



NRL/MR/5540--08-9118

On LSB Spatial Domain Steganography and Channel Capacity

IRA S. MOSKOWITZ

*Center for High Assurance Computer Systems
Information Technology Division*

PATRICIA A. LAFFERTY

FARID AHMED

*The Catholic University of America
Washington, DC*

March 21, 2008

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE (DD-MM-YYYY) 21-03-2008		2. REPORT TYPE NRL Memorandum Report		3. DATES COVERED (From - To) March 2007 – February 2008	
4. TITLE AND SUBTITLE On LSB Spatial Domain Steganography and Channel Capacity				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER 0602235N	
6. AUTHOR(S) Ira S. Moskowitz, Patricia Lafferty,* and Farid Ahmed*				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER 6326	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory, Code 5540 4555 Overlook Avenue, SW Washington, DC 20375-5320				8. PERFORMING ORGANIZATION REPORT NUMBER NRL/MR/5540--08-9118	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research One Liberty Center 875 North Randolph St., Suite 1425 Arlington, VA 22203-1995				10. SPONSOR / MONITOR'S ACRONYM(S)	
				11. SPONSOR / MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.					
13. SUPPLEMENTARY NOTES *Catholic University of America, Washington, DC					
14. ABSTRACT In this paper, we show that for a generic greyscale bitmap it is possible to successfully embed a steganographic payload that survives JPEG compression. We describe our channel model, explain the necessary information theory, and derive capacity estimations to back our hypotheses. We also contrast our approach with other information theoretic efforts in the field of steganography.					
15. SUBJECT TERMS Information hiding Information theory Channel capacity Steganography JPEG Stego Scrubber					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Ira S. Moskowitz
Unclassified	Unclassified	Unclassified	UL	24	19b. TELEPHONE NUMBER (include area code) (202) 404-7930

CONTENTS

1	Introduction	1
2	Scrubbing in the Large	2
3	Stego Scrubber.....	3
3.1	Aspects of scrubbing	4
4	Mutual Information	5
5	Least Significant Bit (LSB) Steganography for Bitmaps	7
5.1	$KM^+ := KM$ modified, $n = 2$	7
6	Good Scrubbing.....	17
7	Conclusions and Future Directions	18
8	Acknowledgements	20
	References.....	20

On LSB Spatial Domain Steganography and Channel Capacity

Ira S. Moskowitz,¹ Patricia A. Lafferty² and Farid Ahmed³

¹Center for High Assurance Computer Systems, Code 5540
Naval Research Laboratory
Washington, DC 20375
`ira.moskowitz@nrl.navy.mil`

&

^{2,3}Dept. of Electrical Engineering and Computer Science
The Catholic University of America
Washington, DC 20064
`2plafferty@smcm.edu, 3ahmed@cua.edu`

Abstract

In this paper, we show that for a generic greyscale bitmap it is possible to successfully embed a steganographic payload that survives JPEG compression. We describe our channel model, explain the necessary information theory, and derive capacity estimations to back our hypotheses. We also contrast our approach with other information theoretic efforts in the field of steganography.

1 Introduction

In this paper, we take an information theoretic approach to some of the issues in image steganography. In steganography we are concerned with sending a covert message so that the existence of said covert message is known only to the sending and receiving parties. If and when the existence of this covert message is no longer hidden, the steganography fails — at this point it is immaterial, with respect to steganography, if the meaning of the message is understood. Steganography is considered to be successful only if it cannot be detected and cannot be thwarted.

The major contribution of this paper, after background material is presented, is to gauge in an information theoretic sense how JPEG compression affects stego payload that may be hidden in the least significant bits of a bitmap image.

In general, the success of steganography depends on the algorithm used, the size and type of the stego media used as a *cover* (the carrier), and the size and

type of the stego payload (or message). In this paper we only concern ourselves with greyscale bitmap cover images. We do not attempt to gauge the effectiveness of the stego-ness of the stego algorithm with our information theoretic analysis; others have attempted this e.g., [3, 18, 17, 4]. Rather, we accept the algorithms under study in this paper as being successful¹ stego algorithms and use information theory to determine how much information may be hidden, even after defensive attacks are made to thwart the stego, in the stego payload. We use this to show the efficacy of our proposed *Stego Scrubber* [16].

In [16] we analyzed various stego algorithms. We showed that our proposed method of scrubbing would cause the stego algorithms in question to fail. By this we mean that the stego algorithm could not “read” the hidden information. We also noted that the simple act of JPEGing an image would often have the same effect. So, at first glance it seems that our ideas about scrubbing can be easily taken care of. However, we showed in [16] that after JPEGing an image, even though the stego algorithm may fail to detect or read the hidden payload, a good part of the stego payload may still remain in the cover image, and that stronger methods, such as our proposed *Stego Scrubber*, are called for. In this paper, we wish to quantify the amount of hidden information that may still be in an image after JPEG compression. We use Shannon’s [21] notion of *capacity* for this quantification metric.

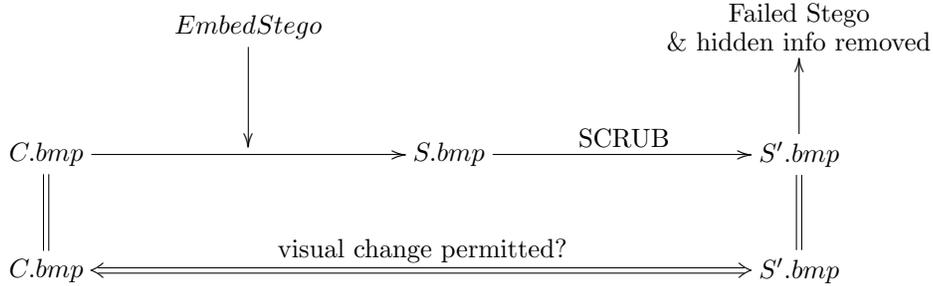
2 Scrubbing in the Large

Ideally, [16] the *Stego Scrubber* will take an image file in whatever format it is in, process it to remove the stego, and then put the image back into its original format with very little degradation to visually important and innocent image factors. Our prototype scrubbing [16, 13] involves a combination of both spatial domain and frequency domain techniques and is designed to work on various image cover formats. Again, though, **in this paper we perform our information theoretic analysis on greyscale bitmap covers**—unless noted otherwise (independent of format, we will always assume greyscale except for some of our concluding remarks).

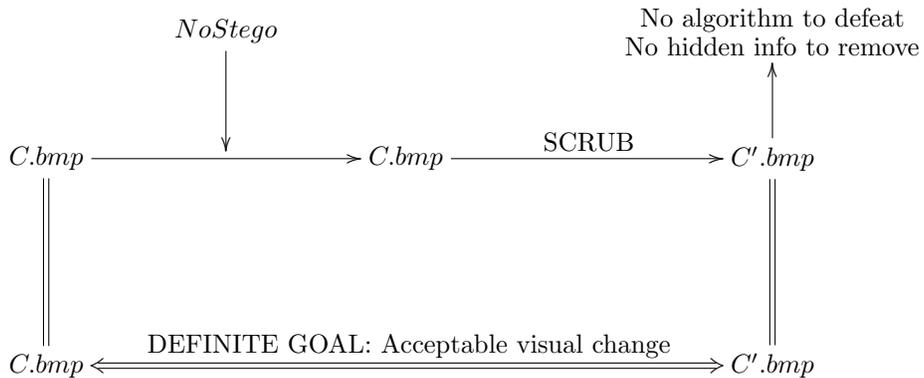
Consider the situation below. We assume that we have a suspect bitmap image $S.bmp$ that has a stego payload in it. We wish to modify the bitmap file $S.bmp$ so as to make sure that the stego algorithm is defeated. But just defeating this or that stego algorithm is not sufficient for us. We want to make sure that any hidden information is actually removed from $S'.bmp$. We cannot

¹Certainly, the stego model that we use later in the paper that writes over the 7th and 8th spatial bit planes is not a strong stego method since it is easily detectable. We use it as a worst case scenario. The goal of our paper is to see if LSB spatial domain stego can survive JPEG compression, not to see if it is detectable or not. Of course, a better LSB spatial domain stego method would not change the lower bit planes in a gross and ad hoc manner. That is a topic worthy of future information theoretic study.

assume that all images were sent by a “bad guy”.



So we do not want to affect the visual qualities of the image too much with our scrubbing, because we do not know if we are dealing with an innocuous image $C.bmp$ or a stego image $S.bmp$. We are taking the philosophy of assuming that it is a stego image for the sake of removing hidden information, but we also view it as an innocent image and do not wish to modify the visual aspects of it too much.



3 Stego Scrubber

In this section we will review the ideas of the Stego Scrubber as put forth in [16], and fully developed in [13]. Note that a similar type architecture has also recently been advocated, but not built, by [10].

Our thinking is that it is an impossible task to detect all steganography. Therefore, in a *controlled environment* we propose a method of scrubbing all image files leaving (and possibly entering) a controlled enclave. This will assure us that even if there was a stego payload in an image file, that we removed it. Of course, this scrubbing has to be performed in a judicious manner so that one does not destroy the benign image content. This is necessary because good guys as well as bad guys may be sending image files. Therefore, as discussed earlier, we need a method of stego scrubbing that does not moderate the image in a “visually” unacceptable manner.

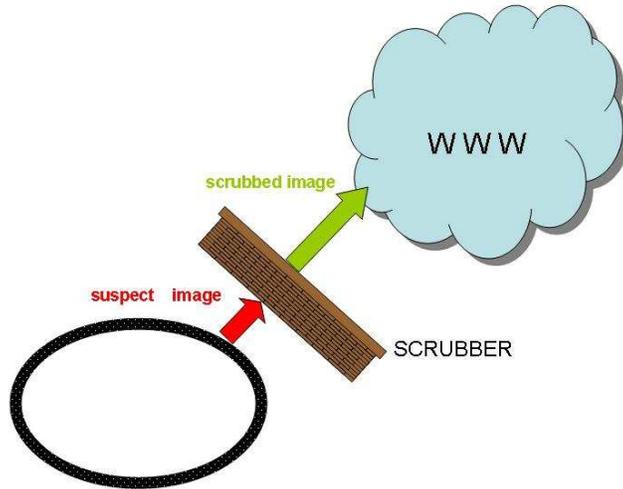


Figure 1: Proposed Use of Stego Scrubber for outgoing files

3.1 Aspects of scrubbing

In general, (stego) scrubbing involves two separate parts: a spatial domain part and a frequency domain part. These different aspects of scrubbing, along with their associated filtering techniques, are discussed in detail in [13]. In fact, Lafferty [13] recommends an averaging of different scrubbing techniques, done in parallel, and then averaged together. This has the meritorious affect of defeating the stego, while at the same time keeping image quality (e.g., PSNR) high. Keep in mind that JPEG compression can be considered a frequency domain type scrubbing operation.

Does JPEG compression alone suffice for removing all steganographic payload? The answer to this is certainly no as one can see by looking at the literature for the various type of JPEG survivable stego algorithms, e.g., [22, 23, 19, 20]. Of course, these are stego algorithms designed to deal with JPEG cover image, whereas, in this paper, we are interested in spatial domain based stego algorithms which use a bitmap cover image. However, in [16] is noted that even though the algorithm may fail to pick up the stego, that the parts of the stego payload may still be inside of an image after various forms of scrubbing. That is, the failure of a steganographic algorithm to reveal the hidden information should not be taken as proof that the image is now clean.

The survivability of LSB type spatial domain steganography against lossy compression has been pessimistically viewed in the field, e.g., [11, 24, 10]. Whitehead [24] stated “... found that some embedding techniques are susceptible to lossy image compression such as JPEG. This result is not surprising since the compression algorithms can be considered a form of channel distortion by removing elements of the image data that are not visually important.” Certainly if our bitmaps have large areas of constant luminance, then by adjusting the

LSBs in a 8×8 block manner one could encode information that would readily survive JPEG compression (in fact, our special “friendly” image at the end of this paper is such an image). Note that Currie and Irvine [8] have previously challenged the assumption that *LSB type stego methods for bitmaps can be defeated by JPEG compression*. Currie and Irvine [8] have noted that, for a bitmap, that LSBs are the worst places to insert steganographic payload. They developed two coding schemes to send stego payload in spite of JPEG compression. However, there are not enough details in [8] to determine which bit planes are used and what type of JPEG compression is in effect. It is our impression that higher order bit planes are in play. This is supported by the claim in [7] that 5 bit planes were used. We note that [7] is the basis for Currie and Irvine [8].

Of course, as noted above, stego methods that exist for JPEGs do not do their hiding in the spatial LSBs. These JPEG type stego algorithms are usually frequency based. To not confuse the reader we wish to emphasize:

...
That we want to see what happens to hidden payloads in the LSBs of bitmaps under JPEG compression, and in this light we will restrict our scrubbing study, for this paper, to bitmap images.

...
 To reiterate: This paper challenges the mindset that JPEG compressing an image is sufficient to destroy the steganography for spatial domain LSB type stego. We agree that JPEGing certainly may cause a stego algorithm to fail, but there may still be hidden payload in the modified image. This is why we advocate randomizing certain LSBs as a spatial scrub in addition to frequency based scrubbing methods. Information theory is used to show the potential for a hidden information to survive JPEG compression. Again, we restrict our in-depth analysis to bitmap images.

4 Mutual Information

We use Shannon’s discrete memoryless communication channel (DMC) as our model [21]. What makes things discrete is that the inputs and outputs are discrete. What makes things memoryless is that every DMC usage is independent and probabilistically identical to all other uses of the DMC. For simplicity, whenever we refer to a channel we mean a DMC. We let X denote the (discrete) input random variable to a channel and Y the (discrete) output random variable. We use the shorthand notation $p(x_i)$ to represent $P(X = i)$, $i = 1, \dots, N$ (similarly for $Y, j = 1, \dots, M$). Notice that the input distribution can be viewed [14] as a point p on the $N - 1$ simplex $\Delta^{N-1} \in \mathbb{R}^N$, where $(p(x_1), \dots, p(x_N)) = (p_1, \dots, p_N) = p \in \Delta^{N-1}$. For a generic random variable we will use R , $P(R = i) = p(r_i)$, $i = 1, \dots, \rho$. The conditional probability of Y conditioned on X is $p(y_j|x_i)$. In a channel $p(y_j|x_i)$ is a measurement of the noise in the channel. For example, a channel is defined to be noiseless if $M = N$ and $p(y_j|x_i) = \delta_{i,j}$, the Kroenecker delta.

The entropy of R is $H(R) = -\sum_i p(r_i) \log p(r_i)$.² $H(R)$ is maximized at $\log \rho$ when R is the (discrete) uniform variable — that is all the $p(r_i)$ are equal. The entropy $H(R)$ is minimized at zero if X has no probabilistic uncertainty³. We may also form the conditional entropy between X conditioned on Y as $H(X|Y) = -\sum_j p(y_j) \sum_i p(x_i|y_j) \log p(x_i|y_j)$ and similarly (the usually unequal) $H(Y|X) = -\sum_i p(x_i) \sum_j p(y_j|x_i) \log p(y_j|x_i)$. We define the mutual information between X and Y to be

$$I(X, Y) = H(X) - H(X|Y) \tag{1}$$

One can easily show that $I(X, Y) = I(Y, X)$, and the units of mutual information are bits per symbol (we do not consider output symbols that take different time values). Note, one can equivalently write the mutual information as:

$$I(X, Y) = \sum_i^N \sum_j^M \log \frac{p(x_i, y_j)}{p(x_i)p(y_j)} \tag{2}$$

Eq. 2 easily shows us that if X and Y are independent that $I = 0$.

Mutual information is a measure of how much information is actually passed over a channel. If a channel is *noiseless* one can show that $H(X|Y) = 0$, so that $I(X, Y) = H(X)$. So the amount of information that is passed in a noiseless channel is based on solely how X behaves. The mutual information ranges from the minimum value of zero to the maximum value of $\log N$. This is important to keep in mind.

For a general channel, the probabilistic behavior of X (not only how many non-probabilistically trivial range elements it has) still greatly affects the mutual information. In fact, the distribution of X is the only variable in the mutual information. This leads us to Shannon's famous [21] definition of capacity C .

$$C = \max_{p \in \Delta^{N-1}} I(X, Y) \tag{3}$$

The maximization is done over all possible distribution values for X .⁴ Since this is a compact set and $I(X, Y)$ is a continuous function, C is well defined. For a noiseless channel we have that $C = \log N$, which is not too surprising.

The definition of capacity is of no worth without a theorem, and Shannon supplied that in the form of his *noisy channel coding theorem* [21]. Shannon's theorem is extremely important: it states that capacity is an upper bound to essentially error-free communication, and if you attempt to transmit at a rate higher than C then you will (with non-trivial probability) have errors. The challenge, of course, is to build the proper codes so that you can transmit at

²All logarithms are base 2 so that we have bits. If we used base e , then our units would be in nats.

³We define $0 \log 0$ and $1 \log 1$ by their limiting values of zero.

⁴For example, if X takes on the values x_1 and x_2 and we set $p(x_1) = x$ and $p(x_2) = \bar{x}$, then we see that the straight line in the (x, \bar{x}) plane from $(0, 1)$ to $(1, 0)$ (the 1-simplex) represents the different possible distribution values for X . It is over this 1-simplex that we maximize the mutual information.

rates less than, but very close to C (we will not discuss this further, there is a huge body of work on error correcting codes and the like). Therefore, we take the capacity C as the measurement of how much information a channel may potentially pass. Keep in mind that the units of capacity are bits per symbol, or equivalently expressed as bits per channel use. In our study the symbols or channel uses turn out to be pixels.

5 Least Significant Bit (LSB) Steganography for Bitmaps

LSB (image⁵) steganography can easily hide large amounts of information without being noticed by the human visual system (HVS) (even though it may be easily detectable by other means). The ability to fool the HVS is the baseline “idiot test” for all steganographic algorithms. LSB steganography, in the small, may be able to avoid detection at the cost though of a small stego payload. Specifically, by *LSB steganography* we mean manipulating the least significant (luminance) bit or the n least significant (luminance) bits (we use $n = 2$ in the body of the paper) of a (greyscale) image. As noted, to avoid confusion our images, unless noted otherwise, are always uncompressed spatial images, so we just use the term bitmap, independent if our experiments are performed on .bmp or on (lossless) .tiff files, etc.

In our opinion, LSB steganography first came into wide view in the seminal paper by Kurak and McHugh [12]. The Kurak McHugh stego algorithm (KM for short), simply substitutes the n least significant bits of image C (C for cover) with the n most significant bits (MSBs) of image E (E for embedded) to form the stego image S . As noted above, we are dealing with 8 bit greyscale images viewed as bitmaps. C, E , and S must have the same dimensions to make the KM algorithm work. Therefore, for a pixel in S , we see that the first $8 - n$ bits are pixel-wise the same as those of C , whereas pixel-wise the last n bits agree with those of E . If n is large, then the HVS may not be fooled. However, if n is small, then unless C is a pathological image [16], one cannot see that the image has been tampered with. In fact, $n < 3$ usually suffices to fool the HVS.

Without loss of generality we will not concern ourselves with the horizontal M and vertical N dimensions of the image. However, we do follow the assumption that $M \times N$ is large. Therefore, we will assume that $M, N \geq 256$. We will return to this later.

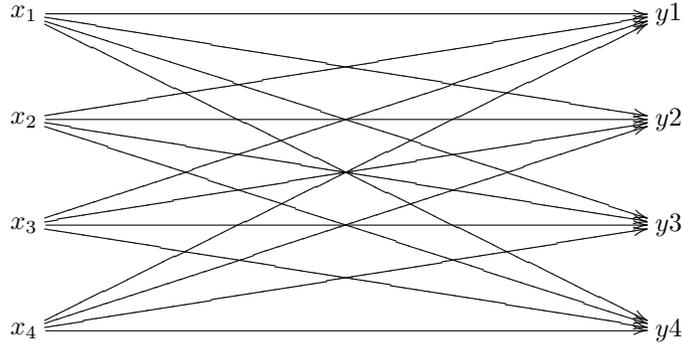
5.1 $KM^+ := KM$ modified, $n = 2$

For KM stego, the 2 LSBs of C are substituted with the 2 MSBs, pixel by pixel, of E . Now we slightly modify the KM algorithm so that the 2 bits that we hide per pixel are not necessarily the 2 MSBs of another image, they could just be

⁵LSBs have been manipulated for other cover types than images for the purposes of steganography—that is beyond the scope of this paper.

information bits that we want to transfer in a covert manner. We denote this method of steganography as KM^+ . How much information does KM^+ pass in a perfect world? Since we have 2 bits per pixel to play with we see that the answer is 2 bits per pixel, or $2MN$ bits per image. This was easy and obvious, but what if there was noise? In other words even though one may attempt to hide 2 specific bits per pixel, one would only have a probabilistic guarantee as to what those 2 bits would be.

The 6 highest bits of a pixel remain unchanged, so we are concerned with putting 2 bits in, and seeing what we get out. More specifically, we have four input symbols and four output symbols, and we wish to determine the behavior between the two sets. We are making a large, but realistic, within modeling bounds, assumption that each pixel represents a use of the channel, and each pixel transmission behaves independently, but probabilistically identically to the other pixels.



Channel transition diagram

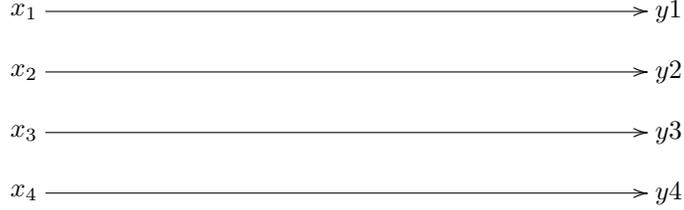
The 4 input symbols x_1, x_2, x_3, x_4 are the range of the input random variable X . They correspond to the values of the 2 LSBs 0,1,2,3. Writing them bitwise (bit₇,bit₈) we have:

$x_1 = (0,0) = 0$, $x_2 = (0,1) = 1$, $x_3 = (1,0) = 2$, $x_4 = (1,1) = 3$. The 4 output symbols y_1, y_2, y_3, y_4 are the range of the output random variable Y (taking the same possible values 0,1,2,3 as X). The 16 arrows in the channel transition diagram represent the conditional probabilities $p(y_j|x_i)$, which is the probability of receiving y_j given that x_i was the input. We form the channel transition matrix

$$T = \begin{pmatrix} p(y_1|x_1) & p(y_2|x_1) & p(y_3|x_1) & p(y_4|x_1) \\ p(y_1|x_2) & p(y_2|x_2) & p(y_3|x_2) & p(y_4|x_2) \\ p(y_1|x_3) & p(y_2|x_3) & p(y_3|x_3) & p(y_4|x_3) \\ p(y_1|x_4) & p(y_2|x_4) & p(y_3|x_4) & p(y_4|x_4) \end{pmatrix}$$

Again, our assumption is that the input corresponds to the initial 2 LSBs of a pixel for image C, and the output corresponds to the 2 LSBs of the same

pixel for S. If we take the raw KM⁺ algorithm without any further processing, then there is no noise and the channel transition diagram simplifies to



Noiseless Channel transition diagram

and the channel transition matrix is the identity matrix

$$T_{noiseless} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Now let us apply some of the information theory from above. If the channel is noiseless, that is the bitmap has the KM⁺ embedding done to it and no compression is applied so that the 2 LSBs are not further modified (no noise), the channel matrix is given as $T_{noiseless}$ and $H(Y|X) = -\sum_i p(x_i) \sum_j p(\delta_{ij}) \log p(\delta_{ij}) = 0$. Therefore, $I = H(Y)$. Since $p(y_j) = \sum_i p(y_j|x_i)p(x_i) = \sum_i p(\delta_{ij})p(x_i) = p(x_j)$, we see that $H(Y) = H(X)$. Since X is a random variable with N non-trivial values, we see that $H(X)$ is maximized for the discrete uniform distribution on N values and the maximum value is $\log N$. Hence $C = \log N$, since $N = 4$, we have, as before, that $C = 2$.

Y is independent of X iff $p(y_j|x_i) = p(y_j) \forall \{i, j\}$. Therefore, we see that independence is equivalent to the rows of T being identical. That is

$$T_{inden} = \begin{pmatrix} p(y_1) & p(y_2) & p(y_3) & p(y_4) \\ p(y_1) & p(y_2) & p(y_3) & p(y_4) \\ p(y_1) & p(y_2) & p(y_3) & p(y_4) \\ p(y_1) & p(y_2) & p(y_3) & p(y_4) \end{pmatrix}$$

Of course, for identical rows we can see by direct calculations, or the remarks after Eq. 2 that the mutual information is identically equal to zero, and of course then the capacity is zero. In general, we need the rows to be different to achieve non-zero capacity. One should keep in mind the capacity of a channel whose 4×4 channel matrix rows are all $(.7, .1, .1, .1)$ has a capacity that is “just as zero” as one whose rows are all $(.25, .25, .25, .25)$ (see [15] for more on this topic).

Thus, two is the largest capacity that can ever be achieved, and zero is the least. If there is noise the capacity will be less, and not as easy to calculate.

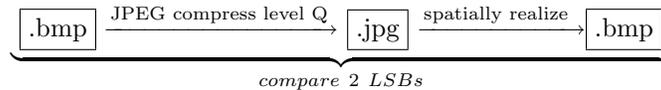
We wish to see what happens if we perform JPEG compression on the bitmap after the 2 LSB substitution. This is our steganography. We model this (see

Scenario B) by randomizing the 2 LSBs, *we stress that this randomization is for modeling the stego, it has nothing to do with scrubbing!* We model the stego by randomization to show that we are free to write over the 2 LSBs as we choose. Of course, if we scrub by also randomizing the 2 LSBs we are sure to remove all of the stego.⁶ Aside from the fact that JPEG is the standard way of dealing with image files on the web, it is also an important consideration in the course of designing and analyzing the Stego Scrubber. We consider the effects of JPEG compression on the image with and without the KM⁺ stego algorithm.

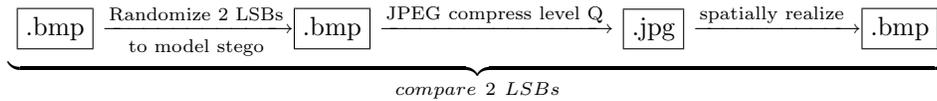
5.1.1 Our Experiments

We use a database of 20 greyscale TIFF images of various sizes downloaded off the Internet (as uncompressed TIFF files, though we do not know their history). We re-size them using the shareware xv [2] to size 256x256 and save them as .bmps. Our concern is what JPEG does to the 2 LSBs. One must keep in mind that JPEG does not act upon the 2 LSBs independently from the higher bits. JPEG quantizes discrete cosine transform (DCT) coefficients, these frequencies are functions of all 8 bit planes for each 8 × 8 pixel block. Therefore, we must analyze the effects of JPEG compression on images that have not had the 2 LSBs modified, and images that have.

(Non-stego) Scenario A: Modeling effects of JPEG compression



(Stego) Scenario B: Modeling effects of KM⁺ Stego followed by JPEG compression



Scenario A: This is our best case scenario. We take each original (cover) bitmap and save it as a JPEG at the compression levels 0 to 100, in steps of 5 using Matlab. We view the before and after of the 2 LSBs of each pixel as a communication channel. As discussed, we want to know, in an information theoretic sense, how much information is passed by this channel. To do that we estimate the capacity under the various JPEG quality levels. Keep in mind that nothing has been done to the LSBs prior to compression. None of the spatial bit values are changed. This would be realistic if steganography were magic, *which*

⁶This is obvious and the mathematics backs it. Randomizing the 2 LSBs of each pixel gives us a channel matrix with equal rows (.25, .25, .25, .25). We return to this later in the paper.

it is not! This unrealistic scenario gives us an information theoretic baseline if the steganographic algorithm was a variant of KM⁺ that was extremely adaptive to the cover image. We want to see what happens to the 2 LSBs via the process of JPEG compression. It is important that we are not only adjusting the 2 LSBs because, as discussed, the higher order bit planes influence what JPEG compression does to the 2 LSBs and we assume that the 2 LSBs are in some way correlated to the higher order bit-planes in a natural image—this is why this best case scenario leaves the 2 LSBs untouched.

That is no stego is occurring in Scenario A. Since stego will change the LSBs, and remove the natural patterns that exist between the LSBs and the higher order bits we view Scenario A (as discussed) as the upper bound for how much information may still be passed even though an image has been compressed with the caveat that steganography has not been performed (again Scenario A may serve as a capacity upper bound for very adaptive steganography).

Scenario B: In this scenario we take the most pessimistic viewpoint in terms of information theory. The steganography does what it likes to the 2 LSBs. We model this, and it is debatable, as randomizing the 2 LSBs. Randomization destroys all correlation between the lower 2 bits and the upper 6 bits, so JPEG compression stands a very good chance of overwriting the LSBs even more than in Scenario A. After the 2 LSBs are randomized the modified bitmap is saved as a JPEG (as in Scenario A).

Channel Matrix Production: Scenarios A & B: For our analysis we only consider the 2 LSBs and ignore the 6 higher order bit planes.⁷ For each of the two scenarios we track how the values of 0,1,2,3 (the least 2 bits) change due to JPEG compression for each of the twenty images. We then form the channel matrix where row i is the normalized (by the total number of i values in each image) number of i values that become (or stay) 0,1,2,3 after the JPEG compression (thus each row entry is between 0 and 1, and they sum to 1). The normalization gives us probabilities. This 4×4 matrix is the probability transition matrix such that the $(i, j)^{th}$ matrix entry is the conditional probability $P(Y = j | X = i)$ of value i being value j after the JPEG compression. Recall that X represents the input, and Y the output.

From the twenty images, and for each scenario, we calculate the average channel matrix for each Q level. These matrices follow below. Note that the scenario A matrices have smaller variances (variance data not shown) than did scenario B.

Scenario A Results: We include the mean channel matrices for Q_A@60% to Q_A@100% in steps of 5.

⁷For example, when we say that the value is 3 we mean that the 7th and 8th bits are one, we do not care what the other bits are.

$$\begin{array}{cc}
Q_A@100\% & Q_A@95\% \\
\left(\begin{array}{cccc} 0.9099 & 0.0445 & 0.0000 & 0.0456 \\ 0.0454 & 0.9106 & 0.0440 & 0 \\ 0.0000 & 0.0434 & 0.9144 & 0.0422 \\ 0.0444 & 0 & 0.0452 & 0.9104 \end{array} \right) & \left(\begin{array}{cccc} 0.3205 & 0.2413 & 0.1981 & 0.2400 \\ 0.2425 & 0.3174 & 0.2400 & 0.2002 \\ 0.1878 & 0.2262 & 0.3570 & 0.2290 \\ 0.2406 & 0.2005 & 0.2410 & 0.3179 \end{array} \right)
\end{array}$$

$$\begin{array}{cc}
Q_A@90\% & Q_A@85\% \\
\left(\begin{array}{cccc} 0.2911 & 0.2425 & 0.2222 & 0.2443 \\ 0.2469 & 0.2850 & 0.2446 & 0.2235 \\ 0.2099 & 0.2301 & 0.3267 & 0.2333 \\ 0.2458 & 0.2219 & 0.2442 & 0.2881 \end{array} \right) & \left(\begin{array}{cccc} 0.2784 & 0.2498 & 0.2265 & 0.2453 \\ 0.2466 & 0.2777 & 0.2461 & 0.2296 \\ 0.2159 & 0.2304 & 0.3199 & 0.2339 \\ 0.2449 & 0.2278 & 0.2467 & 0.2806 \end{array} \right)
\end{array}$$

$$\begin{array}{ccc}
Q_A@80\% & Q_A@75\% & \text{Matlab default} \\
\left(\begin{array}{cccc} 0.2718 & 0.2469 & 0.2335 & 0.2478 \\ 0.2503 & 0.2698 & 0.2469 & 0.2330 \\ 0.2190 & 0.2317 & 0.3124 & 0.2369 \\ 0.2468 & 0.2313 & 0.2483 & 0.2736 \end{array} \right) & \left(\begin{array}{cccc} 0.2712 & 0.2464 & 0.2346 & 0.2479 \\ 0.2491 & 0.2695 & 0.2453 & 0.2360 \\ 0.2217 & 0.2317 & 0.3105 & 0.2362 \\ 0.2480 & 0.2359 & 0.2444 & 0.2717 \end{array} \right) &
\end{array}$$

$$\begin{array}{cc}
Q_A@70\% & Q_A@65\% \\
\left(\begin{array}{cccc} 0.2685 & 0.2466 & 0.2381 & 0.2467 \\ 0.2507 & 0.2604 & 0.2522 & 0.2368 \\ 0.2245 & 0.2298 & 0.3068 & 0.2389 \\ 0.2495 & 0.2366 & 0.2432 & 0.2707 \end{array} \right) & \left(\begin{array}{cccc} 0.2690 & 0.2458 & 0.2388 & 0.2464 \\ 0.2520 & 0.2643 & 0.2443 & 0.2393 \\ 0.2239 & 0.2346 & 0.2754 & 0.2661 \\ 0.2488 & 0.2388 & 0.2558 & 0.2565 \end{array} \right)
\end{array}$$

$$\begin{array}{c}
Q_A@60\% \\
\left(\begin{array}{cccc} 0.2636 & 0.2476 & 0.2390 & 0.2499 \\ 0.2488 & 0.2591 & 0.2518 & 0.2403 \\ 0.2248 & 0.2350 & 0.2737 & 0.2665 \\ 0.2581 & 0.2391 & 0.2462 & 0.2566 \end{array} \right)
\end{array}$$

If one were to use JPEG compression as a means of removing stego, one would have to consider the image degradation to the cover image caused by the JPEG compression. The PSNR is a good measurement of the degradation and pretty much independent of the 2 LSBs. The PSNR for Scenario A for Matlab quality 75 is 34 dB, for quality 60 it is 32 dB. This is the most likely the lowest cover degradation that would be tolerated by users of a Stego Scrubber. When looking at the PSNR values for Scenario B there is not much difference from Scenario A. This is because the 2 LSBs have very little effect upon the PSNR.

Consider the first row of each matrix $Q_A@x\%$. There is the maximum value for the first entry. This is not surprising, because this is the probability that a 0 is received conditioned on a 0 being transmitted. The third entry

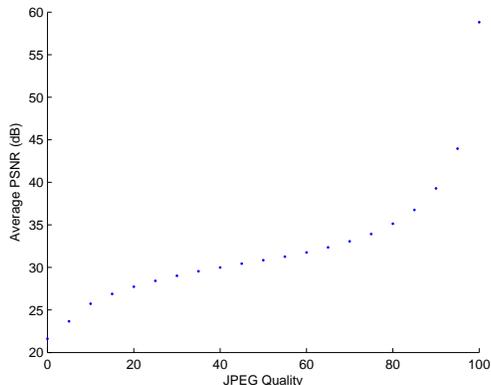


Figure 2: Plot of Average PSNR (Scenario A) vs. Matlab quality for 20 unmodified greyscale 256x256 images.

is the least which is the probability $P(Y = 2|X = 0)$, the other two probabilities are approximately equal. This pattern is repeated on every row, with the fixed point $P(Y = i|X = i)$ being the largest, and the least value being $P(Y = (\alpha, \beta) \oplus (1, 0)|X = (\alpha, \beta))$, that is only the 7th bit changes. The remaining two conditional probabilities are approximately equal.

This allows us to approximate and model the mean channel matrices in the form (for visual clarity we notationally suppress the dependence on Q .)

$$T_{\text{mean}} = \begin{pmatrix} L & (1-L-m)/2 & m & (1-L-m)/2 \\ (1-L-m)/2 & L & (1-L-m)/2 & m \\ m & (1-L-m)/2 & L & (1-L-m)/2 \\ (1-L-m)/2 & m & (1-L-m)/2 & L \end{pmatrix}$$

We model the channel matrices so that we can easily get a capacity bound. L is the average maximum value for each row, and m is the average minimum value for each row. We note that these mean matrices T_{mean} are only approximations, most, not all, but many images behave in a manner close to the mean.

Note that T_{mean} has the property that each row is a permutation of every other row, and each column is a permutation of every other column.

Interesting Observation: This symmetry behavior in the channel matrix, which is not as strong when stego is in the 2 LSBs, may in fact be a *novel method of stego detection!* This is a topic for future study.

This tells us that the channel described by T_{mean} is a symmetric channel and we may apply the well known [1, 3.3.1] result about symmetric channels. Applying this result we have that the capacity, for a channel given by T_{mean} , denoted as C_{mean} is

$$C_{\text{mean}} = 2 + L \log L + m \log m + (1-L-m) \log(1-L-m)/2 \quad (4)$$

For each matrix $Q@x\%$ we took the average of the four maximum values to get L and the average of the four minimum values to get m . In general, the problem of obtaining capacity exactly is non-trivial. This is because closed form solutions only exist for binary input binary output discrete memoryless channels and for certain type of channels with symmetries as shown in Eq. 4. However, since we are only dealing with four inputs and four outputs the numerical optimization of the mutual information can be easily performed. Table 1 shows the Q levels vs. the capacity obtained by the symmetry assumption and by numerical optimization of the mutual information. Of course, since we numerically found the capacity we could have done without the discussion of the symmetry assumption, but we felt that it was a worthwhile discussion for the reader and may be used when numerical techniques become too time consuming and for generalizations and extensions of the theory. Also, as noted above, this symmetry may be interpreted as telling us that we are dealing with a “stego-free” image which is of extreme importance to the detection community. Also, the reader should keep in mind that the capacity for individual images varies from the average behavior we have been concentrating on—in short, we are looking for back of the envelope type estimates to gauge the potential stego threat in general. Of course, as we do later, one should also study the images that are best suited to defeating stego compression with respect to the stego payload. *The capacity results for Scenario A may be interpreted as an upper bound on stego capacity*, which would be the case if the stego algorithm was very adaptive to the higher order bit planes of the image.

Table 1: Capacity for Scenario A

Q	Symmetry approx. bits/pixel	Numerical calculation bits/pixel
60	.001	.002
65	.001	.003
70	.003	.005
75 (default)	.004	.006 ⁻
80	.004	.006 ⁺
85	.006	.01 ⁻
90	.01	.01 ⁺
95	.03	.03
100	1.49	1.48

Even though capacity values are low on a bits per pixel basis, at the default Matlab Q level of 75 we do have an information theoretic payload around 50 bytes per 256×256 image. Not a lot, but not zero either! Smart coding can help us exploit this. Keep in mind that this is Scenario A where no stego is present—we have used the natural bit planes of a bitmap image. In Scenario B we change that, though. We use Fig. 3 to show a best possible case scenario. Now we will analyze Scenario B as described above.

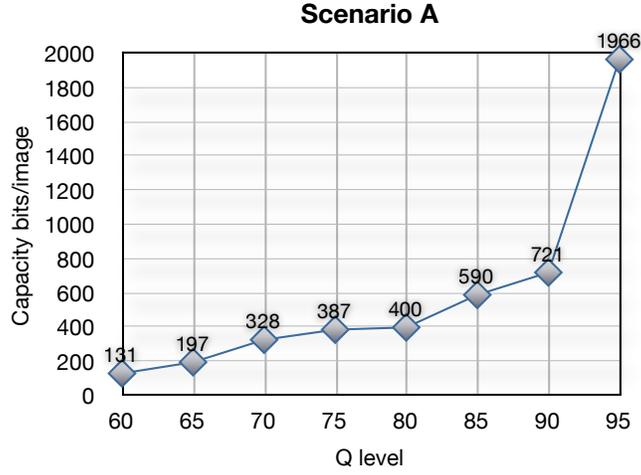


Figure 3: Plot of Capacity (bits per pixel) vs. Q , $Q \leq 95$

Scenario B Results: Let us show the mean channel matrices for Scenario B.

$$\begin{array}{l}
 Q_B@100\% \\
 \begin{pmatrix} 0.9084 & 0.0386 & 0 & 0.0520 \\ 0.0480 & 0.9085 & 0.0436 & 0 \\ 0 & 0.0446 & 0.9076 & 0.0477 \\ 0.0509 & 0 & 0.0410 & 0.9080 \end{pmatrix}
 \end{array}
 \begin{array}{l}
 Q_B@95\% \\
 \begin{pmatrix} 0.2700 & 0.2769 & 0.2268 & 0.2263 \\ 0.2396 & 0.2875 & 0.2592 & 0.2137 \\ 0.2143 & 0.2600 & 0.2867 & 0.2390 \\ 0.2248 & 0.2265 & 0.2773 & 0.2714 \end{pmatrix}
 \end{array}$$

$$\begin{array}{l}
 Q_B@90\% \\
 \begin{pmatrix} 0.2460 & 0.2765 & 0.2493 & 0.2282 \\ 0.2375 & 0.2707 & 0.2618 & 0.2301 \\ 0.2298 & 0.2607 & 0.2717 & 0.2378 \\ 0.2287 & 0.2496 & 0.2750 & 0.2467 \end{pmatrix}
 \end{array}
 \begin{array}{l}
 Q_B@85\% \\
 \begin{pmatrix} 0.2390 & 0.2674 & 0.2613 & 0.2322 \\ 0.2360 & 0.2619 & 0.2693 & 0.2328 \\ 0.2317 & 0.2586 & 0.2727 & 0.2370 \\ 0.2296 & 0.2519 & 0.2778 & 0.2406 \end{pmatrix}
 \end{array}$$

$$\begin{array}{l}
 Q_B@80\% \\
 \begin{pmatrix} 0.2408 & 0.2742 & 0.2520 & 0.2330 \\ 0.2364 & 0.2732 & 0.2559 & 0.2344 \\ 0.2343 & 0.2690 & 0.2591 & 0.2376 \\ 0.2329 & 0.2642 & 0.2621 & 0.2409 \end{pmatrix}
 \end{array}
 \begin{array}{l}
 Q_B@75\% \\
 \begin{pmatrix} 0.2376 & 0.2671 & 0.2620 & 0.2333 \\ 0.2353 & 0.2648 & 0.2654 & 0.2346 \\ 0.2336 & 0.2620 & 0.2683 & 0.2361 \\ 0.2338 & 0.2574 & 0.2712 & 0.2376 \end{pmatrix}
 \end{array}
 \begin{array}{l}
 \text{Matlab} \\
 \text{default}
 \end{array}$$

$$\begin{array}{l}
 Q_B@70\% \\
 \begin{pmatrix} 0.2369 & 0.2607 & 0.2679 & 0.2345 \\ 0.2357 & 0.2587 & 0.2700 & 0.2357 \\ 0.2343 & 0.2576 & 0.2722 & 0.2358 \\ 0.2349 & 0.2555 & 0.2732 & 0.2364 \end{pmatrix}
 \end{array}
 \begin{array}{l}
 Q_B@65\% \\
 \begin{pmatrix} 0.2354 & 0.2723 & 0.2563 & 0.2360 \\ 0.2362 & 0.2697 & 0.2590 & 0.2351 \\ 0.2338 & 0.2679 & 0.2608 & 0.2376 \\ 0.2339 & 0.2669 & 0.2608 & 0.2384 \end{pmatrix}
 \end{array}$$

$$Q_B@60\% \begin{pmatrix} 0.2379 & 0.2594 & 0.2676 & 0.2352 \\ 0.2366 & 0.2581 & 0.2689 & 0.2363 \\ 0.2357 & 0.2586 & 0.2695 & 0.2362 \\ 0.2350 & 0.2579 & 0.2705 & 0.2367 \end{pmatrix}$$

Table 2 is a table of the capacity, to two significant (except for $Q = 100$) digits, for Scenario B.

Table 2: Capacity for Scenario B

Q	Numerical calculation bits/pixel
60	.000015
65	.000040
70	.000043
75 (default)	.00014
80	.00023
85	.00047
90	.0014
95	.0067
100	1.47

Not surprisingly, $Q_B@100\%$ is similar in form to $Q_A@100\%$, we have an average value of $L = .9081$, and an average value of $m = 0$. Therefore, we approximate the capacity of $Q_B@100\%$ by using, as before, Eq. 4, and we have that $C_{Q_B@100\%} \approx 1.47$. However (we ignore $Q=95\%$ for now) other capacities are much lower and we cannot make the symmetry assumptions that we did for Scenario A (again this gives credence to the idea of using the symmetry as a stego detector).

What is interesting that this modeling of 2 bit LSB steganography shows that theoretically there is non-zero stego payload possible even though the image has been JPEGed.

We wish to emphasize an important point—the data we were using was over an average of twenty images. What is to stop us from using an image that gives us even more room for steganography? Not surprisingly, images with large areas of constant luminance do that for us, since JPEG compression respects these areas.

The channel matrix for Figure 5, our stego friendly image, at JPEG quality 75% is

$$T_{Q@75\% \text{ friendly}} \begin{pmatrix} 0.1400 & 0.3706 & 0.3534 & 0.1360 \\ 0.1347 & 0.3515 & 0.3798 & 0.1340 \\ 0.1325 & 0.3235 & 0.4048 & 0.1392 \\ 0.1249 & 0.3063 & 0.4297 & 0.1391 \end{pmatrix}$$

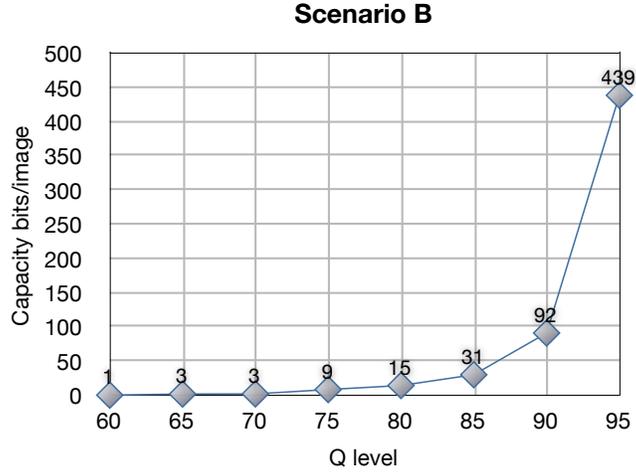


Figure 4: Plot of Capacity (bits per pixel) vs. Q , $Q \leq 95$

One can see that the rows of this matrix $T_{Q@75}$ are very different. Numerical calculations show that the capacity is .0052 bits per pixel. This is 37 times the average capacity for 75% quality from our database of twenty images. For a 256×256 image this would result in a payload of 341 bits. This means that if a transmitter picks their cover image properly, they stand a good chance of getting a moderate sized stego payload through even though the image has been JPEG compressed. We do remind the reader that the physical reason the above image is so stego friendly is that it has large areas of constant luminance, which can be manipulated in a block by block manner that JPEG compression respects. Please note that a more generic image such as Fig. 6 has a capacity at Q level 75, about twice that of the mean Q level 75 capacity.

6 Good Scrubbing

Part of the Stego Scrubber as put forth in [16], involves randomizing the LSBs of an image in its bitmap form. Let us see, from an information theoretic viewpoint, what that does to the stego payload. Again, we are only concerning ourselves with the 2 LSBs. Consider that the input 2 bits. The probability of an input value going to 0, 1, 2, or 3 are all equal and equal to .25 (assuming a proper randomizer). Therefore, we see that in this case, under our construction of the transition probabilities ($P(Y = j|X = i)$) we would have a 4×4 channel matrix with each value being .25 (the larger the image the closer we come to this idealized situation).⁸ Since a channel matrix with identical rows has mutual

⁸We numerically simulated this for 256×256 images and got matrices that were very close to each entry being .25. Our numerical capacity calculations in these situation resulted in

information, and thus capacity zero, we see that stego hidden in the 2 LSBs of a bitmap, even though it may survive JPEG compression, will not survive our Stego Scrubber.

In terms of information theory, we see that the Q_B matrices give us a model for the effects of JPEG compression on a generic 2 LSB stego payload. Scrubbing the 2 LSBs at this point tells us that we are forming a cascade of channels; the first channel being given by Q_B and the second channel being given by the matrix Ω , where Ω is a 4×4 matrix with each entry being $1/4$. From Feinstein's data theorem result [9, Sec 3.3], we know that the capacity of a cascade is less than or equal to the capacity of any channel making up the cascade. Since the capacity of the channel represented by Ω is zero, we see that, from an information theoretic viewpoint, no information can be passed if we scrub the 2 LSBs of a bitmap image with stego in the 2 LSBs (this of course is also obvious from common sense). We can also obtain this by the fact that matrix multiplication gives us $Q_B \cdot \Omega = \Omega$, since each row in Q_B sums to one.

Keep in mind that the Stego Scrubber [16, 13] is designed to run on JPEG images also!⁹ The Stego Scrubber will scrub JPEG images that potentially have stego payload hidden in the 2 LSBs of their spatial realizations. Therefore, one of the actions that the Stego Scrubber will perform is to spatially realize a JPEG image and then randomize the 2 LSBs, then the Stego Scrubber will convert the bitmap back into a JPEG image at the same, or different compression level. Let us consider this from an information theoretic viewpoint. The channel matrix describing the channel from the stego laden spatial realization of the JPEG to the bitmap with the 2 LSBs randomized is simply Ω (this follows from the argument in the preceding paragraph). If we convert this scrubbed bitmap to a JPEG and then back into a bitmap, the channel matrix describing that transition will be denoted as M , and is actually one of the Q_B matrices (since the actions are the same). No matter what M actually is as long as it is a channel matrix for a discrete memoryless channel we have the algebraic fact that $\Omega \cdot M$ is a matrix with identical rows. Note that $\Omega \cdot M$ is not necessarily a matrix with every entry $1/4$, but it is a matrix such that every entry in the j^{th} column is the average of the four column entries of the j^{th} column of M . Of course, a matrix with identical rows has zero mutual information and zero capacity. What is interesting is that both Ω and $\Omega \cdot M$ describe channels with zero capacity, but Ω is unique in that it is a doubly stochastic matrix. Extended work in this area is being done by Martin [15].

7 Conclusions and Future Directions

We have shown that it is possible for a stego payload in the 2 LSBs of a bitmap image to survive JPEG compression. We have only analyzed in detail greyscale bitmaps in this paper, but our results extrapolate to color bitmaps also.

capacities on the order of 10^{-5} or less (some where 10^{-4}). These ϵ -capacity results are due to the vagaries of numerical simulation and random number generation.

⁹Until this paragraph, we have only dealt with bitmaps in this paper.

We have learned that the stego payload size is small, and certain images exist as covers that are better than others, and proper error corrected codes need to be used to send and retrieve the hidden information. Also, many of our results were for average behavior; however for the very important example given by Fig. 5 we did not use average behavior.

We have discussed how the Scenario A capacity values may be used as an upper bound, and the Scenario B capacity values as a lower bound for potential stego payload. We have discussed how the symmetric matrices demonstrated for Scenario A may be useful as a stego detection tool. Our results further justify the use of LSB randomization in our proposed Stego Scrubber.



Figure 5: Stego friendly



Figure 6: Stego more normal

We ask the reader to keep in mind that we used the simplifying assumptions that each pixel, viewed as a communication channel, is independent and identical to all other pixels in a communication theory view. This, of course, is not true for JPEG compression. In fact, the pixels in 8×8 blocks are highly dependent upon each other via the discrete cosine transformation. We used our simplifying assumptions to show that JPEG stego survivability was possible in the 2 LSBs, not as the be all and end all of the discussion. Certainly using blocks, or other channel models, can results in other capacity estimations. In fact, we see that our stego friendly image had large areas where 8×8 blocks were constant. Some preliminary experiments have shown us that we can change those block values to easily obtain a robust stego encoding. In fact, we could have just demonstrated that fact without the information theory, but we chose not to in order to study, at the basic level the information theoretic basics for surviving JPEG compression.

Certainly further theoretic work is called for. In future work, we would be most interested to compare our capacity estimates with the existing purely theoretical Gaussian assumption-type results, e.g., see [18, 17]. Historically one should also look at Costa's dirty paper [5] article, which is an extension of Shannon's Gaussian channel capacity results [21] and [6, Eq. 10.20]. More detailed of some of our analysis will also be in [13].

8 Acknowledgements

We are grateful for helpful conversations with Keye Martin and Gerard Allwein.

References

- [1] Robert B. Ash. *Information Theory*. Dover, 1990.
- [2] John Bradley. xv. <http://www.trilon.com/xv/>.
- [3] Christian Cachin. An information-theoretic model for steganography. In D. Aucsmith, editor, *Information Hiding, Second International Workshop*, pages 306–318. Springer-Verlag, LNCS 1525, April 1998.
- [4] R. Chandramouli and Nasir Memon. Steganography capacity: A steganalysis perspective. In *Proc. SPIE Security and Watermarking of Multimedia Contents*, pages 173–177. SPIE, January 2003.
- [5] Max H.M. Costa. Writing on dirty paper. *IEEE Transactions on Information Theory*, 29:439–441, May 1983.
- [6] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley Interscience, 1991.
- [7] Daniel L. Currie III and Hannelore Campbell. Implementation and efficiency of steganographic techniques in bitmapped images and embedded data survivability against lossy compression schemes. Master’s thesis, Naval Postgraduate School, 1996.
- [8] Daniel L. Currie III and Cynthia E. Irvine. Surmounting the effects of lossy compression on steganography. In *Proc. 19th National Information Systems Security Conference*, 1996.
- [9] Amiel Feinstein. *Foundations of Information Theory*. McGraw Hill, 1958.
- [10] Guillermo A. Francia III and Tyler S. Gomez. Steganography obliterators: An attack on the least significant bits. In *Proc. InfoSecCD Conference06*, pages 85–91, Kennesaw, GA, 2006.
- [11] Neil F. Johnson and Sushil Jajodia. Exploring steganography: Seeing the unseen. *IEEE Computer*, 31(2):26–34, 1998.
- [12] Charles Kurak and John McHugh. A cautionary note on image downgrading. In *Proc. IEEE Eighth Annual Computer Security Applications Conference*, pages 153–159. IEEE, 1992.
- [13] Patricia A. Lafferty. *Obfuscation and the Steganographic Active Warden Model*. PhD thesis, Department Electrical Engineering and Computer Science, The Catholic University of America, 2008. to appear.

- [14] Xue-Bin Liang. On a conjecture of Majani and Rumsey. In *Proc. ISIT 2004*, page 62, Chicago, IL, 2004.
- [15] Keye Martin. How to randomly flip a quantum bit. *Preprint*, 2008.
- [16] Ira S. Moskowitz, Patricia A. Lafferty, and Farid Ahmed. A method for removing steganographic payloads from images. In *Proc. IEEE SMC IAW*, West Point, NY, June 2007.
- [17] Pierre Moulin and Ralf Koetter. Data-hiding codes. *Proceedings of the IEEE*, 93(12):2083–2126, December 2005.
- [18] Pierre Moulin and M. Kivanç Mihçak. A framework for evaluating the data-hiding capacity of image sources. *IEEE Transactions on Image Processing*, 11(9):1029–1042, September 2002.
- [19] Richard E. Newman, Ira S. Moskowitz, LiWu Chang, and Murali M. Brahmadേശam. A steganographic embedding undetectable by JPEG compatibility steganalysis. In F.A.P. Petitcolas, editor, *Information Hiding, Fifth International Workshop*, pages 259–277. Springer-Verlag, LNCS 2578, October 2002.
- [20] Richard E. Newman, Ira S. Moskowitz, and Mahendra Kumar. J2: Refinement of a topological image steganographic method. In *Proc. CNIS*, Berkeley, CA, September 2007. ACTA Press.
- [21] Claude E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423,623–656, 1948.
- [22] Derek Upham. Jpeg-jsteg-v4.0.
- [23] Andreas Westfeld. F5 — A steganographic algorithm. In I.S. Moskowitz, editor, *Information Hiding, 4th International Workshop*, pages 289–302. Springer-Verlag, LNCS 2137, April 2001.
- [24] Anthony Whitehead. Towards eliminating steganographic communication. In *Proc. 3rd Annual Conference on Privacy, Security, and Trust*, 2005.