WET ZZW CONSTRUCTION FOR STEGANOGRAPHY

Tomáš Filler and Jessica Fridrich

Department of ECE, SUNY Binghamton, NY, USA
{tomas.filler, fridrich}@binghamton.edu

ABSTRACT

Wet paper codes are an essential tool for communication with non-shared selection channels. Inspired by the recent ZZW construction for matrix embedding [11], we propose a novel wet paper coding scheme with high embedding efficiency. The performance is analyzed under the assumption that wet cover elements form an i.i.d. Bernoulli sequence. Attention is paid to implementation details to minimize capacity loss in practice.

1. INTRODUCTION

In steganography, the sender communicates with the receiver by hiding her messages in generally trusted media, such as digital images, so that it is hard to distinguish between the original (cover) objects and stego objects carrying messages. Usually, the message is embedded in the cover image by slightly modifying colors of selected pixels (selection channel). If the selection channel is not completely shared between the sender and the recipient, we speak of a non-shared selection channel, in which k out of n cover elements are allowed to be changed (*dry* pixels) while the rest are not to be modified during embedding (*wet* pixels). In the binary case, Wet Paper Codes [7] (WPC) can communicate up to k bits as a syndrome of a binary linear code. Many steganographic algorithms use WPCs as their design element [7, 9].

When the payload, m, is smaller than the number of dry elements, $m < k$, non-optimized WPCs [7] would make on average $R_\mathrm{a} = m/2$ embedding changes, leading to embedding efficiency $e = 2$ bits per change. It is also known [6] that the embedding efficiency of any steganographic scheme that embeds m bits using on average R_a embedding changes must satisfy $e \triangleq m/R_\mathrm{a} \leq \alpha/H^{-1}(\alpha)$, where $\alpha \triangleq m/k$ is the relative message length (w.r.t. dry elements), $H^{-1}(x)$ is the inverse of the binary entropy function $H(x) = -x \lg x - (x-1)\lg(x-1)$ on $x \in [0, 0.5]$, and $\lg$ is logarithm at the base 2. To the best of our knowledge, the embedding efficiency of existing WPCs [6, 8, 10] is still far from the bound.

In the special case when all cover elements are dry, there exist numerous so called *matrix embedding* methods approaching the bound, see e.g., [4, 11]. The ZZW construction [11] starts with an (n, m, R_a) code C able to embed m bits into n cover elements using on average R_a changes, and provides a family of $(n2^p, m + pR_\mathrm{a}, R_\mathrm{a})$ codes C_p, $p \geq 0$, that follow the bound as $\alpha \to 0$ (or $p \to \infty$) [3]. This construction is important for steganography because the relative payload must decrease with increasing size of the cover object in order to maintain the same level of security [2].

In this paper, we propose "wet ZZW construction" for building WPCs with high embedding efficiency with similar properties as ZZW. For small payloads $\alpha < \frac{1}{4}$, the family of WPCs obtained from the simplest form of the construction outperforms all known WPCs in terms of embedding efficiency. The wet ZZW construction can be extended to ternary codes (useful when embedding changes are bounded by 1) using the approach from [12].

The construction is described in Section 2 and its performance is analyzed in Section 3. Section 4 describes a specific implementation of WPCs that maximize the payload while minimizing the overhead. These WPCs are necessary for the wet ZZW construction and are used in Section 5 to build practical WPCs with high embedding efficiency. The paper is concluded in Section 6.

2. THE WET ZZW CONSTRUCTION

This section describes the encoding and decoding algorithms of codes obtained via the wet ZZW construction. For a given WPC C_t able to embed m_t bits into k_t dry elements out of total n_t elements by R_a changes on average, we construct WPCs, C_p, $p \geq 1$, with decreasing relative payload α_p and increasing embedding efficiency e_p.

Let $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n) \in \{0,1\}^n$ be a binary representation of the cover, $\mathcal{S} \subset \{1, \ldots, n\}$ the selection channel, $|\mathcal{S}| = k$, and $\{\mathbf{x}_j | j \in \mathcal{S}\}$ the set of dry cover elements. The output of the encoder is a binary vector $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n) \in \{0,1\}^n$ representing the stego object that conveys the message $\mathbf{m} \in \{0,1\}^m$ and satisfies $\mathbf{y}_j = \mathbf{x}_j, \forall j \notin \mathcal{S}$. We reserve the symbol $\oplus$ for bitwise eXclusive OR (XOR).

The work on this paper was supported by Air Force Office of Scientific Research under the research grant number FA9550-08-1-0084. 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 AFOSR or the U.S. Government.

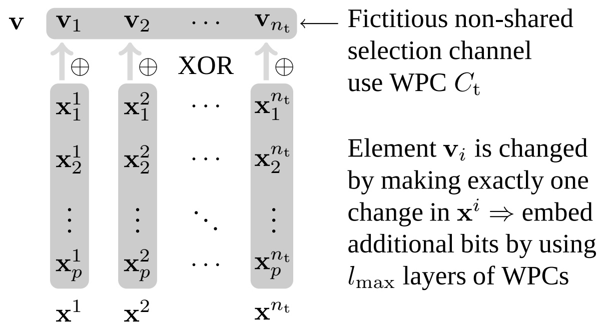
Encoding algorithm: Given a WPC C_t, integer $p \geq 1$, message $\mathbf{m}$, and set $\mathcal{S}$, the encoder processes the cover vector $\mathbf{x} \in \{0,1\}^{n_\mathrm{t}p}$ as follows. First, the cover is divided into n_t columns of p bits, $\{\mathbf{x}^i\}_{i=1}^{n_\mathrm{t}}$, $\mathbf{x}^i = (\mathbf{x}_1^i, \ldots, \mathbf{x}_p^i)^T = (\mathbf{x}_{(i-1)p+1}, \ldots, \mathbf{x}_{ip})^T$. Then, the vector $\mathbf{v} = (\mathbf{v}_1, \ldots, \mathbf{v}_{n_\mathrm{t}})$ is calculated as XOR of all n_t columns (see Figure 1),$\mathbf{v}_i = \oplus_{j=1}^p \mathbf{x}_j^i$. We say that $\mathbf{v}_i$ is dry if at least one element $\mathbf{x}_j^i$, $j = 1, \ldots, p$, is dry, otherwise it is wet. Let $\mathbf{v}$ contain k_t dry elements out of n_t. Think of $\mathbf{v}$ as some fictitious non-shared selection channel and embed m_t bits in it by making on average R_a changes. Every element $\mathbf{v}_i$ that needs to be changed will be changed by making exactly one change in $\mathbf{x}^i$. If the column $\mathbf{x}^i$ contains d dry elements, we can embed additional bits (up to $\lg d$) by choosing the element for the final change. The algorithm works in an iterative fashion.

Let $\mathcal{T} \subset \{1, \ldots, n_\mathrm{t}\}$ be the set of all indices i for which $\mathbf{v}_i$ needs to be changed. For $i \in \mathcal{T}$, define $\mathcal{D}_i = \{1 \leq j \leq p \,|\, \mathbf{x}_j^i \text{ is dry}\}$, the set of indices of all dry elements in the ith column that can be used for changing $\mathbf{v}_i$. The following $l_{\max} = \lceil \lg p \rceil$ iterations are used to narrow the sets $\mathcal{D}_i^1 = \mathcal{D}_i$ to $\mathcal{D}_i^{l_{\max}}$ containing just one element that must be changed to embed the payload and thus transform $\mathbf{x}$ to $\mathbf{y}$.

In the lth iteration, $1 \leq l \leq l_{\max}$, the binary vector $\mathbf{h}^l = (\mathbf{h}_1^l, \ldots, \mathbf{h}_p^l)$, $\mathbf{h}_j^l = \lfloor (j-1)/2^{l_{\max}-l} \rfloor \bmod 2$ is used to narrow down the sets $\mathcal{D}_i^l$ and embed additional m_l bits as follows. Define the lth channel $\mathbf{z}^l = (\mathbf{z}_1^l, \ldots, \mathbf{z}_{n_\mathrm{t}}^l)$ as $\mathbf{z}_i^l = \oplus_{j=1}^p \mathbf{h}_j^l \mathbf{x}_j^i$. Let $\mathcal{D}_i^l = \mathcal{D}_i^l(0) \cup \mathcal{D}_i^l(1)$, where $\mathcal{D}_i^l(x) = \{j \in \mathcal{D}_i^l | \mathbf{h}_j^l = x\}$. The element $\mathbf{z}_i^l$ is dry if and only if $i \in \mathcal{T}$ and $\mathcal{D}_i^l(0) \neq \emptyset$ and $\mathcal{D}_i^l(1) \neq \emptyset$. In all other cases, $\mathbf{z}_i^l$ is wet and fixed to its original value with the exception when $i \in \mathcal{T}$ and $\mathcal{D}_i^l(1) \neq \emptyset$, $\mathcal{D}_i^l(0) = \emptyset$, in which case $\mathbf{z}_i^l \leftarrow 1 - \mathbf{z}_i^l$ because one dry element in $\mathcal{D}_i^l(1)$ will be changed later. Assuming there are d_l dry elements in $\mathbf{z}^l$, we can embed up to $m_l \leq d_l$ bits on average by using the maximum capacity WPCs W_l realized by [5, 7], obtaining thus the final values $\hat{\mathbf{z}}_i^l$ for all dry elements $\mathbf{z}_i^l$. The sets $\mathcal{D}_i^l$ are narrowed to $\mathcal{D}_i^{l+1} = \mathcal{D}_i^l(|\hat{\mathbf{z}}_i^l - \mathbf{z}_i^l|)$, for all dry elements, and $\mathcal{D}_i^{l+1} = \mathcal{D}_i^l$ for all wet $\mathbf{z}_i^l$.

In summary, one message bit is embedded in every column in lth iteration, whenever the vector $\mathbf{h}^l$ can be used to distinguish two different dry elements in $\mathbf{x}^i$. This way, the encoding algorithm can on average embed up to $m_\mathrm{t} + d_1 + \cdots + d_{l_{\max}}$ bits with R_a changes (there is exactly one change in every column $\mathbf{x}^i$, $i \in \mathcal{T}$). This bound on the payload is the most general result we can obtain without making any assumptions about the selection channel over $\mathbf{x}$. For a given selection channel and p, we denote the expected number of message bits as m_p and define the *relative message length* (w.r.t. dry elements) as $\alpha_p \triangleq m_p/k$ and *embedding efficiency* as $e_p \triangleq m_p/R_\mathrm{a}$.

Decoding algorithm: Given a WPC C_t and integer $p \geq 1$, the decoder processes the stego vector $\mathbf{y} \in \{0,1\}^{n_\mathrm{t}p}$ and outputs the original message $\mathbf{m}$ as follows. First, by forming the same columns $\{\mathbf{y}^i\}_{i=1}^{n_\mathrm{t}}$, $\mathbf{y}^i = (\mathbf{y}_1^i, \ldots, \mathbf{y}_p^i)^T = (\mathbf{y}_{(i-1)p+1}, \ldots, \mathbf{y}_{ip})^T$, the first m_t bits of the message are extracted by applying C_t to the vector $\mathbf{v} = (\mathbf{v}_1, \ldots, \mathbf{v}_{n_\mathrm{t}})$ of XORed columns, $\mathbf{v}_i = \oplus_{j=1}^p \mathbf{y}_j^i$. The remaining $l_{\max}$ parts of the message are extracted through $l_{\max}$ iterations as follows.



v
$\mathbf{v}_1$
$\mathbf{v}_2$
$\cdots$
$\mathbf{v}_{n_\mathrm{t}}$
Fictitious non-shared selection channel use WPC C_t
$\oplus$
XOR
$\mathbf{x}_1^1$
$\mathbf{x}_1^2$
$\mathbf{x}_1^{n_\mathrm{t}}$
$\mathbf{x}_2^1$
$\mathbf{x}_2^2$
$\mathbf{x}_2^{n_\mathrm{t}}$
$\mathbf{x}_p^1$
$\mathbf{x}_p^2$
$\mathbf{x}_p^{n_\mathrm{t}}$
$\mathbf{x}^1$
$\mathbf{x}^2$
$\mathbf{x}^{n_\mathrm{t}}$
Element $\mathbf{v}_i$ is changed by making exactly one change in $\mathbf{x}^i \Rightarrow$ embed additional bits by using $l_{\max}$ layers of WPCs

Fig. 1. Block of $n_\mathrm{t}p$ elements in wet ZZW construction.

Similar as in the encoding algorithm, in the lth iteration, $1 \leq l \leq l_{\max}$, the vector $\mathbf{z}^l = (\mathbf{z}_1^l, \ldots, \mathbf{z}_{n_\mathrm{t}}^l)$ is computed as $\mathbf{z}_i^l = \oplus_{j=1}^p \mathbf{h}_j^l \mathbf{y}_j^i$. The lth part of the message is extracted using the decoder from WPC W_l used in the encoding part applied to vector $\mathbf{z}^l$. Here, we assume that the decoder knows the number of dry elements d_l. This assumption will enable us to obtain an upper bound on the performance of the wet ZZW construction. Later, we remove this assumption.

For $p = 1$, the resulting WPC is the same as C_t. If all cover elements are dry and p is in the form $p = 2^r$ for some integer r, the wet ZZW construction reduces to the original ZZW construction [11].

3. ANALYSIS OF THE WET ZZW CONSTRUCTION

In this section, we study the embedding efficiency of codes obtained using the wet ZZW construction. First, in Section 3.1 we determine the efficiency without considering the communication overhead. Section 3.2 shows that the efficiency decreases surprisingly quickly with increasing overhead, which motivates our work on minimizing the overhead in Section 4.

To compute the number of message bits and obtain the embedding efficiency, we need to accept some assumption about the non-shared selection channel over $\mathbf{x}$. It is a common practice to permute cover elements before embedding using a pseudo-random permutation obtained from a stego key. The permutation breaks dependencies between cover elements, justifying the following assumption.

Assumption 1: The non-shared selection channel over $\mathbf{x}$ is realized as an i.i.d. Bernoulli(ρ) process, $Pr(\mathbf{x}_i \text{ is dry}) = \rho \triangleq k/n$.

Consequently, the non-shared selection channel over $\mathbf{v}$ is also i.i.d. with $Pr(\mathbf{v}_i \text{ is dry}) = 1 - (1-\rho)^p$. Thus, the "top" channel is becoming dry exponentially fast. By this and for a suitably large p, the WPC C_t does not even need to be a

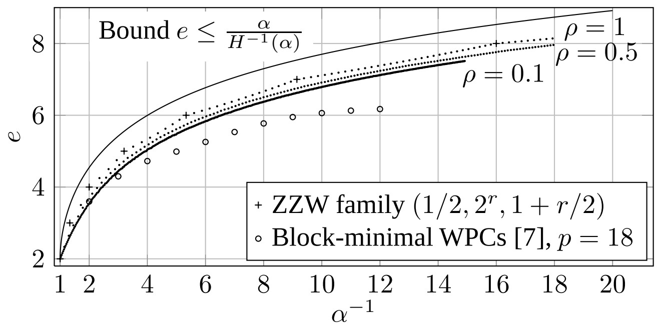


Fig. 2. Embedding efficiency (1) of wet ZZW WPCs in the ideal case of zero overhead for several values of dry element density ρ. Here, C_{t} are maximum-payload WPCs and $n = 10^6$. The performance is compared with block-minimal WPCs [6] with codimension $p = 18$. The ZZW matrix embedding family $(1/2, 2^r, 1 + r/2)$ is shown for reference.

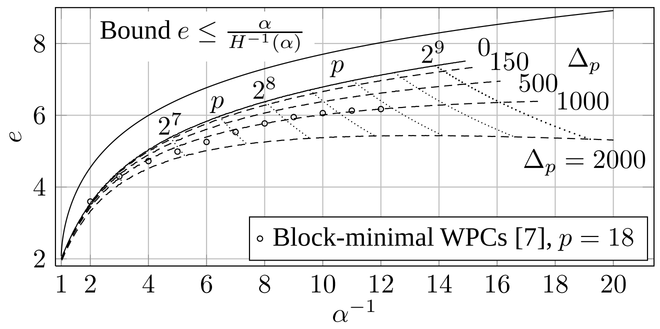


Fig. 3. Impact of overhead size Δ_p for $1 \le p \le 560$ on embedding efficiency (2) of wet ZZW WPCs. Here, C_{t} are maximum-payload WPCs and $n = 10^6$. The density of dry elements is $\rho = 0.1$.

WPC, giving us the option to use ordinary syndrome codes. However, care needs to be taken since $\mathbf{v}$ could contain some wet elements in practice.

3.1. Impact of wet elements on embedding efficiency

In the ZZW construction, since all elements in each column $\mathbf{x}^i$ of length $p = 2^r$ are dry, we can embed $\lg 2^r = r$ bits per column by making one change. If some elements in $\mathbf{x}^i$ are wet and the column contains g dry elements, the number of embeddable bits is bounded by $\lg g$. From this point of view, there is no loss in having wet elements in $\mathbf{x}^i$, because α and e are calculated w.r.t. dry elements. In practice, however, the embedding efficiency decreases when wet elements are present.

The first source of loss is due to the random nature of dry elements. From Assumption 1, the number of dry elements in column $\mathbf{x}^i$, g_i, is i.i.d. binomial with parameters p and ρ. For example, if g_i only attains two values, $g_i \in \{\gamma_1, \gamma_2\}$, the code resulting from the wet ZZW construction is a blockwise direct sum[1] of two codes for which $g_i = \gamma_1$ and $g_i = \gamma_2$, $\forall i$, respectively. The embedding efficiency of the resulting code is a convex combination of efficiencies of the individual codes, which is lower than the efficiency of the code designed for the same rate with a fixed number of dry elements in column $\mathbf{x}^i$.

The second source of loss is due to the selection of the last dry element in column $\mathbf{x}^i$. To avoid any loss and embed the maximum number of bits, the set $\mathcal{D}_i^l$ should be narrowed to $\mathcal{D}_i^l(x)$ at a dry $\mathbf{z}_i^l$ with probability proportional to $\left|\mathcal{D}_i^l(x)\right|$, for $x \in \{0, 1\}$. However, $Pr(\mathcal{D}_i^{l+1} = \mathcal{D}_i^l(1)) = Pr(\left|\hat{\mathbf{z}}_i^l - \mathbf{z}_i^l\right| = 1) = 1/2$ because the solution $\mathbf{z}_i^l$ obtained from WPC W_l is equally likely to be 0 or 1. Thus, the encoder is suboptimal. One reason for this is that the decoder does not have access to sets $\mathcal{D}_i^l$ and thus is not able to utilize this information.

[1]Code $C = \{(\mathbf{c}_1, \mathbf{c}_2) | \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2\}$ is blockwise direct sum of codes C_1 and C_2.

Figure 2 shows the embedding efficiency of codes obtained by the wet ZZW construction when the WPC C_{t} was chosen to embed maximal possible payload (denote this code by C_{MAX}), e.g., WPCs from [5, 7] which embed k_{t} bits into k_{t} dry elements while making $k_{\text{t}}/2$ changes on average. The non-shared selection channels were assumed to follow Assumption 1 with density of dry elements ρ. For $n = n_{\text{t}}p = 10^6$ cover elements and integer $p \ge 1$, the average number of bits, m_p, the encoder can embed, is obtained as follows. Define the matrix $\mathbb{C} = (c_{ij})$, where c_{ij} is the average number of random bits the encoder can embed by changing exactly one dry element in a random vector of length i with $0 \le j \le i$ dry elements. The matrix $\mathbb{C}$ was obtained experimentally as a mean over 1000 random samples. Define $k_{\text{t}}(p) = n_{\text{t}}(1 - (1 - \rho)^p)$ as the average number of dry elements in $\mathbf{v}$ and

$$m_{\text{c}}(p) = \sum_{j=1}^{p} \binom{p}{j} \frac{\rho^j (1 - \rho)^{p-j}}{1 - (1 - \rho)^p} c_{pj}$$

as the average number of bits the encoder can embed by changing exactly one dry element in one column of length p. Here, the average c_{pj} was computed w.r.t. the positive binomial distribution. Finally, $m_p = k_{\text{t}}(p) + k_{\text{t}}(p) m_{\text{c}}(p)/2$,

$$\alpha_p = \frac{m_p}{n\rho}, \qquad e_p = \frac{2m_p}{k_{\text{t}}(p)}. \qquad (1)$$

The final and most severe source of loss is related to the practical realization of $l_{\max}$ layers of WPCs W_l and is studied in the following section.

3.2. Loss analysis due to WPC's overhead

In order to implement $l_{\max}$ layers of WPCs in the encoding algorithm, we need $l_{\max} = \lceil \lg p \rceil$ WPCs $\{W_l\}_{l=1}^{l_{\max}}$ where we can embed the maximum possible payload. Here, the embedding efficiency is of no concern because we have to make

$R_{\rm a}$ changes anyway and the bits embedded while choosing the place for them are thus "for free." The implementation of these codes can be based either on random codes [7] with cubic complexity or LT codes [5] with log-linear complexity. LT codes are preferable for their complexity, but due to the sparse structure of the LT matrix, the number of bits that can be communicated is decreased by θ percent of dry elements. Although the overhead θ tends to zero as the number of dry elements increases to infinity, the loss for finite lengths is not negligible. For example, the overhead of an LT code designed for 10^4 message bits with parameters $c = 0.1$ and $\delta = 5$ is about 6%, while the probability that the code will succeed in embedding is 0.75.

Let $m_p = m_{\rm t} + \sum_{l=1}^{l_{\max}} d_l$ be the total number of bits the encoder can embed assuming the WPCs W_l do not have any overhead for given $p \geq 1$. This number of bits cannot be achieved in practice, since the parameters of W_l (the number of dry bits d_l) are not known to the decoder. For this reason, there must be some loss, say Δ_p bits in total, which should be as small as possible. Figure 3 shows the loss in embedding efficiency when $C_{\rm t} = C_{\rm MAX}$, $n = n_{\rm t}p = 10^6$, $\rho = 0.1$, and $\Delta_p = \{0, \ldots, 2000\}$. The results were obtained experimentally in the same way as in Figure 2. Here,

$$\alpha_p = \frac{m_p - \Delta_p}{n\rho}, \qquad e_p = \frac{m_p - \Delta_p}{k_{\rm t}(p)/2}. \qquad (2)$$

4. WPCS WITH SMALL OVERHEAD

This section introduces practical WPCs designed for maximum payload with minimum possible overhead. Such codes are needed to realize the WPCs W_l in the wet ZZW construction. For a given cover $\mathbf{x} \in \{0,1\}^n$, selection channel $\mathcal{S} \subset \{1, \ldots, n\}$, $|\mathcal{S}| = k$, and shared binary matrix $\mathbb{D} \in \{0,1\}^{m \times n}$, the WPC encoder outputs the stego vector $\mathbf{y} \in \{0,1\}^n$ such that $\mathbb{D}\mathbf{y} = \mathbf{m}$ and $\mathbf{y}_i = \mathbf{x}_i$, $\forall i \notin \mathcal{S}$, where $\mathbf{m} \in \{0,1\}^m$ is the desired message. Let $\mathbb{H} \in \{0,1\}^{m \times k}$ be a binary matrix obtained from $\mathbb{D}$ by removing all columns i, $i \notin \mathcal{S}$. If $\mathbb{H}$ has full rank (in binary arithmetic), $\mathbf{y}$ can be found by Gaussian elimination with complexity $\mathcal{O}(m^2 k)$ [7].

Random codes proposed in [7] are suitable for small overhead because an $m \times k$ random binary matrix $\mathbb{H} = (h_{ij})$, $Pr(h_{ij} = 1) = 0.5$, is of full rank with high probability even for small overhead $k - m$ (see Figure 1 in [1]). With $k \to \infty$, the probability that a random $k \times k$ matrix has full rank approaches $0.28878...$ and the average number of extra random columns needed to bring the matrix to full rank is $1.6066...$. Thus, the average absolute overhead is about 2 bits (c.f. with the large overhead of LT codes). However, the high computational complexity of Gaussian elimination limits the usage of random codes to small payloads $m \lessapprox 10^4$.

To achieve small overhead even for larger message lengths with low complexity, we endow matrix $\mathbb{D}$ with the following form

$$\mathbb{D} = \begin{pmatrix} \mathbb{D}_1 \\ \mathbb{D}_2 \\ \mathbb{D}_3 \end{pmatrix},$$

where $\mathbb{D}_2 \in \{0,1\}^{m_1 \times n}$ is a random sparse LT matrix, and $\mathbb{D}_1 \in \{0,1\}^{h \times n}$, $\mathbb{D}_3 \in \{0,1\}^{m_2 \times n}$ are random binary matrices. When solving $\mathbb{D}\mathbf{y} = \mathbf{m}$, we can substitute the known wet values $\mathbf{y}_j = \mathbf{x}_j$, $\forall j \notin \mathcal{S}$, move them to the right hand side (RHS), and remove from $\mathbb{D}_i$ all columns j, $j \notin \mathcal{S}$, obtaining thus submatrices $\mathbb{H}_i$, $i \in \{1,2,3\}$. The syndrome $\mathbb{D}_1\mathbf{y}$ communicates $h \leq 30$ bits that encode m_1 and m_2 needed to form $\mathbb{D}_2$ and $\mathbb{D}_3$ at the decoder. The matrix $\mathbb{D}_1$ is generated using the stego key in a pseudo-random fashion (h is shared). The syndromes $\mathbb{D}_2\mathbf{y}$ and $\mathbb{D}_3\mathbf{y}$ communicate the payload. The columns of $\mathbb{D}_2$ are sparse random binary vectors whose Hamming weight follows the robust soliton distribution [5] with parameters c and δ. The parameter m_1 is chosen so that $\mathbb{H}_2$ has full rank with high probability, i.e., the LT code is designed to have sufficient absolute overhead, say g bits. Finally, the random rows of $\mathbb{D}_3$ are added "on the fly" while running the Gaussian elimination on $\mathbb{D}$ in order to maximize m_2 while still having full rank of $\mathbb{D}$ (see below).

We now describe how to efficiently carry out Gaussian elimination with matrix $\mathbb{H}$ obtained from $\mathbb{D}$ and we supply the missing details how the sender generates $\mathbb{D}_3$. First, run the "LT process" [5] of complexity $\mathcal{O}(m_1 \log(m_1/\delta))$ over $\mathbb{H}_2$, but perform column and row permutations over the whole $\mathbb{H}$. This brings $\mathbb{H}_2$ to the upper diagonal form. Run Gaussian elimination on $\mathbb{H}_1$ and bring it to the form $[\mathbb{Z}, \mathbb{T}, \mathbb{G}]$, where $\mathbb{Z}$ is a zero $h \times m_1$ matrix, $\mathbb{T}$ is $h \times h$ upper diagonal, and $\mathbb{G}$ is $h \times (k - h - m_1)$. In doing so, remember all operations on the RHS of $\mathbb{H}_1$. This part of the RHS (h bits) will only be known after m_2 is determined. To find m_2 and generate $\mathbb{D}_3$, the sender keeps appending random dense rows to the whole matrix one-by-one and each time runs the Gaussian elimination on them to see if $\mathbb{H}_3$ has full rank (stops when $\mathbb{H}$ ceases to be of full rank). Again, all operations with the RHS of $\mathbb{D}_1$ are remembered. At the end of this process, $\mathbb{H}$ will be upper diagonal, ready for back-substitution. The role of $\mathbb{D}_3$ is to lower the absolute overhead from the LT code. Finally, m_1 and m_2 are encoded and substituted into the RHS of $\mathbb{D}_1$ and the whole system with $h + m_1 + m_2$ message bits on the RHS can be solved using back-substitution.

The complexity of this algorithm is dominated by the complexity of the Gaussian elimination on $\mathbb{H}_3$, $\mathcal{O}(m_2^2 k) \approx \mathcal{O}(g^2 k)$. The parameters of the robust soliton distribution should be chosen so that the overhead $g \lessapprox 10^4$ with high probability. By running computer simulations, the average overhead taken over different matrices $\mathbb{D}$, $E[k - m_1 - m_2 - h]$, was smaller than 2 bits.

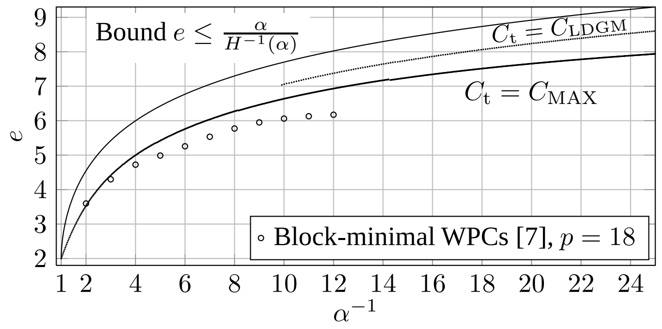


Fig. 4. Embedding efficiency of practical codes obtained from the wet ZZW construction with different C_{t}; $n = 10^6$ for $C_{\mathrm{t}} = C_{\mathrm{MAX}}$, $n = 4 \cdot 10^6$ for $C_{\mathrm{t}} = C_{\mathrm{LDGM}}$, and loss $\Delta_p = 20\lceil \lg p \rceil$ bits. Dry elements density $\rho = 0.1$.

5. PRACTICAL WPCS OBTAINED FROM WET ZZW

As mentioned in Section 3, all elements of the non-shared selection channel $\mathbf{v}$ can be made almost dry for large enough p. By this observation, the code C_{t} can be taken as Low Density Generator Matrix (LDGM) code [4]. Although these codes were designed only for dry elements, information about very few wet elements can be easily incorporated into the algorithm by correcting the optimal codeword to satisfy the requirements from wet elements. Since the codewords in the LDGM matrix are sparse, adding few codewords will not increase the number of changes and thus this operation can be used as a post-correction step to satisfy the wet elements. For this reason, the LDGM code $(n_{\mathrm{t}}, n_{\mathrm{t}}/2, 0.116 n_{\mathrm{t}})$, denoted C_{LDGM}, can be used for $n_{\mathrm{t}} \ge 10^4$ as a WPC C_{t}. For smaller n_{t}, the number of changes is larger.

Figure 4 shows the experimental results obtained for $n = n_{\mathrm{t}}p = 10^6$ and $n = 4 \cdot 10^6$ elements, $\rho = 0.1$, and $\Delta_p = 20\lceil \lg p \rceil$ for $p \ge 1$, along with the WPCs obtained from [6]. WPCs in Figure 4 were obtained from $C_{\mathrm{t}} = C_{\mathrm{MAX}}$ and $C_{\mathrm{t}} = C_{\mathrm{LDGM}}$ and can be realized in practice because the overhead of 20 bits (18 bits to communicate the parameters of W_l plus two bits overhead) can be realized by WPCs described in Section 4. For small payloads $\alpha < \frac{1}{4}$, these codes clearly outperform all known codes.

6. CONCLUSION

Wet paper codes are an essential tool for the steganographer as they allow the sender to place the embedding changes within the cover in an arbitrary manner that does not have to be shared with the recipient. Minimizing the number of embedding changes further decreases statistical detectability of embedding.

This paper describes a new approach to wet paper codes that achieve high embedding efficiency, outperforming current state of the art. The codes are built using an idea similar to the ZZW construction for matrix embedding. The embedding efficiency is estimated by combining analytic and experimental results under the assumption that the non-shared selection channel is i.i.d. Bernoulli. We pay close attention to implementation issues and minimizing the overhead as it may rather significantly influence the resulting embedding efficiency.

7. REFERENCES

[1] I. F. Blake and C. Studholme. Properties of random matrices and applications. Unpublished report available at `http://www.cs.toronto.edu/~cvs/coding`.

[2] T. Filler, A. D. Ker, and J. Fridrich. The Square Root Law of steganographic capacity for Markov covers. In *Proc. SPIE, EI*, volume 7254, pages 08 1–08 11, 2009.

[3] J. Fridrich. Asymptotic behavior of the ZZW embedding construction. *IEEE TIFS*, 4(1):151–153, Mar. 2009.

[4] J. Fridrich and T. Filler. Practical methods for minimizing embedding impact in steganography. In *Proc. SPIE, EI*, volume 6505, pages 02–03, 2007.

[5] J. Fridrich, M. Goljan, and D. Soukal. Efficient wet paper codes. In *7th Inform. Hiding Worksh. (IHW)*, volume 3727 of *LNCS*, pages 204–218. Springer-Verlag, 2005.

[6] J. Fridrich, M. Goljan, and D. Soukal. Wet paper codes with improved embedding efficiency. *IEEE TIFS*, 1(1):102–110, 2006.

[7] J. Fridrich, M. Goljan, D. Soukal, and P. Lisoněk. Writing on wet paper. In *IEEE TSP, Special Issue on Media Security*, volume 53, pages 3923–3935, Oct. 2005.

[8] F. Galand and C. Fontaine. How Reed-Solomon codes can improve steganographic schemes. *EURASIP Journal on Information Security*, 2009.

[9] J. Kodovský, J. Fridrich, and T. Pevný. Statistically undetectable JPEG steganography: Dead ends, challenges, and opportunities. In *Proc. 9th ACM Multim. & Sec. Workshop*, pages 3–14, 2007.

[10] D. Schönfeld and A. Winkler. Embedding with syndrome coding based on BCH codes. In *Proc. 8th ACM MmSec. Workshop*, pages 214–223, 2006.

[11] W. Zhang, X. Zhang, and S. Wang. Maximizing steganographic embedding efficiency by combining Hamming codes and wet paper codes. In *10th IHW*, volume 5284 of *LNCS, pg. 60–71*. Springer-Verlag, 2008.

[12] X. Zhang, W. Zhang, and S. Wang. Efficient double-layered steganographic embedding. *Electronics Letters*, 43:482–483, Apr. 2007.