $m_1\times m_2$ matrix of real numbers t_{ij}, $t_{ij} = x_{ij}/2^d$. The quantizer Q is the same uniform scalar quantizer as in Example 1.

Example 3 (JPEG compression). For grayscale images, the transformation T maps a square $m_1\times m_2$ matrix of integers x_{ij}, into a $8\lceil m_1/8\rceil\times 8\lceil m_2/8\rceil$ matrix of real numbers t_{ij} in a block-by-block manner ($\lceil z\rceil$ is the smallest integer larger than or equal to z). In each 8×8 pixel block B_x, the corresponding block B_t in t_{ij} is DCT(B_x)./$\boldsymbol{Q}$, where DCT is the 2D DCT transform, $\boldsymbol{Q}$ is the quantization matrix, and the operation “./” is an element-wise division. The quantizer Q is again given by (10).

4.2.2 Perturbed quantizer

As discussed above, one of the simplest SRs that the sender can formulate is to require the intermediate values t_i of changeable elements $y_i = Q(t_i)$ to be ε-close to the middle of the quantization intervals of Q:

$$C = \{i \mid i\in \{0, \ldots, n\},\ t_i\in [L+0.5-\varepsilon, L+0.5+\varepsilon] \text{ for some integer } L\}. \tag{11}$$

The tolerance ε could in principle be adaptive and depend on the neighborhood of the pixel x_i. It can also be made key dependent, if desired. For this SR, the act of embedding a a random message in the cover image X is well modeled with the probabilistic process $X \to Q_\varepsilon \circ T(X) = Y'$, where Q_ε is the perturbed quantizer and L is an integer,

$$Q_\varepsilon(z) = \begin{cases} L & \text{for } L \le z < L + 0.5 - \varepsilon \\ L+1 & \text{for } L + 0.5 + \varepsilon \le z < L + 1 \\ L \text{ or } L+1 & \text{with equal probability for } L + 0.5 - \varepsilon \le z < L + 0.5 + \varepsilon\,, \end{cases} \tag{12}$$

and Y' is the stego image represented using an integer vector $y' \in J^m$. Note that $Q_\varepsilon = Q$ for $\varepsilon = 0$. The quantizers Q and Q_ε are identical with the exception of the interval $[L+0.5–\varepsilon, L+0.5+\varepsilon)$ where their output differs in 50% of cases. It can be easily shown that, assuming t is a random variable uniformly distributed on [0, 1], the average quantization error $t – Q(t)$ introduced by the scalar quantizer (10) is 1/4, while for the perturbed quantizer (12) it is $1/4+\varepsilon^2$. Thus, the difference between the average error of both quantizers is ε^2, which for $\varepsilon = 0.1$ is at least by one order of magnitude smaller than the average quantization error. Also, note that $–2\varepsilon \le |t – Q(t)| – |t – Q_\varepsilon(t)| \le 2\varepsilon$ for all t.

4.2.3 Embedding while double compressing

The SR can be defined differently based on other heuristics, the image format, and properties of image pixels/coefficients. For example, in Ref 3 a different example of a SR is given when the information-reducing transformation is recompression of the cover JPEG image using a *lower* JPEG quality factor. In this paper we briefly repeat some of the results to illustrate the point that wet paper codes enable construction of steganographic techniques with substantially better steganographic security than previously proposed schemes.

During recompression of a JPEG file, certain values of the DCT coefficients in the cover JPEG file occur in the middle of quantization intervals during the second compression. All these coefficients will be the changeable coefficients. Due to the decompression to the spatial domain, rounding and clipping errors are introduced, which make the second quantization of changeable coefficients resemble a random rather than deterministic process. Thus, the sender rounds the changeable coefficients “up or down” at his will and uses them as the set of “dry” coefficients in a wet paper code.

For certain combinations of quality factors for both JPEG compressions, this embedding technique provides a very large capacity up to 0.5 bits per non-zero DCT coefficient of the stego file. At the same time, the blind JPEG steganalyzer[4] was unable to distinguish between purely double compressed images and fully embedded double compressed images[3]. Table 1 shows the detection accuracy $\rho = 2A–1$, where A is the area under the ROC curve, for a simple linear classifier trained on 1400 cover and 1400 stego (fully embedded) grayscale images of natural scenes and tested on 400 never seen images. The table also shows the detection accuracy for other state-of-the-art steganographic techniques for JPEG images. It is very apparent that the new method offers improved resistance to steganalysis.

bpc	F5	F5_111	OG	MB1	MB2	PQ
0.05	0.2410	0.6451	0.8789	0.2197	0.1631	~ 0
0.1	0.5386	0.9224	0.9929	0.4146	0.3097	0.0484
0.2	0.9557	0.9958	0.9991	0.7035	0.5703	0.0979
0.4	0.9998	0.9999	U	0.9375	0.8243	0.1744
0.6	1.0000	1.0000	U	0.9834	U	U
0.8	1.0000	1.0000	U	0.9916	U	U

Table 1. Detection accuracy ρ for F5 algorithm[28] (F5), F5 with matrix embedding (1,1,1) (F5_111), OutGuess[29] 0.2 (OG), Model based Steganography[30] without and with deblocking (MB1 and MB2, respectively), and the proposed Perturbed Quantization[3] during double compression for different embedding rates expressed using bpc = bits per non-zero stego DCT coefficient (U = unachievable rate). All but the PQ algorithm, were tested with quality factor $Qf = 80$. The PQ algorithm was tested with $Qf_1 = 85$ and $Qf_2 = 70$.

5. CONCLUSIONS

The main contributions of this paper are as follows. First of all, this paper reveals an important relationship between memories with defective cells[1] and steganography. The defective cells correspond to those cover object elements designated by the sender to be avoided for embedding and are *not* shared with the recipient. Because in steganography the number of defective cells could be quite large, we coin a new term for this steganographic channel – *writing on wet paper*. This is a metaphor for a steganographic channel in which the sender embeds message bits into a (dry) subset of elements of the cover object and communicates the message to the recipient, who does not have any information about the selection rule applied by the sender. If the selection rule is determined by side information available only to the sender but in principle unavailable to the recipient (and any attacker), this scenario provides improved steganographic security compared to schemes with a public selection rule[14,15].

Second, we propose a simple random linear code for memories with a *large number* of defects and show how it can be applied for our steganographic channel. This code enables on average communication of *k* bits given *k* "dry" elements (*n*–*k* defective cells). The LT process[20] is used to efficiently implement wet paper codes at a slight loss in available capacity.

Third, we illustrate how wet paper codes can be used to solve some fundamental problems of adaptive steganography and we briefly discuss a new approach to steganography for digital media called Perturbed Quantization[3]. In Perturbed Quantization, the sender embeds a secret message while downgrading the cover object using some information-reducing operation, such as lossy compression, A/D conversion, downsampling, etc. The sender uses his knowledge of the *unprocessed* object and embeds data into those pixels/coefficients whose values are the most "uncertain" after the processing. We illustrate the methodology on the example of recompressing a JPEG image with a lower quality factor. Using heuristic arguments supported with blind steganalysis, it is shown that Perturbed Quantization is significantly less detectable than existing steganographic methods for JPEG images while providing a relatively large capacity.

We note that the writing on wet paper and the proposed wet paper code can be thought of as a generalization of the selection channel[10]. The wet paper is also a special case of the general problem of communication with informed sender[5]. While the Costa's dirty paper code[6] is relevant for watermarking[7,8], the wet paper is a suitable model for steganography. Both channels are special cases of the general problem of communication with informed sender.

There are numerous applications of the wet paper code in steganography and general data embedding. For example, we name the masking of embedding places for public key steganography[9,10]. In public key steganography, the message is encrypted using the standard public key encryption methodology and hidden using a public selection channel. The public knowledge of the selection channel undermines the security of the steganography. The wet paper code enables public extraction of the encrypted message using the public decoding matrix $\boldsymbol{D}$, but at the same time, it completely masks the pixels that were used for embedding, since they can be used by the sender in an arbitrary manner.

Another application is constructing steganographic schemes that, besides the secret shared stego key, contain an element of true randomness and thus cannot be subjected to brute force stego key searches[31]. As the last application, we mention data hiding in binary images proposed by Wu[18]. In this application, the sender first identifies the set of "flippable" pixels that can be modified for embedding. Because this set of pixels is not shared with the recipient, Wu proposed block embedding combined with random shuffling. The block embedding however, leaves most of the flippable pixels unused and only a fraction of the embedding capacity is used. Because this situation exactly corresponds to writing on wet paper, the capacity of this data hiding method can be dramatically improved.

ACKNOWLEDGEMENTS

The work on this paper was supported by Air Force Research Laboratory, Air Force Material Command, USAF, under the research grant number 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.

REFERENCES

1. A.V. Kuznetsov and B.S. Tsybakov, "Coding in a Memory with Defective Cells", *Probl. Inform. Transmission* **10**, pp. 132–138, 1974.
2. C. Heegard and A. El-Gamal, "On the Capacity of Computer Memory with Defects," *IEEE Trans. Inf. Th.* **29**, pp. 731–739, 1983.
3. J. Fridrich, M. Goljan, and D. Soukal, "Perturbed Quantization Steganography with Wet Paper Codes", *Proc. ACM Multimedia and Security Workshop*, Magdeburg, Germany, Sep. 20–21, pp. 4–15, 2004.

4. J. Fridrich, "Feature-Based Steganalysis for JPEG Images and its Implications for Future Design of Steganographic Schemes", in: J. Fridrich (ed.): 6th International Workshop. *Lecture Notes in Computer Science*, vol. 3200. Springer-Verlag, New York (to appear), 2004.
5. S.I. Gelfand and M.S. Pinsker, "Coding for channel with random parameters," *Probl. Pered. Inform.* (*Probl. Inform. Transm.*) **9**(1), pp. 19–31, 1980.
6. M. H. M. Costa, "Writing on dirty paper," *IEEE Trans. Inform. Theory* **29**(3), pp. 439–441, 1983.
7. P. Moulin and J. A. O'Sullivan, "Information-Theoretic Analysis of Information Hiding," *IEEE Trans. on Inf. Th.* **49**(3), pp. 563–593, 2003.
8. B. Chen and G. Wornell, Quantization Index Modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding", *IEEE Trans. on Inf. Th*. **47**(4), pp. 1423–1443, 2001.
9. F.A.P. Petitcolas and S. Katzenbeisser, (eds.), *Information Hiding Techniques for Steganography and Digital Watermarking*, Artech House Books, 2000.
10. R.J. Anderson and F.A.P. Petitcolas, "On the Limits of Steganography", *IEEE Journal of Selected Areas in Communications*, Special Issue on Copyright and Privacy Protection **16(**4), pp. 474–481, 1998.
11. C. Cachin, "An Information-Theoretic Model for Steganography", in: D. Aucsmith (ed.): Information Hiding. 2nd International Workshop. *Lecture Notes in Computer Science*, vol. 1525. Springer-Verlag, New York, pp. 306–318, 1998.
12. J. Zöllner, H. Federrath, H. Klimant, A. Pfitzmann, R. Piotraschke, A. Westfeld, G. Wicke, and G. Wolf, "Modeling the Security of Steganographic Systems", in: D. Aucsmith (ed.): Information Hiding. 2nd International Workshop. *Lecture Notes in Computer Science*, vol. 1525. Springer-Verlag, New York, pp. 344–354, 1998.
13. S. Katzenbeisser and F.A.P. Petitcolas, "Defining Security in Steganographic Systems", *SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents IV*, vol. 4675, San Jose, pp. 50–56, 2002.
14. E. Franz, "Steganography Preserving Statistical Properties", in: F.A.P. Petitcolas (ed.): Information Hiding. 5th International Workshop. *Lecture Notes in Computer Science*, vol. 2578. Springer-Verlag, New York, pp. 278–294, 2002.
15. J. Fridrich and R. Du, "Secure Steganographic Methods for Palette Images", in: A. Pfitzmann (ed.): Information Hiding. 2nd International Workshop. *Lecture Notes in Computer Science*, vol. 1768, Springer-Verlag, New York, pp. 47–60, 2000.
16. M. Karahan, U. Topkara, M. Atallah, C. Taskiran, E. Lin, E. Delp, "A Hierarchical Protocol for Increasing the Stealthiness of Steganographic Methods", *Proc. ACM Multimedia and Security Workshop*, Magdeburg, Germany, September 20–21, pp. 16–24, 2004.
17. A. Westfeld and R. Böhme, "Exploiting Preserved Statistics for Steganalysis", in: J. Fridrich (ed.): 6th International Workshop. *Lecture Notes in Computer Science*, vol. 3200, Springer-Verlag, New York (to appear), 2004.
18. M. Wu, Tang E., and Liu B., "Data Hiding in Digital Binary Image", *Proc. Conf. on Multimedia & Expo* (ICME'00) (CD version), New York City, 2000.
19. L. M. Marvel, G. W. Hartwig, Jr., and C. Boncelet, Jr., "Compression-Compatible Fragile and Semi-Fragile Tamper Detection", *Proc. SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents II*, vol. 3971, San Jose, pp. 131–139, 2000.
20. M. Luby, "LT Codes", *Proc. The 43rd Annual IEEE Symposium on Foundations of Computer Science*, November 16–19, pp. 271–282, 2002.
21. R. Zamir, S. Shamai, U. Erez, "Nested Linear/Lattice Codes for Structured Multiterminal Binning", IEEE *Trans. Inf. Th*. **48**(6), pp. 1250–1276, 2002.
22. C. Heegard, "Partitioned Linear Block Codes for Computer Memory with 'Stuck-at' Defects," *IEEE Trans. Inf. Th.* **29**, pp. 831–842, 1983.
23. G. Cohen, "Applications of Coding Theory to Communication Combinatorial Problems, *Discrete Math.* **83** (2–3), pp. 237–248, 1990.

24. R. Crandall, “Some Notes on Steganography”, posted on Steganography Mailing List, http://os.inf.tu-dresden.de/~westfeld/crandall.pdf, 1998.
25. R.P. Brent, S. Gao, and A.G.B. Lauder, “Random Krylov Spaces Over Finite Fields”, *SIAM J. Discrete Math*. **16**(2), pp. 276–287, 2003.
26. J. Fridrich, M. Goljan, Petr Lisoněk, and D. Soukal, “Writing on Wet Paper”, submitted to *IEEE Trans. on Sig. Proc*., Special Issue on Media Security, 2005.
27. A. Westfeld and A. Pfitzmann, “Attacks on Steganographic Systems”, in: A. Pfitzmann (ed.): 3rd International Workshop. *Lecture Notes in Computer Science*, vol.1768. Springer-Verlag, New York, pp. 61−75, 2000.
28. A. Westfeld, “High Capacity Despite Better Steganalysis (F5–A Steganographic Algorithm)”, in: I.S. Moskowitz (ed.): 4th International Workshop on Information Hiding, *Lecture Notes in Computer Science*, vol. 2137. Springer-Verlag, New York, pp. 289–302, 2001.
29. N. Provos, “Defending Against Statistical Steganalysis”, 10th USENIX Security Symposium. Washington, DC 2001.
30. P. Sallee, “Model Based Steganography”, In: T. Kalker, I.J. Cox, and Yong Man Ro (eds.): International Workshop on Digital Watermarking, *Lecture Notes in Computer Science*, vol. 2939. Springer Verlag, New York, pp. 154–167, 2004.
31. J. Fridrich, M. Goljan, D. Soukal, and T. Holotyak, “Forensic Steganalysis: Determining the Stego Key”, *submitted to IEEE Trans. on Sig. Proc.*, June 2004.