Writing on Wet Paper

Jessica Fridrich, Miroslav Goljan, Petr Lisoněk, David Soukal

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 *C* with the recipient. The set *C*, also called the selection channel, 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 a simple variable-rate random linear code that gives the sender a convenient flexibility and control over the embedding process and is thus suitable for practical implementation. The importance of the wet paper scenario 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 and is less vulnerable to steganalytic attacks compared to existing methods with shared selection channels.**

Index Terms — **Codes, communication, informed coding, memory with defective cells, perturbed quantization, steganography, wet paper.**

I. Motivation

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], which is actually one of the first examples of the general theory of coding with side information. Due to certain specifics of this channel when used in passive warden steganography (see 2nd paragraph in Section II-B), for stego applications we coin a new term for this channel and call it "writing on wet paper", which intentionally evokes analogy with Costa's work.

To explain the metaphor "writing on wet paper", imagine that *X* is an image exposed to rain and the sender can only slightly modify the dry spots of the cover image *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. This communication setup gives the sender 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 pixel 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]–[17].

The "wet paper" channel is highly relevant to steganography and arises in several 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 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, may severely limit the embedding capacity [38], complicate the embedding algorithm, and/or 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 not always justifiable from the point of view of steganographic security.

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 acquisition, lossy compression, dithering, downsizing, etc. Most today's steganographic algorithms[1] disregard this side information (the 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 has access to the unquantized values of the processed image

Manuscript received February 11, 2005. 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.

[1] A few notable exceptions include the fragile authentication by Marvel et al. [18] and the embedding-while dithering method [15].

(e.g., non-rounded DCT coefficients during compression or non-rounded pixel values after downsampling) and may use this information for coefficient/pixel selection (Section IV). This side information is largely ignored by most 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 uses information most of which is removed during quantization. On the contrary, we view the fact that one can create an embedding scheme where the location of dry pixels is unavailable to the recipient (and any attacker) as a good property that can substantially improve the steganographic security and remove the above mentioned problem of adaptive steganography.

In Section II, we describe a variable-rate random linear code for writing on wet paper and show that the average payload per pixel that can be communicated asymptotically reaches the channel capacity as the number of pixels tends to infinity. In Section III, we give a detailed description of the encoder and decoder while paying close attention to implementation issues. We also discuss further improvements, such as minimizing the impact of embedding changes or using random linear codes with sparse matrices. In Section IV, we describe several practical steganographic schemes that use wet paper codes, most notably the Perturbed Quantization [3], and briefly discuss their steganographic security using heuristic arguments and blind steganalysis. The paper is concluded in Section V where we outline future research directions and list a few other applications of wet paper codes.

II. Wet Paper Codes for Steganography

A. 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, $j \in C \subset \{1, \ldots, n\}$. The changeable elements may be used and modified by the sender to communicate a secret message to the recipient. The remaining 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 $P: J \rightarrow \{0,1\}$. Although we do not consider it in this paper, this mapping could in principle depend on the element position i 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 elements from the stego object $\{P(y_i)\}_{i=1}^{n}$.

Obviously, if the recipient could determine the same set of 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.

B. 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. All 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 in 1974 by Tsybakov and Kusnetsov [1]. A simple random binning argument (see, e.g., [19]) proves existence of codes whose rate asymptotically reaches the Shannon capacity of this channel, which is k/n [2][20][21]. For non-binary alphabets, this capacity can be achieved, for example, using an algebraic coding scheme that uses the cosets of an erasure correction code as bins [19]. A noisy generalization of this channel is given in [19], where it is shown that the early work on partitioned codes [22][23] are cases of nested linear codes capable of achieving the Shannon capacity.

There are certain specifics of steganographic applications that distinguish writing on wet paper from defective memory and thus in our opinion warrant coining a new term. First, in a typical steganographic application, both the number of defective cells (wet elements) and the dry elements may be quite large. For example, in Perturbed Quantization while recompressing a JPEG file [3], briefly described in Section IV-B, for a typical JPEG image, $n \sim 10^6$ and $k \sim 10^4$. Second, the number of changeable elements k may vary greatly from image to image and thus one cannot assume a reasonable upper bound on the number of defective cells without sacrificing the embedding capacity. Third, fortunately, coding for steganographic applications has much less stringent requirements on computational complexity because data embedding is typically run off-line as opposed to coding for writing to defective memories. While it is acceptable to spend a few seconds embedding 10000 bits but it is unacceptable to spend this time writing data into memory.

Considering these specifics of stego applications, in the next section we describe a simple variable-rate random linear code that also enables the sender to communicate on average k bits. The simplicity of this code enables efficient implementation of the wet paper scenario for steganography. Another advantage of this code is its flexibility and control it gives to the sender to choose which elements should be modified, which further improves steganographic security and minimizes the impact of embedding changes (Section III-A).

C. Wet paper codes

1) Encoder

In Sections II and III, we use capital non-bold letters to denote the cover and stego object and their subsets, bold small letters for vectors, and bold capital letters for matrices. The proposed code can be viewed as a generalization of the selection channel proposed by Anderson and Petitcolas [10] where one message bit is embedded as the parity of a group of individual elements. In the selection channel, at most one element 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.

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

$$\boldsymbol{D}\boldsymbol{b}' = \boldsymbol{m}\,. \tag{1}$$

Thus, the sender needs to solve a system of linear equations in GF(2). The question of solvability of (1) is discussed in detail in Section II-C3. Note that the selection channel of [10] is a special case of (1) when $\boldsymbol{D}$ is of dimensions $1\times n_B$ consisting of all 1's, $\boldsymbol{D} = [1\ 1\ \ldots\ 1]$, where n_B is the number of elements in each group.

2) Decoder

The modified stego object $Y = \{y_i\}_{i=1}^{n}$ is sent to the recipient. The decoding is very simple because the recipient first forms the vector $b'_i = \text{Parity}(y_i)$ and then obtains the message $\boldsymbol{m} = \boldsymbol{D}\boldsymbol{b}'$ using the shared matrix $\boldsymbol{D}$. Note that the recipient does not need to know the set of changeable elements C to read the message.

We now 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. This allows us to use variable length message structure, where the length of the message is in some header. 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 $\lceil x\rceil$ denotes the smallest integer larger than or equal to x. The recipient first generates 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 $\boldsymbol{m} = \boldsymbol{D}\boldsymbol{b}'$.

The decoding mechanism is similar to that of matrix embedding [24][25], where the recipient also extracts the message bits by multiplying the parity vector by an appropriate 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 by the sender (or the cover object, or some randomness) beforehand and determines the embedding capacity.

3) Average maximal payload

We now investigate the issue of solvability of (1) and determine 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 (see Lemma 4 in [26])

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

From Lemma 1 and (A2) in Appendix A, it can be shown that for a large fixed k, $P_{q,k}(q)$ quickly approaches 1 with decreasing $q < k$ (Figure 1). This suggests that the sender can on average communicate close to k bits to the recipient. We now prove that the expected number of bits that can be communicated is approximately equal to k.

Trying to embed the longest possible message, the sender keeps on adding rows to $\boldsymbol{D}$ while (3) still has a solution. The probability that the sender can communicate at least $k-r$ ($r \ge 0$) bits will be denoted $p_{\ge k-r}$. This will happen when the first $k-r$ rows in $\boldsymbol{H}$ form a submatrix whose rank is $k-r$ or its rank is $k-r-i$ and each of the i linearly dependent rows is compatible with the corresponding bit on the right hand side, which will happen with probability 2^{-i}. Thus,

$$p_{\ge k-r} = \sum_{i=0}^{k-r}\frac{1}{2^i}P_{k-r,k}(k-r-i)\,, \tag{5}$$

while the probability that one can communicate at least $k + r$ ($r \geq 0$) bits is, using a similar argument,

$$p_{\geq k+r} = \sum_{i=0}^{k} \frac{1}{2^{r+i}} P_{k+r,k}(k-i) \,. \tag{6}$$

From (5–6), the expected maximum number $q_{\max}$ of bits that can be communicated using k changeable bits is

$$q_{\max}(k) = \sum_{i=1}^{\infty} i p_{=i} = \sum_{i=1}^{\infty} i(p_{\geq i} - p_{\geq i+1}) \,, \tag{7}$$

where $p_{=i} = p_{\geq i} - p_{\geq i+1}$ is the probability that one can communicate exactly i bits. In Figure 2, we show the probability distribution $p_{=i}$, which appears to be symmetrical about i=k and quickly falls to zero to the sides. This indicates that $q_{\max}(k) \approx k$, which is indeed the case. A precise formulation of this statement and its derivation is given in Appendix A. This result means that on average, using the wet paper code described above, the sender will be able to communicate approximately k bits to the recipient.

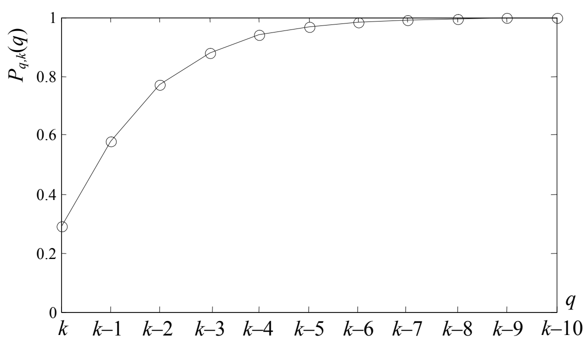

Fig. 1. Probability that a random $q\times k$ binary matrix has rank q (for k=100).

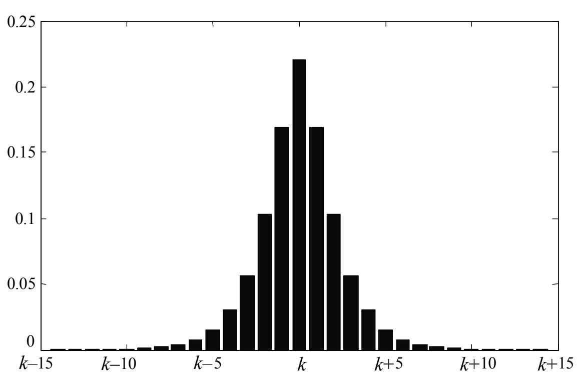

Fig. 2. PDF $p_{=i}$ for the number of communicated bits.

III. Practical Encoder/Decoder Implementation

The main complexity of this communication setup is on the side of the sender, who needs to solve q linear equations for k unknows in GF(2) (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 $k > 10^5$. As explained in Section II-B, we would like to stress that the computational requirements are less of an obstacle in steganography, because the embedding is usually performed off-line, as opposed to coding for a communication channel where real time performance is essential. Our goal is to find an implementation of the encoder/decoder with embedding of the order of a few seconds for typical steganographic scenarios, cover images, and payloads.

By far the best performance and most flexible method for solving (3) was obtained using structured Gaussian elimination by dividing the bit-stream $\boldsymbol{b}$ into β disjoint pseudo-random subsets B_i and using the Gaussian elimination on each subset separately. This can bring down the computational requirements substantially because the complexity of Gaussian elimination decreases by the factor of β^3 while the number of solvings increases β-times, which gives performance improvement of β^2. The division into subsets, however, requires communication of the message length in each subset, which leads to a slight decrease in channel capacity (typically a few percent). Overall, the small decrease in capacity is well worth the significant improvement in speed. The performance of the structured Gaussian elimination is evaluated and compared to other solvers in Section III-D.

Let us assume that the communicating parties know the range of typical values of the rate $r = k/n$, $r_1 \leq r \leq r_2$. If the range is unknown or r_2/r_1 is too large, the sender can modify the pseudo-code below and communicate r to the recipient (see Section III-C). The specific value of r is influenced by the cover object content, the SR, and other specifics of the embedding algorithm. To keep the encoding time short, we desire approximately $k_{\text{avg}} \sim 250$ changeable bits in each subset. We also require all subsets to be approximately of the same size. Thus, we choose the number of sets $\beta = \lceil nr_2/k_{\text{avg}} \rceil$. The size n_i of each subset B_i will be $n_i \in \{\lfloor n/\beta \rfloor, \lceil n/\beta \rceil\}$ chosen so that $n_1+n_2+\ldots+n_\beta = n$. Both the encoder and decoder must follow the same pseudo-random process for dividing $\boldsymbol{b}$ into subsets. This process may use the stego key as a parameter.

The number of changeable bits k_i varies for each subset B_i and follows the hypergeometric distribution (see Section IIIB1) with mean value k/β. The sender allocates the number of bits q_i embedded in i-th subset dynamically (see the pseudo-code below) by trying to embed as many bits q_i in each subset as possible (as in Section II-C3). Note that in this setup the sender determines q_i at the end of the embedding and thus cannot embed q_i in the header of the same subset. We solve this problem by embedding the payload lengths q_i in the last subsets and let the recipient proceed backwards.

Without changing the notation, we will assume that the bits $\boldsymbol{b}$ are permuted using a pseudo-random permutation generated from a shared secret stego key. Then, the subsets B_i can simply be taken as segments of n_i consecutive bits and $\boldsymbol{b} = (\boldsymbol{b}^{(1)}, \boldsymbol{b}^{(2)}, \ldots, \boldsymbol{b}^{(\beta)})$, where $\boldsymbol{b}^{(i)}$ is a vector of n_i bits from B_i. The pseudo-code for the encoding and decoding algorithm is given below (follow Figure 3). We assume r_1, r_2, and k_{avg} are publicly known parameters agreed upon by the sender and the recipient.

We explain Steps E4 and E12 in more detail. In both steps, the sender forms an upper diagonal $q_i \times k_i$ matrix from $\boldsymbol{H}^{(i)}$ using Gaussian elimination, exchanging columns as needed to obtain 1's everywhere on the main diagonal. Since there are more unknowns than equations ($q_i \le k_i$), the sender can set $v_i = 0$ for $i = q_i+1, \ldots, k_i$ and run the back substitution in Gaussian elimination for the unknowns v_i, $i = 1, \ldots, q_i$. Assuming the message is a random bit-stream, the expected number of 1's in the solution vector $\boldsymbol{v}$ will be $q_i/2$. Thus, on average we embed q_i bits by making $q_i/2$ changes (embedding efficiency 2).

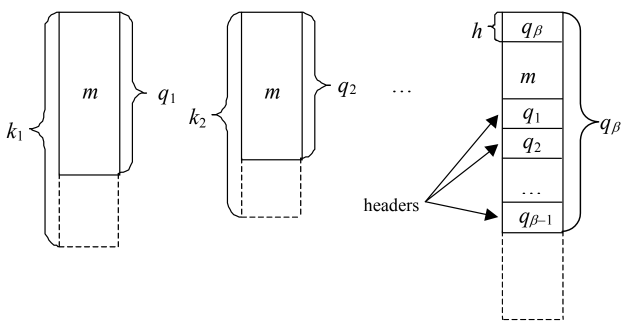

Fig. 3. Placement of message bits and headers within subsets.

Also, the encoder is allowed to decrease q_i whenever it cannot form an upper diagonal matrix (with ones on the diagonal) from $\boldsymbol{H}^{(i)}$ using Gaussian elimination and by exchanging columns. Note that the encoding process may fail in the last subset because this is the only subset in which the sender doesn't have the freedom to decrease q_β. To minimize the probability of this happening when q is close to k, we force the encoder to embed slightly more bits in all other subsets than in the last one. This is the reason why the sender starts dividing the message bits with $q+10$ rather than q.

Encoder

E0. Calculate $\beta = \lceil nr_2/k_{\text{avg}} \rceil$. Using a PRNG, generate a random binary matrix $\boldsymbol{D}$ with $\lceil n/\beta \rceil$ columns and sufficiently many rows

E1. Determine the header size $h = \lceil \log_2(nr_2/\beta) \rceil + 1$, $q = |\boldsymbol{m}| + \beta h$

E2. $\boldsymbol{b}' \leftarrow \boldsymbol{b}$, $i \leftarrow 1$

E3. $q_i = \lceil k_i(q+10)/k \rceil$, $q_i = \min\{q_i, 2^h-1, |\boldsymbol{m}|\}$, $\boldsymbol{m}^{(i)} \leftarrow$ the next q_i bits in $\boldsymbol{m}$

E4. Select the first n_i columns and q_i rows from $\boldsymbol{D}$ and denote this submatrix $\boldsymbol{D}^{(i)}$. Solve q_i equations $\boldsymbol{H}^{(i)}\boldsymbol{v} = \boldsymbol{m}^{(i)} - \boldsymbol{D}^{(i)}\boldsymbol{b}^{(i)}$ for k_i unknowns $\boldsymbol{v}$, where $\boldsymbol{H}^{(i)}$ is a $q_i \times k_i$ submatrix of $\boldsymbol{D}^{(i)}$ consisting of those columns of $\boldsymbol{D}^{(i)}$ that correspond to changeable bits in B_i. If this system does not have a solution, the encoder decreases q_i till a solution is found

E5. According to the solution $\boldsymbol{v}$, obtain the i-th segment $\boldsymbol{b}'^{(i)}$ of the vector $\boldsymbol{b}'$ by modifying or leaving $\boldsymbol{b}^{(i)}$ unchanged

E6. Binary encode q_i using h bits and append them to $\boldsymbol{m}$

E7. Remove the first q_i bits from $\boldsymbol{m}$

E8. $q \leftarrow q - q_i$, $k \leftarrow k - k_i$, $i \leftarrow i+1$

E9. IF $i < \beta$ GOTO 3

E10. IF $i = \beta$, $q_\beta \leftarrow q$

E11. Binary encode q_β using h bits and *prepend* to $\boldsymbol{m}$, $\boldsymbol{m}^{(\beta)} \leftarrow \boldsymbol{m}$

E12. Select the first n_β columns and q_β rows from $\boldsymbol{D}$ and denote this submatrix $\boldsymbol{D}^{(\beta)}$. Solve q_β equations $\boldsymbol{H}^{(\beta)}\boldsymbol{v} = \boldsymbol{m}^{(\beta)} - \boldsymbol{D}^{(\beta)}\boldsymbol{b}^{(\beta)}$ for k_β unknowns $\boldsymbol{v}$. If this system does not have a solution, exit and report failure to embed the message.

E13. According to the solution $\boldsymbol{v}$, obtain the β-th segment $\boldsymbol{b}'^{(\beta)}$ of the vector $\boldsymbol{b}'$ by modifying or leaving $\boldsymbol{b}^{(\beta)}$ unchanged

Decoder

D0. Calculate $\beta = \lceil nr_2/k_{\text{avg}} \rceil$. Using a PRNG, generate a random binary matrix $\boldsymbol{D}$ with $\lceil n/\beta \rceil$ columns and sufficiently many rows

D1. Determine the header length $h = \lceil \log_2(nr_2/\beta) \rceil + 1$

D2. $i \leftarrow \beta$

D3. $\boldsymbol{D} \leftarrow$ the first n_β columns from $\boldsymbol{D}$
$\boldsymbol{D}^{(\beta)} \leftarrow$ the first h rows of $\boldsymbol{D}$, read q_β as $\boldsymbol{D}^{(\beta)}\boldsymbol{b}'^{(\beta)}$

D4. $\boldsymbol{D}^{(\beta)} \leftarrow$ the next $q_\beta - h$ rows of $\boldsymbol{D}$, $\boldsymbol{m} = \boldsymbol{D}^{(\beta)}\boldsymbol{b}'^{(\beta)}$

D5. $i \leftarrow i - 1$

D6. Decode q_i from the last h bits of $\boldsymbol{m}$ and remove the last h bits from $\boldsymbol{m}$

D7. Select the first n_i columns and q_i rows from $\boldsymbol{D}$ and denote this submatrix $\boldsymbol{D}^{(i)}$. $\boldsymbol{D} \leftarrow$ the first q_i rows of $\boldsymbol{D}^{(i)}$, prepend $\boldsymbol{D}\boldsymbol{b}'^{(i)}$ to $\boldsymbol{m}$, $\boldsymbol{m} \leftarrow \boldsymbol{D}\boldsymbol{b}'^{(i)} \& \boldsymbol{m}$

D8. IF $i > 1$ GOTO 5
ELSE $\boldsymbol{m}$ is the extracted message

Notice also that the sender reserves one more bit for headers to cover a possibly larger k_i in a subset than the expected value k/β. Because the header in each subset has h bits, the message length in one subset must not exceed 2^h-1 (Step E3). The maximum number of bits that can be communicated using this algorithm is approximately $k - \beta h = k - \beta\lceil \log_2(r_2k/\beta) \rceil$. The performance of this algorithm is discussed in Section III-D.

A. Minimizing the impact of embedding

When embedding a shorter than maximal message, in Steps E4 and E12 the sender has freedom in choosing which unknowns v_i should be set to 0 and which will be determined by the Gaussian elimination. This freedom can be used to further minimize the impact of embedding. The SR is usually formulated in quantitative terms and thus it will be possible to associate with each changeable element x_i a numerical value $f(x_i)$ that somehow expresses its "fitness" to be included in the set of changeable elements. When solving q_i equations $\boldsymbol{H}^{(i)}\boldsymbol{v} = \boldsymbol{m}^{(i)} - \boldsymbol{D}^{(i)}\boldsymbol{b}^{(i)}$ for k_i unknowns $\boldsymbol{v}$, the sender can solve for those unknowns v_i that correspond to elements with the largest fitness and set the remaining v_i's to zero. This way, the impact of embedding is further minimized and the security improved.

B. Imposing structure on D to speed up the coding

An obvious question to ask is whether it is possible to solve (3) faster by imposing some structure on the matrix $\boldsymbol{D}$ (and thus indirectly on $\boldsymbol{H}$). Recall that $\boldsymbol{H}$ is obtained from $\boldsymbol{D}$ by selecting those columns of $\boldsymbol{D}$ that correspond to changeable elements. The matrix $\boldsymbol{D}$ is generated from a secret stego key and thus does not depend on the cover object or the secret message. Because the positions of changeable elements will be different for different covers, the sender has no control over the process of selecting the submatrix $\boldsymbol{H}$ from $\boldsymbol{D}$. One could, however, impose some structure on the columns in $\boldsymbol{D}$, such as requesting a certain number of ones in each column. Note that, however, introducing any regularity into $\boldsymbol{D}$ is likely to lead to codes with suboptimal performance because, as shown in Section II, random matrices $\boldsymbol{D}$ achieve an average payload that asymptotically equal to the maximal payload, k. On the other hand, it may be worth sacrificing the embedding capacity a little in return for a faster and simpler performance (part of a future effort).

Another possibility to speed up the coding is to use sparse matrices $\boldsymbol{D}$ and $\boldsymbol{H}$. Let us assume that the elements of $\boldsymbol{H}$ are realizations of an i.i.d. random variable τ with range {0,1} and $Prob(\tau=1) = \delta = 1- Prob(\tau=0)$ with $\delta < 1/2$. The smaller the density of ones δ, the faster the Gaussian elimination can be carried out in Steps E4 and E12. Also, allowing $\boldsymbol{H}$ to be sparse opens up new possibilities for solving (3) using solvers for sparse matrices (Section III-D). What needs to be clarified, however, is how the sparseness influences the embedding capacity $q_{\max}(k,\delta)$, which now depends on δ. Figure 4 shows that the capacity stays very close to $q_{\max}(k,1/2)$ till a certain critical value of the density δ is reached. Then, it abruptly falls to zero.

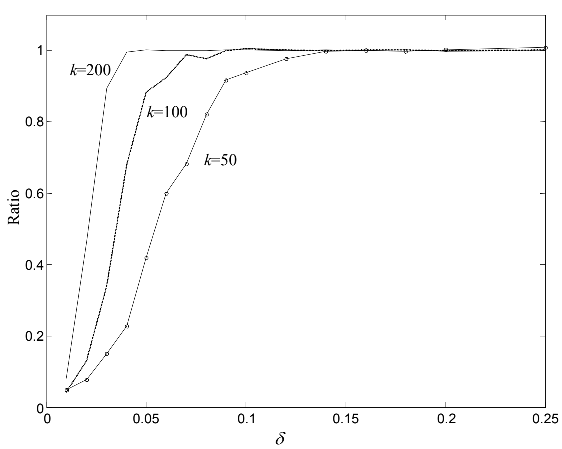

Fig. 4. Ratio $q_{\max}(k,\delta)/q_{\max}(k,1/2)$ as a function of δ for three values of k (averaged over 100 randomly generated matrices).

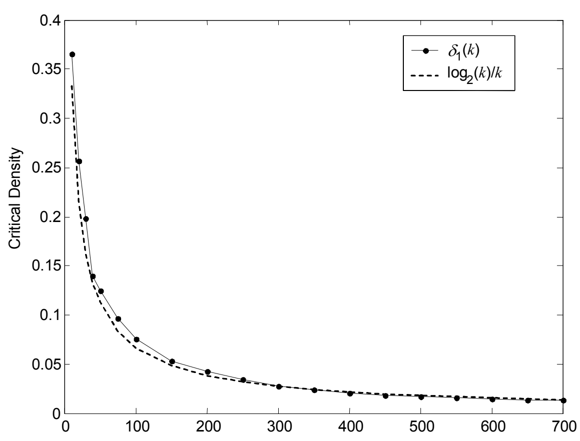

Fig. 5. Comparison of critical density $\delta_1(k)$ and $(\log_2 k)/k$.

According to the result proved by Cooper [28], the probability that a random binary $k\times k$ matrix with density δ is nonsingular tends to $\lim_{k\to\infty} P_{k,k}(k) = 0.2889\ldots$, provided $\delta > (\log_2 k + d(k))/k$ for any $d(k)\to\infty$. Although this result does not tell us how quickly this limit is attained for a given d and is limited to square matrices (thus cannot be used to evaluate $P_{k-r,k}(s)$ in (7)), it suggests that the critical density might be close to $(\log_2 k)/k$. Since this is an asymptotic result, we resorted to computer experiments and determined the value of the density $\delta_1(k)$ for which $1 - q_{\max}(k,\delta_1)/q_{\max}(k,1/2) < 0.01$ for a few small values of k. Figure 5 shows that $(\log_2 k)/k$ is, indeed, a good approximation of $\delta_1(k)$.

1) Using sparse matrices with structured Gaussian elimination

Because in the structured Gaussian elimination, the number of changeable bits k_i in each subset varies, when using sparse matrices we need to guarantee that in each subset B_i the density of 1s in $\boldsymbol{H}^{(i)}$ does not fall below the critical density $\delta(k_i) = (\log_2 k_i)/k_i$ for each i. Recall that the recipient does not know k (and thus k_i) and only knows that $nr_1 \le k \le nr_2$. Because $\delta(k)$ is decreasing, the critical density will be determined by the smallest k_i that one can encounter during embedding. Thus, the sender and recipient will set the density $\delta = (\log_2 k_{\min})/k_{\min}$, where k_{min} is the largest integer for which the probability $\mathrm{Prob}(k_i<k_{\min}) \le p$, where p is a small number (e.g., $p = 0.01$) shared by both communicating parties.

The probability $P(t; n, k, n_i)$ that in a set of n_i randomly selected bits there will be t changeable bits (assuming we are selecting the bits from a set of n bits that contains k changeable bits) is

$$P(t;n,k,n_i) = \frac{\binom{n_i}{t}\binom{n-n_i}{k-t}}{\binom{n}{k}}. \tag{8}$$

Thus, the sender and the recipient determine the density

$$\delta = (\log_2 k_{\min})/k_{\min}, \tag{9}$$

where $k_{\min}$ is the largest integer satisfying the inequality

$$\sum_{t=0}^{k_{\min}} P(t;n,nr_1,\lceil n/\beta \rceil) \le p. \tag{10}$$

C. Communicating the rate r

In applications where the range for $r = k/n$ is very large, it may happen that the actual rate $r \ll r_2$. As a result, there will be more subsets with fewer changeable bits and the header needed to communicate q_i will occupy a relatively larger portion of the embedding capacity q_i. This increases the overhead and decreases the usable embedding capacity. To overcome this problem, we briefly describe a slightly different implementation, in which the sender communicates the rate r, $0 < r < 1$, encoded using u bits, to the receiver, thus effectively narrowing the range $[r_1, r_2]$. The communication of r makes the whole scheme more flexible especially when the covers are very diverse, such as images in raw, JPEG, palette formats, and various audio formats, because the communicating parties do not need to research (and agree on) the typical range of r for all these different cover types.

For a given $r = k/n$, the sender communicates using u bits the integer l for which $r_1 = l2^{-u} \le r < (l+1)2^{-u} = r_2$, $0 \le l < 2^u$. In order to communicate l, Steps E1 and E12 in the pseudo-code for the encoder are modified as follows

E1. Set $\beta = \lceil nr_2/k_{avg} \rceil$. Determine the header size $h = \lceil \log_2(k_{avg}) \rceil + 1$, $q = |\boldsymbol{m}| + \beta h + u$

E12. Select the first n_β columns and $q_\beta - u$ rows from $\boldsymbol{D}$ and denote this submatrix $\boldsymbol{D}^{(\beta)}$. Using a PRNG seeded with the stego key, generate a pseudo-random binary matrix $\bar{\boldsymbol{D}}$ with n columns and u rows with density $\bar{\delta} = 1/2$. Solve q_β equations

$$\begin{bmatrix} \bar{\boldsymbol{H}} \\ \boldsymbol{H}^{(\beta)} \end{bmatrix} \boldsymbol{v} = \begin{bmatrix} bin(l) \\ \boldsymbol{m}^{(\beta)} \end{bmatrix} - \begin{bmatrix} \bar{\boldsymbol{D}}\bar{\boldsymbol{b}} \\ \boldsymbol{D}^{(\beta)}\boldsymbol{b}^{(\beta)} \end{bmatrix}, \qquad (11)$$

for k_β unknowns $\boldsymbol{v}$, where $\bar{\boldsymbol{b}} = (\boldsymbol{b}'^{(1)}, \boldsymbol{b}'^{(2)}, \ldots, \boldsymbol{b}'^{(\beta-1)}, \boldsymbol{b}^{(\beta)})^{\mathrm{T}}$ and $bin(l)$ is a column-wise binary encoding of l. If this system does not have a solution, exit and report failure to embed the message.

In (11), the equations $\boldsymbol{H}^{(\beta)}\boldsymbol{v} = \boldsymbol{m}^{(\beta)} - \boldsymbol{D}^{(\beta)}\boldsymbol{b}^{(\beta)}$ are solved together with u equations $\bar{\boldsymbol{H}}\boldsymbol{v} = bin(l) - \bar{\boldsymbol{D}}\bar{\boldsymbol{b}}$, where $\bar{\boldsymbol{D}}$ is a $u \times n$ pseudo-random matrix with density $\bar{\delta} = 1/2$ and $\bar{\boldsymbol{H}}$ is a $u \times k$ submatrix of $\bar{\boldsymbol{D}}$ consisting of those columns of $\bar{\boldsymbol{D}}$ that correspond to changeable bits.

The decoder will first read u bits $bin(l) = \bar{\boldsymbol{D}}\boldsymbol{b}'$, calculate $r_1 = l2^{-u}$, $r_2 = (l+1)2^{-u}$, $\beta = \lceil nr_2/k_{avg} \rceil$, δ (from (9) and (10)), $h = \lceil \log_2(k_{avg}) \rceil + 1$, and then continues with Steps D0 to D9 for the decoder with a fixed range $[r_1, r_2]$. Note that since the decoder doesn't know β and δ before he decodes u, the matrix $\bar{\boldsymbol{D}}$ must have a fixed number of columns (say n) and a known density, which we set at 1/2.

D. *Other methods for solving linear equations in GF(2)*

In Section III-B, we experimentally showed that $q_{max}(k) \approx k$ holds for random sparse binary matrices $\boldsymbol{H}$ with the density of ones $\delta \ge (\log_2 k)/k$. This fact opens up new possibilities for solving (3) significantly faster using techniques designed for sparse matrices, such as the probabilistic algorithms of Lanczos [29] and Wiedemann [30]. Both methods have complexity proportional to $k(k+\omega k)(\log k)^c$, where ω is the average number of ones in each row of $\boldsymbol{H}$ and c is a small positive constant. We have implemented the Wiedemann and Lanczos methods and the Gaussian elimination and compared their performance on a set of regular square random $k \times k$ matrices with $\delta = (\log_2 k)/k$ for k = 250, …, 30000 (Table I). For each k, several matrices were generated and the running times averaged. All experiments were performed on a high end PC under Cygwin 1.5.9 running under Windows XP Professional. The CPU is Intel Pentium 4 with the HT technology running at 2.4GHz with an 8KB internal cache and a 512KB on-board L2 cache.

Wiedemann method. We implemented the basic algorithm described in the original paper [30]. We obtained the best results for a combined solution in which some parts of the system store their results so that individual elements of the system matrix are arrays of bytes while the most time-critical part (the Berlekamp-Massey algoritm) is implemented by storing elements of the system as bits.

Lanczos method. The implementation is based on the description of the algorithm in [29]. Because the exponentiation and logarithm tables for the finite field (which are jointly used to perform multiplication in the field) are accessed in a completely random way, the computation is much faster if these tables can permanently reside in the cache. The total size of these tables is 8×2^r integer variables when the field GF(2^r) is used (i.e., algebraic extension of degree r of GF(2)), and so the memory requirement is $4 \times 8 \times 2^r$ bytes = 2^{r-5} kilobytes. We obtained the best results for r = 14 and the cache size of 2^9 kilobytes (Table I).

TABLE I
AVERAGE RUNNING TIMES IN SECONDS (THE FASTEST TIMES ARE HIGHLIGHTED IN BOLD FACE, U=UNACHIEVABLE).

k_{avg}	Wiedemann	Lanczos	Gauss	Total solving time $k = 10^5$
250	0.00971	0.00802	**0.00110**	0.440
500	0.0349	0.0322	**0.00499**	0.998
1000	0.134	0.147	**0.023**	2.30
2000	0.542	0.828	**0.122**	6.10
5000	3.74	7.73	**2.74**	54.8
10000	**15.5**	35.4	17.4	174
20000	**72.1**	172	106	530
30000	**U**	371	302	1208

Gaussian elimination. Since the system being solved is binary, we can improve the performance of the Gaussian elimination significantly by storing the elements of the system matrix as bits of the largest integer variable a given architecture/language offers. We implemented the code in C++. Each row of the system is stored as an array of `int`s, where an `int` is 32 bit long on our architecture. Given the fact that we work in GF(2), all the basic operations we need – addition and multiplication – can be very effectively carried out in "parallel" by using the C++ operators ^ (the bitwise eXclusive OR) and & (the bitwise AND), respectively. Because these operators are natively supported by the hardware, we obtain a constant time improvement of 32.

The last column is the total time Steps E4 and E12 contribute to the total embedding time for the structured Gaussian method assuming the bit-stream is divided into β subsets, $\beta = k/k_{avg}$. It was calculated as $T\lceil k/k_{avg} \rceil$, where T is the solving time for the Gaussian elimination method (third column). For $k_{avg} = 30000$, the Wiedemann algorithm could not be run as the problem of this size did not fit into the main memory. Note that neither method leads to practical embedding times if applied to the whole bit-stream ($k = k_{avg}$) for $k \ge 10^4$. On the other hand, the structured Gaussian elimination gives the best performance and practical embedding times for subsets with $k_{avg} = 250$ or 500.

IV. 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.

A. Adaptive steganography

A typical steganographic algorithm for digital media objects (images, audio, video) embeds one bit per object sample (pixel, DCT coefficient, index) by applying an embedding operation to the sample. This embedding operation is applied if the sender needs to adjust the "parity" of the sample to match the message bit. The samples 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 samples 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 samples may become available to the attacker [17]. Also, since the act of embedding itself modifies the image, 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 sample 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). 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 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. Because σ is a function of the 7 MSBs, which are *invariant* with respect to the embedding operation, the recipient will be able to determine the same set of message-carrying pixels as the sender and thus read the message. Note that in this scenario, an attacker also has access to the message-carrying pixels.

Example 2 (Statistics-Preserving LSB Embedding). 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 [31] 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 in [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 how to communicate the selection channel. 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.

B. Perturbed quantization

In this section, we give a short description and analysis of an embedding method called Perturbed Quantization that was previously proposed by the authors of this paper [3]. 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[2]. The sender may

[2] The authors are currently working on a better justification of this heuristic statement using statistical modeling and prove for a certain image model that the embedding is ε-secure in the Cachin's sense [11].

designate such samples (pixels, DCT coefficients) as changeable and use them, together with the wet paper code, for steganography. The remaining samples will be quantized without any changes (those are the wet pixels or "memory defects"). We call this method 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 samples 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) most of which is removed during quantization and is unavailable to the attacker. On the other hand, the attacker may utilize the correlations among neighboring pixels to estimate the unquantized coefficients, thus leaving a small space for possible attacks. Countering this, probably the best the sender can do is to avoid using coefficients whose original (unquantized) values can be predicted by the attacker with sufficient accuracy. However, this again needs to be done with caution as it provides some information about the selection channel to the attacker.

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 $x\in I^m$, 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

$$F = Q \circ T: I^m \rightarrow J^n, \tag{12}$$

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

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 u by coordinates.

$$Q(u) = round(u), \tag{13}$$

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 u_{ij}, $u_{ij}=x_{ij}/2^d$. 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 u_{ij} in a block-by-block manner ($\lceil z\rceil$ denotes the smallest integer larger than or equal to z). In each 8×8 pixel block B_x, the corresponding block B_u in u_{ij} is DCT(B_x)./q, where DCT is the two-dimensional DCT transform, q is the quantization matrix, and the operation "./" is an element-wise division. The quantizer Q is (13).

2) Perturbed quantizer

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

$$C = \{\, i \mid i\in\{0, \ldots, n\},\ u_i\in[L+0.5-\varepsilon, L+0.5+\varepsilon] \text{ for some integer } L\}. \tag{14}$$

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 random message in the cover image X is well modeled with the probabilistic process $X \rightarrow Q_\varepsilon \circ T(X) = Y'$, where Q_ε is the perturbed quantizer and L is an integer,

$$Q_\varepsilon(z) = \begin{cases} L & L \leq z < L+0.5-\varepsilon \\ L+1 & L+0.5+\varepsilon \leq z < L+1 \\ L \text{ or } L+1 & L+0.5-\varepsilon \leq z < L+0.5+\varepsilon, \end{cases} \tag{15}$$

where the third choice occurs with equal probability. The symbol Y' denotes 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 u is a random variable uniformly distributed on $[0, 1]$, the average quantization error $u - Q(u)$ introduced by the scalar quantizer (13) is 1/4, while for the perturbed quantizer 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

$$-2\varepsilon \leq |u - Q(u)| - |u - Q_\varepsilon(u)| \leq 2\varepsilon \text{ for all } u. \tag{16}$$

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 our previous work [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. Because this steganography is discussed in detail in [3], in this paper we discuss it very briefly just to illustrate the point that the wet paper codes enable construction of steganographic techniques that have substantially better steganographic security than previously proposed schemes.

During recompression of a JPEG file, certain values of the DCT coefficients occur in the middle of quantization intervals during the second compression. These coefficients will be the changeable coefficients. Due to the rounding and clipping errors during decompression to the spatial domain, these errors make the second quantization of changeable coefficients resemble a random rather than deterministic process. Thus, the sender can round the changeable coefficients up or down and use 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 of approximately 0.5 bits per non-zero DCT coefficient of the stego file. At the same time, the blind JPEG steganalyzer of [4] was unable to distinguish between purely double compressed images and fully embedded double compressed images [3]. Table II 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) images and tested on 400 never seen images. The methods tested are F5 [25], F5 with matrix embedding (1,1,1) (F5_111), OutGuess 0.2 (OG) [36], Model Based Steganography [37] without and with deblocking (MB1 and MB2, respectively), and the proposed Perturbed Quantization (PQ) [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 $Q = 80$. The PQ algorithm was tested with $Q_1 = 85$ and $Q_2 = 70$. It is very apparent that the new method offers significantly better resistance to steganalysis than other current JPEG steganography techniques.

TABLE II
DETECTION ACCURACY ρ

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

V. Conclusions

First, 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 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]–[17].

Second, we propose a simple variable-rate random linear code for memories with a *large number* of defects and show how it can be applied for our steganographic channel. We prove that this code enables on average communication of k bits given k "dry" elements ($n-k$ defective cells). The code lends itself to efficient practical implementations and offers flexibility and control to the sender over which cover object elements will be modified. This further minimizes the impact of embedding changes (Section III-A).

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 [4], 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 different 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 removal of shrinkage in the F5 algorithm [25] and improving its embedding efficiency. Obviously, nullifying a DCT in F5 embedding coefficient will no longer be a problem for the decoder if the wet paper code is employed. 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 [27]. As the last application, we mention data hiding in binary images proposed by Wu [38]. 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 problem exactly corresponds to writing on wet paper, the capacity of this data hiding method can be dramatically improved.

In the future, we plan to investigate in more detail the steganographic security of Perturbed Quantization. In particular, it seems plausible to prove its ε-security in the Cachin's sense [11] assuming an appropriate model of the cover object. Finally, we plan to further study methods for increasing the embedding efficiency (Section III-A) and simplifying the coding process by imposing structure on matrix H (Section III-B).

APPENDIX A (CALCULATING THE AVERAGE CODING RATE)

Lemma 1. *The function* $\pi(n)=\prod_{i=1}^{n}\left(1-2^{-i}\right)$, *n*≥0, *is monotonically decreasing with* $\lim_{n\to\infty}\pi(n)=\pi(\infty)=0.2889...$

Furthermore, for n≥0,

$$\pi(n)=\pi(\infty)(1+\rho_\pi(n)),\ \text{where}\ \ 0<\rho_\pi(n)<2^{2-n}. \tag{A1}$$

Proof.

$$\begin{aligned}\pi(n)&=\prod_{i=1}^{\infty}\left(1-2^{-i}\right)\prod_{i=n+1}^{\infty}\left(1-2^{-i}\right)^{-1}\\&=\pi(\infty)\exp\left(-\sum_{i=n+1}^{\infty}\ln(1-2^{-i})\right)\\&=\pi(\infty)\exp\left(\sum_{k=1}^{\infty}\frac{1}{k}\sum_{i=n+1}^{\infty}\left(2^{-k}\right)^{-i}\right)\\&=\pi(\infty)\exp\left(\sum_{k=1}^{\infty}\frac{1}{k(2^k-1)2^{kn}}\right)\end{aligned}$$

by expanding the natural logarithm using Taylor expansion and exchanging the sums. Thus, for $n>1$

$$\begin{aligned}\pi(\infty)<\pi(n)&<\pi(\infty)\exp\left(\sum_{k=1}^{\infty}2^{-kn}\right)\\&=\pi(\infty)\exp\left(\frac{1}{2^n-1}\right)<\pi(\infty)\exp\left(2^{1-n}\right)<\pi(\infty)(1+2^{2-n})\end{aligned}$$

and $\pi(\infty)=0.2889...$ by direct calculation. □

Using $\pi(n)$, we rewrite (6), (7), and (8)

$$P_{q,k}(s)=2^{s(q+k-s)-qk}\,\frac{\pi(q)\pi(k)}{\pi(s)\pi(q-s)\pi(k-s)}. \tag{A2}$$

$$\begin{aligned}p_{\geq k-r}&=\sum_{i=0}^{k-r}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}\frac{\pi(k-r)\pi(k)}{\pi(k-r-i)}\\&=\pi(\infty)\sum_{i=0}^{k-r}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}\frac{\left(1+\rho_\pi(k-r)\right)\left(1+\rho_\pi(k)\right)}{1+\rho_\pi(k-r-i)}\end{aligned} \tag{A3}$$

$$\begin{aligned}p_{\geq k+r}&=\frac{1}{2^r}\sum_{i=0}^{k}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}\frac{\pi(k+r)\pi(k)}{\pi(k-i)}\\&=\frac{\pi(\infty)}{2^r}\sum_{i=0}^{k}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}\frac{\left(1+\rho_\pi(k+r)\right)\left(1+\rho_\pi(k)\right)}{1+\rho_\pi(k-i)}\end{aligned} \tag{A4}$$

Lemma 2. *For* $0\leq r\leq k/2$,

$$p_{\geq k-r}=\pi(\infty)\sum_{i=0}^{\infty}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}+\rho_1(r,k), \tag{A5}$$

where $|\rho_1(r,k)|<2^{3-k/4}$.

For $k/2\leq r\leq k$, $1-p_{\geq k-r}<2^{3-k/2}$. (A6)

$$\text{For } r\geq 0,\ p_{\geq k+r}=\frac{\pi(\infty)}{2^r}\sum_{i=0}^{\infty}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}+\rho_2(r,k), \tag{A7}$$

where $|\rho_2(r,k)|<2^{3-k/2}$.

For $r\geq 0$, $p_{\geq k+r}<2^{3-r}$. (A8)

Proof. We first prove (A5). Let

$$Q(a,b)=\pi(\infty)\sum_{i=a}^{b}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}\frac{(1+\rho_\pi(k-r))(1+\rho_\pi(k))}{1+\rho_\pi(k-r-i)}\ \text{and}$$

$Q_0(a,b)=\pi(\infty)\sum_{i=a}^{b}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}$. Using Lemma 1, it is easy to show that for any $0<l<l'<k-r$

$$\begin{aligned}&Q_0(0,l)(1-2^{2-k+r+l})<\frac{1}{1+\rho_\pi(k-r-l)}Q_0(0,l)<Q(0,l)\\&\quad<Q_0(0,l)(1+\rho_\pi(k/2))^2<Q_0(0,l)(1+2^{4-k/2})\end{aligned} \tag{A9}$$

$$\begin{aligned}&0<Q_0(l,l')<Q_0(l,\infty)<2^{1-l^2}\ \text{and}\\&0<Q(l,l')<Q(l,\infty)<2^{1-l^2}.\end{aligned} \tag{A10}$$

Writing $p_{\geq k-r}=Q(0,k/4)+Q(k/4+1,k-r)$ and using (A9) and (A10)

$$\begin{aligned}&(1-2^{2-k/4})\left(Q_0(0,\infty)-Q_0(k/4+1,\infty)\right)\\&\quad<Q(0,k/4)+Q(k/4+1,k-r)\\&\quad<\left(Q_0(0,\infty)-Q_0(k/4+1,\infty)\right)(1+2^{4-k/2})+2^{1-k^2/16}\end{aligned}$$

After applying (A10) to $Q_0(k/4+1,\infty)$ and replacing the bounds with less tight bounds, we finally obtain

$$p_{\geq k-r}=Q(0,k-r)=Q_0(0,\infty)+\rho_1(r,k),$$

where $|\rho_1(r,k)|<2^{3-k/4}$.

To prove (A6), we write

$$\begin{aligned}1-2^{2-k/2}&<1-\rho_\pi(k/2)<p_{\geq k+r}\\&=\frac{\pi(k)}{\pi(r)}+\sum_{i=1}^{\infty}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}\frac{\pi(k-r)\pi(k)}{\pi(k-r-i)}\\&<\frac{1+\rho_\pi(k)}{1+\rho_\pi(r)}+2^{1-r}<1+2^{3-k/2}.\end{aligned}$$

To prove (A7), we define

$$R(a,b)=\pi(\infty)2^{-r}\sum_{i=a}^{b}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}\frac{(1+\rho_\pi(k+r))(1+\rho_\pi(k))}{1+\rho_\pi(k-i)},$$

$$R_0(a,b)=\pi(\infty)2^{-r}\sum_{i=a}^{b}\frac{2^{-i^2-ri-i}}{\pi(i)\pi(i+r)}.$$

Using Lemma 1, we obtain

$$\begin{aligned}(1-2^{2-k/2})R_0(0,k/2) &< \frac{R_0(0,k/2)}{1+\rho_\pi(k/2)} < R(0,k/2)\\ &< R_0(0,k/2)(1+\rho_\pi(k))^2 \\ &< R_0(0,k/2)(1+2^{3-k}),\end{aligned} \tag{A11}$$

$$0 < R_0(k/2+1,k) < 2^{-k^2/4} \quad \text{and} \quad 0 < R(k/2+1,k) < 2^{-k^2/4}. \tag{A12}$$

Writing $p_{\geq k+r} = R(0, k/4) + R(k/4+1, k-r)$ and using (A11) and (A12),

$$(1-2^{2-k/2})\left(R_0(0,\infty)-R_0(k/2+1,\infty)\right) < R(0,k)$$

$$< \left(R_0(0,\infty)-R_0(k/2+1,\infty)\right)(1+2^{3-k})+2^{-k^2/4}.$$

Thus, $p_{\geq k+r} = R(0,k) = R_0(0,\infty)+\rho_2(r,k)$, where $|\rho_2(r,k)| < 2^{3-k/2}$.

The inequality (A8) is easily proved directly from (A4). □

Lemma 3. *$p_{=k-r} = p_{=k+r} + \rho_3(r, k)$, where*

$|\rho_3(r,k)| < 2^{5-k/4}$ *for $0 \leq r \leq k/2$,*

$p_{=i} < 2^{4-k/2}$ *for $0 \leq i < k/2$,*

$p_{=i} < 2^{4-i+k}$ *for $i > k+k/2$.*

Proof. The second and third inequalities are easily proved from (A6) and (A8). The approximate symmetry $p_{=k-r} \cong p_{=k+r}$ for $r \leq k/2$ is proved as follows

$$\begin{aligned}p_{\geq k-r+1} &= \pi(\infty)\sum_{i=0}^{\infty}\frac{2^{-i^2-(r-1)i-i}}{\pi(i)\pi(i+r-1)}+\rho_1(r,k)\\ &= \pi(\infty)\sum_{i=0}^{\infty}\frac{2^{-i^2-ri-i}(2^i-2^{-r})}{\pi(i)\pi(i+r)}+\rho_1(r,k)\\ &= -p_{\geq k+r}+\sum_{i=0}^{\infty}\frac{2^{-i(i+r)}}{\pi(i)\pi(i+r)}+\rho_1(r,k)+\rho_2(r,k)\end{aligned}$$

or

$$p_{\geq k-r+1}+p_{\geq k+r} = \sum_{i=0}^{\infty}\frac{2^{-i(i+r)}}{\pi(i)\pi(i+r)}+\rho_1(r,k)+\rho_2(r,k).$$

Thus, $p_{=k-r} - p_{=k+r} = p_{\geq k-r} + p_{\geq k+r+1} - (p_{\geq k-r+1} + p_{\geq k+r}) =$

$$\sum_{i=0}^{\infty}\frac{2^{-i(i+r)}\left(1-\dfrac{2^{r+1}}{2^{r+i+1}-1}\right)}{\pi(i)\pi(i+r)}+\rho_3(r,k)=\rho_3(r,k)\text{, where}$$

$$\rho_3(r,k)=\rho_1(r,k)+\rho_1(r-1,k)+\rho_2(r,k)+\rho_2(r-1,k)<2^{5-k/4}.$$

We now prove that the sum is indeed equal to zero. After some tedious but straightforward algebra, we can rewrite the sum as

$$\sum_{i=0}^{\infty}\frac{2^i(2^{r+i+1}-1-2^{r+1})}{2^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{2^k}\right)\prod_{k=1}^{i}(2^{k+r+1}-1)(2^k-1)}. \tag{A13}$$

To prove that (A13) is zero, we first prove by induction with respect to n that for r, n positive and $t \neq 0$

$$\begin{aligned}s_{r,n} &= \sum_{i=0}^{n}\frac{t^i(t^{r+i+1}-1-t^{r+1})}{t^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{t^k}\right)\prod_{k=1}^{n}(t^{k+r+1}-1)(t^k-1)}\\ &= -\frac{1}{t^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{t^k}\right)\prod_{k=1}^{n}(t^{k+r+1}-1)(t^k-1)}.\end{aligned} \tag{A14}$$

For $n = 0$ we need to prove that

$$\frac{t^{r+1}-1-t^{r+1}}{t^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{t^k}\right)} = -\frac{1}{t^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{t^k}\right)}\text{, which is true.}$$

Assuming (A14) is true for n, we write

$$\begin{aligned}s_{r,n+1} &= \frac{-1}{t^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{t^k}\right)\prod_{k=1}^{n}(t^{k+r+1}-1)(t^k-1)}+\\ &\quad+\frac{t^{n+1}(t^{r+n+2}-1-t^{r+1})}{t^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{t^k}\right)\prod_{k=1}^{n+1}(t^{k+r+1}-1)(t^k-1)}\\ &= \frac{-(t^{n+r+2}-1)(t^{n+1}-1)+t^{n+1}(t^{r+n+2}-1-t^{r+1})}{t^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{t^k}\right)\prod_{k=1}^{n+1}(t^{k+r+1}-1)(t^k-1)}\\ &= -\frac{1}{t^{r+1}\prod_{k=1}^{r+1}\left(1-\frac{1}{t^k}\right)\prod_{k=1}^{n+1}(t^{k+r+1}-1)(t^k-1)},\end{aligned}$$

which completes the induction step.

The fact that (A13) is zero is now easily proved by observing that for a fixed r and $t = 2$, $\lim_{n\to\infty} s_{r,n} = 0$. □

Lemma 4. $q_{\max}(k) = \sum_{i=1}^{\infty} ip_{=i} = \sum_{i=1}^{\infty} i(p_{\geq i}-p_{\geq i+1}) = k+\rho_4(k)$, *where* $|\rho_4(k)| < k^2 2^{8-k/4}$.

Proof. $$\sum_{i=1}^{\infty} ip_{=i} = \sum_{i=0}^{k/2} ip_{=i} + \sum_{i=k/2+1}^{k+k/2} ip_{=i} + \sum_{i=k+k/2+1}^{\infty} ip_{=i}.$$

Using Lemma 3, $\sum_{i=0}^{k/2} ip_{=i} < \frac{k(k/2+1)}{2}2^{4-k/2} < k^2 2^{2-k/2}$,

$$\begin{aligned}\sum_{i=k+k/2+1}^{\infty} ip_{=i} &< \sum_{i=k+k/2+1}^{\infty} i2^{4-i+k}\\ &< 2^{4+k}\frac{2^{-k-k/2-1}}{1-1/2}\left(k+k/2+1+1\right) < k2^{5-k/2}\end{aligned}$$

because

$$\sum_{i=a}^{b} iq^i \leq \frac{q^a}{1-q}\left(a+\frac{q}{1-q}\right) \text{ for any } 0 < a < b,\ 0 \leq q < 1.$$

Also, $\sum_{i=k/2+1}^{k+k/2} ip_{=i} = k + \rho(k), \quad \text{where } \rho(k) < k^2 2^{6-k/4}$.

Adding all three sums together, we obtain $q_{\max}(k) = k + \rho_4(k)$, where $\rho_4(k) < k^2 2^{8-k/4}$. □

Acknowledgment

Special thanks belong to Pierre Moulin and Ralf Koetter for many useful discussions.

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. Inform. Theory* **29**, pp. 731–739, 1983.

[3] J. Fridrich, M. Goljan, and D. Soukal, "Perturbed Quantization Steganography with Wet Paper Codes", *Proc. ACM Multimedia 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: Fridrich, J. (ed.): 6th International Workshop on Information Hiding. *Lecture Notes in Computer Science*, vol. 3200. Springer-Verlag, New York, pp. 67–81, 2005.

[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, editors, *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: Aucsmith, D. (ed.): 2nd International Workshop on Information Hiding. *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, G. Wolf, "Modeling the Security of Steganographic Systems", In: Aucsmith, D. (ed.): 2nd International Workshop on Information Hiding. *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", *Proc. SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents IV*, vol. 4675, Electronic Imaging, San Jose, pp. 50–56, 2002.

[14] E. Franz, "Steganography Preserving Statistical Properties", In: Petitcolas, F.A.P. (ed.): 5th International Workshop on Information Hiding. *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: Pfitzmann A. (ed.): 2nd International Workshop on Information Hiding. *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 Multimediam Workshop*, Magdeburg, Germany, September 20–21, pp. 16–24, 2004.

[17] A. Westfeld and R. Böhme, "Exploiting Preserved Statistics for Steganalysis", In: Fridrich, J. (ed.): 6th International Workshop on Information Hiding. *Lecture Notes in Computer Science*, vol. 3200. Springer-Verlag, New York, pp. 82–96, 2005.

[18] 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.

[19] R. Zamir, S. Shamai, U. Erez, "Nested Linear/Lattice Codes for Structured Multiterminal Binning", IEEE *Trans. Inf. Th*. **48**(6), pp. 1250–1276, 2002.

[20] F. Fu and R. Yeung, "On the Capacity and Error-Correcting Codes of Write-Efficient Memories," *IEEE Trans. Inform. Theory* **46**, pp. 2299–2314, 2000.

[21] G. Cohen, "Applications of Coding Theory to Communication Combinatorial Problems, *Discrete Math.* **83** (2–3), pp. 237–248, 1990.

[22] B. S. Tsybakov, "Defect and Error Correction," *Probl. Peredach. Inform.* **11**, pp. 21–30, July–Sept. Translated from Russian, 1975.

[23] C. Heegard, "Partitioned Linear Block Codes for Computer Memory with 'Stuck-at' Defects," *IEEE Trans. Inform. Theory* **29**, pp. 831–842, 1983.

[24] R. Crandall, "Some Notes on Steganography", posted on Steganography Mailing List, http://os.inf.tu-dresden.de/ ~westfeld/crandall.pdf, 1998.

[25] A. Westfeld, "High Capacity Despite Better Steganalysis (F5–A Steganographic Algorithm)", In: Moskowitz, I.S. (ed.): 4th International Workshop on Information Hiding. *Lecture Notes in Computer Science*, vol. 2137. Springer-Verlag, New York, pp. 289–302, 2001.

[26] R.P. Brent, S. Gao, A.G.B. Lauder, "Random Krylov Spaces Over Finite Fields", *SIAM J. Discrete Math*. **16**(2), pp. 276–287, 2003.

[27] J. Fridrich, M. Goljan, and D. Soukal, "Searching for the Stego Key", *Proc. SPIE Electronic Imaging*, *Security, Steganography, and Watermarking of Multimedia Contents VI*, San Jose, 2004.

[28] C. Cooper, "On the Rank of Random Matrices", *Random Structures and Algorithms* **16**(2), pp. 209–232, 2000.

[29] B.A. LaMacchia and A.M. Odlyzko, "Solving Large Sparse Linear Systems over Finite Fields", In: Menezes, A.J., and Vanstone, S.A. (eds.): Advances in Cryptology – CRYPTO '90, Springer Verlag, *Lecture Notes in Computer Science*, vol. 537, pp. 109–133, 1991.

[30] D.H. Wiedemann, "Solving Sparse Linear Equations Over Finite Fields", *IEEE Trans. on Inf. Th.* **32**(1), pp. 54–62, 1986.

[31] A. Westfeld and A. Pfitzmann, "Attacks on Steganographic Systems", In: Pfitzmann A. (eds.): 3rd International Workshop on Information Hiding. *Lecture Notes in Computer Science*, vol.1768. Springer-Verlag, New York, pp. 61–75, 2000.

[32] H. Farid and L. Siwei, "Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines", In: Petitcolas, F.A.P. (ed.): 5th International Workshop on Information Hiding. *Lecture Notes in Computer Science*, vol. 2578. Springer-Verlag, New York, pp. 340–354, 2002.

[33] I. Avcibas, N. Memon, and B. Sankur, "Steganalysis using Image Quality Metrics", *Proc. SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents III*, San Jose, pp. 523–531, 2001.

[34] R. Tzschoppe, R. Bäuml, J.B. Huber, and A. Kaup, "Steganographic System Based on Higher-Order Statistics", *Proc. SPIE Electronic Imaging*, *Security and Watermarking of Multimedia Contents V*, Santa Clara, pp. 156–166, 2003.

[35] J. Eggers, R. Bäuml, and B. Girod, "A Communications Approach to Steganography", *Proc. SPIE Electronic Imaging*, *Security and Watermarking of Multimedia Contents IV*, Santa Clara, pp. 26–49, 2002.

[36] N. Provos, "Defending Against Statistical Steganalysis", 10th USENIX Security Symposium. Washington, DC 2001.

[37] P. Sallee, "Model Based Steganography", In: T. Kalker, I.J. Cox, Yong Man Ro (Eds.), International Workshop on Digital Watermarking. *Lecture Notes in Computer Science*, vol. 2939. Springer Verlag, New York, pp. 154–167, 2004.

[38] M. Wu, Tang E., and Liu B., "Data Hiding in Digital Binary Image", *Proc. Conf. on Multimedia & Expo* (ICME'00), New York City, 2000.