Model-Based Steganography

Phil Sallee

University of California, Davis
Davis, CA 95616, USA
sallee@cs.ucdavis.edu
http://redwood.ucdavis.edu/phil

Abstract. This paper presents an information-theoretic method for performing steganography and steganalysis using a statistical model of the cover medium. The methodology is general, and can be applied to virtually any type of media. It provides answers for some fundamental questions which have not been fully addressed by previous steganographic methods, such as how large a message can be hidden without risking detection by certain statistical methods, and how to achieve this maximum capacity. Current steganographic methods have been shown to be insecure against fairly simple statistical attacks. Using the model-based methodology, an example steganography method is proposed for JPEG images which achieves a higher embedding efficiency and message capacity than previous methods while remaining secure against first order statistical attacks.

1 Introduction

Steganography, derived from the Greek words for 'covered writing', is the science of hiding information so that it remains undetected except by its intended receiver. It is necessary when one wishes to communicate privately without arousing the suspicion that would be caused by sending an encrypted message in plain view. The secret communication is hidden inside a larger message, referred to as the *cover message*, which can be transmitted without arousing any suspicion. The resulting message which contains the hidden content is referred to as the *stego message* or *steganogram*. A number of methods have been proposed for hiding messages in digital media including JPEG images, and MP3 audio files[8, 12,9]. Current methods generally encode the messages in the least significant bits (LSBs) of the cover media coefficients. While such LSB encoding is often not detectable by visual inspection, it can alter the statistical properties of the coefficients in easily detectable ways [12,11]. This is because by altering the LSBs indiscriminately, the marginal statistics (histograms) of the coefficient values will be changed in ways that make steganographic tampering evident.

By reducing the size of the message, these kinds of statistical signatures can be made less evident. Obviously, however, one would prefer to use a steganography method that is secure despite having a large capacity, where capacity is defined as the ratio between the size of the message and the size of the cover data in which it is hidden [12]. Recently, some methods have been devised which offer reasonably high capacity steganography while attempting to preserve the marginal statistics of the cover coefficients. One such method for encoding messages inside JPEG images is F5 [12]. Rather than flipping LSBs to encode the message bits, F5 increments and decrements coefficient values in order to maintain coefficient histograms that appear unaltered. However, it has been shown that F5 still changes the histograms of the coefficients in a detectable way. By estimating the original histograms of the coefficients from a cropped and re-JPEG'd version of the image, differences between the steganogram's histograms and the estimated original histograms become evident [7]. Another method which preserves marginal statistics more successfully is the OutGuess algorithm[9]. OutGuess reserves around half of the available coefficients for the purpose of correcting the statistical deviations in the global coefficient histogram caused by changing LSBs in the other half. For example, if a coefficient's value was moved from histogram bin A to bin B during the encoding process, another coefficient has to be moved from bin B to bin A to correct this change. While this is effective at maintaining the global histogram of the coefficients, it reduces the capacity by about half.

This raises the following types of questions: Is it possible to avoid detection by attacks that rely on marginal statistics of the coefficients without sacrificing half of the message capacity? What is the maximum message size that can be embedded in a given cover medium without risking detection? How can we achieve this maximum capacity? For answers, we turn to a new methodology based on statistical modeling and information theory. This paper presents a general framework for performing steganography and steganalysis using a statistical model of the cover media. To demonstrate the value of the model-based approach, an example steganography method is proposed for JPEG images which achieves a higher message capacity than previous methods while remaining secure against first order statistical attacks.

2 General Methodology

2.1 Compression and Steganography

Before describing the details of the model-based approach, it is helpful to first discuss the relationship between compression and steganography. This relationship has been previously discussed in [1] but it is useful to review it here. Suppose we had a method for perfect compression of some cover media, such as images taken from the real world. Thus, we could feed our compressor random scenes from our world and it would return perfectly compressed, truly random bit sequences (containing no statistical regularities) for each image. This is only possible if our compressor has available to it a complete and perfect model of the statistical properties found in natural scenes. Every statistical redundancy, every predictable quality, must be taken into account in order to accomplish this task - edges, contours, surfaces, lighting, common objects, even the likelihood of finding objects in certain locations.

We could, of course, place these compressed bit sequences in the corresponding decompressor to get back our original images. But suppose we instead had the idea to put our own random bit sequences into the decompressor. Out would come sensible images of the real world, sampled from the machine's perfect statistical model. Nothing would prevent us from also putting compressed and encrypted messages of our own choosing through the decompressor and obtaining for each message an image which should arouse no suspicion whatsoever were we to send it to someone. Assuming that our encryption method produces messages that appear random without the proper key, and that our intended receiver has the same image compressor we do, we will have perfectly secure steganography. Steganography is considered perfectly secure if there is no statistical difference between the class of cover messages and the class of stego messages [3].

Granted, this is decidedly unhelpful in that we cannot hope to obtain such a compression machine. But let us now consider a different approach that uses the same concept of decompression for practical steganography without the necessity of having a perfect model of the cover media. Assume instead that we have a model which captures some, but not all, of the statistical properties of the cover media. We can use a similar paradigm to provide steganography that is undetectable by all except those that possess a superior model of the cover media, or more specifically, a model which captures statistical properties of the cover media that are not captured by our model. This is accomplished by applying this decompression paradigm with a parametric model to replace only a least significant portion of cover media that has been sampled from the real world. The security of this steganography system will depend on the ability of the assumed model to accurately represent the distribution over cover messages. Specifically, such steganography will be *ϵ-secure against passive adversaries*, as defined by Cachin[3], where ϵ is the relative entropy between the assumed model and the true distribution over cover messages. Thus, this model-based approach provides a principled means for obtaining steganography that is provably secure in the information theoretic sense, insofar as the statistical model upon which it is based captures the statistical properties of the cover media.

As long as it remains possible that someone possesses a better model of the cover media, we cannot be sure that such steganography is completely undetectable. But if we consider a steganographic algorithm to be reasonably secure if it is not detectable by a specific statistical model, we can start to make some definitive statements regarding the maximum message length that can be securely hidden with this model and give a working strategy for obtaining this capacity. This approach will hopefully shift the emphasis which has up to now been placed on embedding methods towards more principled steganography methods based on statistical models. That is, we can start asking how to best model our cover data rather than trying to anticipate specific attacks or invent clever ways to flip least significant bits. And we can ask whether a steganographic method embeds messages optimally given its assumed statistical model. This provides us with a unifying framework with which to view and improve steganography and steganalysis methods.

2.2 Method

Let x denote an instance of a class of potential cover media, such as JPEG compressed images transmitted via the internet. If we treat x as an instance of a random variable X, we can consider the probability distribution $P_X(x)$ over transmissions of this class of media. Thus, if we transmit signals drawn from P_X, we can be assured that they are indistinguishable from similar transmissions of the same class regardless of how many such signals we transmit. Since P_X represents data taken from the real world, we can draw a valid instance from P_X using a digital recording device. Given such a sample, x, we separate it into two distinct parts, x_α which remains unperturbed, and x_β which will be replaced with x'_β, our encoded message. For LSB encoding, x_α represents the most significant bits of the cover coefficients as well as any coefficients not selected to send the message, and x_β represents the least significant bits of the selected coefficients. We can consider these parts as instances of two dependent random variables X_α and X_β. Using our model distribution $\hat{P}_X$, we can then estimate the distribution over possible values for X_β conditioned on the current value for X_α: $\hat{P}_{X_\beta|X_\alpha}(X_\beta|X_\alpha = x_\alpha)$. Provided that we select x'_β so as to obey this conditional distribution, the resulting $x' = (x_\alpha, x'_\beta)$ will be correctly distributed according to our model $\hat{P}_X$.

Now, in truth, it would appear that we haven't gained anything from this since we cannot model $P_{X_\beta|X_\alpha}$ perfectly any more than we could perfectly model P_X. However, we have gained something quite important. If we make a careful choice as to how X_α and X_β are separated, we can ensure that our changes to x_β are difficult or impossible to detect using the most sophisticated model of P_X on the planet: the human perceptual system. For instance, if we generate random samples from current image models, the result at best looks like 1/f noise or texture. But while the human visual system is fantastic at modeling images, it lacks a certain degree of precision. This lack of precision is what LSB encoding methods exploit. However, even the simplest models, such as those that capture the marginal statistics of X_β, do not lack this precision, and thus can be used to detect when LSBs are modified by some other distribution.

The solution proposed here is to use a parametric model of P_X to estimate $P_{X_\beta|X_\alpha}$, and then use this conditional distribution to select x'_β so that it conveys our intended message and is also distributed according to our estimate of $P_{X_\beta|X_\alpha}$. We can accomplish this task using the decompression paradigm previously discussed. Given a message M that is assumed to be compressed and encrypted so that it appears random, decompress M according to the model distribution $\hat{P}_{X_\beta|X_\alpha}$ using an entropy decoder, where x_α is part of an instance x drawn from the true distribution P_X via a digital recording device. While this cannot guarantee perfect security unless our model of P_X is perfect, it prevents all attacks except for those that use a better model of $P_{X_\beta|X_\alpha}$ than ours. Unless an attacker models statistical properties of X that we do not, or models them more accurately, our steganogram x' will contain the same measured statistical properties as others drawn from the true distribution P_X.

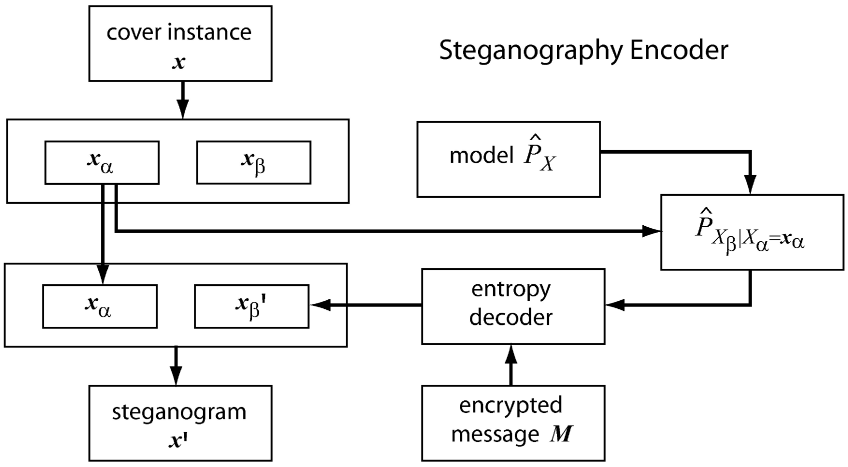

Fig. 1. Model-based steganography encoder: A cover x, such as an image, is split into two parts x_α (e.g. MSBs) and x_β (e.g. LSBs). A parametric model $\hat{P}_X$ over possible instances X is used to calculate the distribution over possible x_β instances given x_α. These probabilities are passed to an entropy decoder and used to decompress the encrypted message M, creating x'_β which is combined with x_α to create the steganogram.

Figure 1 illustrates the proposed model-based method for encoding steganography. First, an instance x of our class of cover media X is separated into x_α and x_β. x_α is fed to our model estimate of P_X which is used to compute the conditional probability distribution $P_{X_\beta|X_\alpha}$. The compressed and encrypted message M is given to an entropy decoder which uses $P_{X_\beta|X_\alpha}$ to decompress M resulting in a sample x'_β drawn from this distribution. The parts x_α and x'_β are then combined to form the steganogram x', distributed according to our model P_X which is transmitted to our receiver. Figure 2 illustrates the method used to recover the original message. Our steganogram x' is divided into x_α and x'_β. The x_α portion is fed into the model P_X which is again used to compute the condition distribution $P_{X_\beta|X_\alpha}$. Thus, the same model is given to the entropy encoder that was fed into the entropy decoder during the encoding stage. The entropy decoder returns the encrypted message. Assuming we have a key, we can decrypt the message and verify its contents. If on the other hand, we do not have the key, the encrypted message will appear random, which is the same result we would get from decoding an instance of X that does not contain steganography. Note that the encryption key does not necessarily need to be a private key. If we use a public key encryption method, we can just as easily obtain a method for public key steganography as suggested in [1].

2.3 Capacity

Determining how large a message can be hidden inside a cover message without becoming detectable has been a long unanswered question. If what we mean

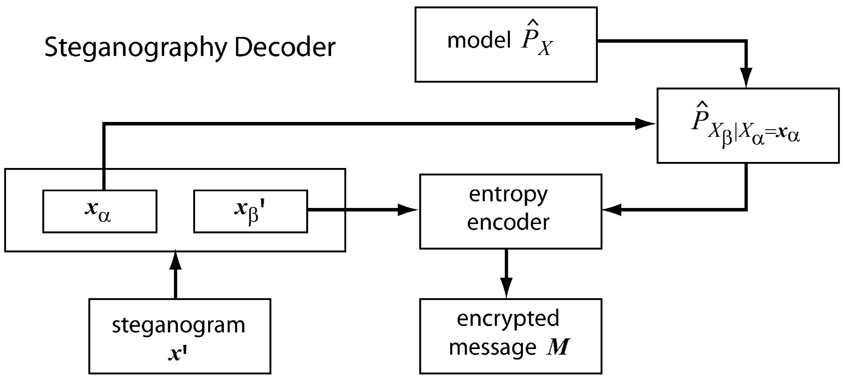

Fig. 2. Model-based steganography decoder: A steganogram x' is split into parts x_α and x'_β. A parametric model $\hat{P}_X$ is used to calculate the same probability distribution over possible x_β sequences that was used in the encoding process. x'_β is then fed into the entropy encoder which uses these probabilities to return the original message M.

by detectable is detectable by any method, this question remains unanswered as we would need to model P_X perfectly to ensure total security. However, we can estimate the average maximum message length that can be hidden without becoming detectable by our measured statistics of P_X. If we consider that X_β is being used as an information channel, we know from information theory that the maximum amount of information on average that can be transmitted through such a channel is equal to the entropy of the conditional distribution $\hat{P}_{X_\beta|X_\alpha}$:

$$H(X_\beta|X_\alpha = x_\alpha) = -\sum_{x_\beta} \hat{P}_{X_\beta|X_\alpha}(x_\beta|x_\alpha) \log_2 \hat{P}_{X_\beta|X_\alpha}(x_\beta|x_\alpha) \tag{1}$$

Using our model, this capacity limit can be measured for a given x_α. We can also see that our encoding method will be able to encode messages with this length on average, since an entropy encoder is designed to achieve this limit. Note that this limit is a function of x_α, and thus may vary depending on the content of our particular cover instance x.

2.4 Implicit Models Used by Current Methods

With this framework, we can gain a deeper understanding of current steganography methods. For instance, we can view current methods which encode messages in the coefficient LSBs at a rate of one bit per coefficient as equivalent to the model-based approach but using an implicit model that assumes the coefficients of their cover media are statistically independent, and that each coefficient is distributed according to a uniform distribution. Any other distribution would not have an equal probability of a 0 or 1 for every LSB. In this case x_β represents the LSBs of the coefficients, and the entropy decoder will simply copy bits of the message into the LSBs of the coefficients. Since coefficient histograms are definitely not uniform, one can easily see that encoding at such a rate must result in a significant change to the marginal statistics of the coefficients. While methods such as OutGuess attempt to compensate for this change by making compensatory changes to extra coefficients, we can see that this approach is not guaranteed to obtain maximum capacity, and is likely to reduce the capacity we could achieve if we incorporate the true marginal distributions of the coefficients into our model.

2.5 Steganalysis

We can also use this framework to perform steganalysis. If we have a target instance x, and we suspect that a steganographic system encoded a message into x_β using a weaker model of P_X than ours, we can measure the negative log likelihood of x_β given x_α under our model: $-\log_2 \hat{P}_{X_\beta|X_\alpha}(X_\beta = x_\beta | X_\alpha = x_\alpha)$, which has an expected value equal to the entropy $H(X_\beta | X_\alpha = x_\alpha)$, our expected message length. While this likelihood value can be computed directly, an equivalent method is to use the steganographic decoding process we have already described on x and measure the length of the resulting "message". It is easy to see why this works. If the statistics of x_β violate our expectation according to our model, we can expect a significantly longer message string. This is because the entropy coder assigns longer compressed bit sequences for x_β values which are less probable according to our model.

3 Applying the Methodology to JPEG

In order to demonstrate how the model-based methodology works in practice, we will now describe an example steganography system that is applied to compressed images stored in the file format defined by the Joint Photographic Experts Group (JPEG). Although JPEG is undoubtedly not the best compression format available, it is chosen for this demonstration because of its abundant use in email transmissions and on public internet sites. While the discussions from this point on will be aimed specifically at this JPEG implementation, the method used here can be easily applied to other file formats. In the JPEG compression standard, images are broken into 8x8 blocks. Each pixel block is passed through a 2-dimensional DCT (Discrete Cosine Transform) to produce 64 DCT coefficients for each block. Compression is accomplished by quantizing these DCT coefficients and then encoding them using a Huffman (or other entropy) encoder. The amount of compression is determined by the quantizer step size used before the entropy encoding, which is lossless.

The method described here is not intended to be secure against any known attack, but rather is primarily intended to demonstrate the methodology described in the previous section. We will use a fairly simple model which captures only the marginal statistics of the quantized DCT coefficients. Our total image model, then, assumes that images are generated by statistically independent

DCT coefficients. While this takes into account some correlations between image pixels, it is still a very limited image model as it does not describe higher order dependencies or even correlations across 8x8 blocks. It is expected that more complete image models which take into account joint statistics of the DCT coefficients would provide better steganographic security, and could also be used to attack this method. An example of such an attack is described by Farid and Lyu[5], who detect steganographic content by examining the marginal statistics of wavelet coefficients. Since the wavelet basis is much better than the DCT basis at describing the structure found in images, this would describe certain dependencies present between DCT coefficients. Taking into account joint statistics while encoding a message into DCT coefficients appears difficult, however, and so to some degree we are limited in our steganographic security by the image model imposed by our choice of cover media. If these methods were applied to a wavelet compression format such as JPEG 2000 instead of JPEG, however, it would provide resistance to attacks which use marginal statistics of wavelet coefficients.

3.1 Model

As with many steganographic methods, we will modify the least significant portions of the coefficients to encode our hidden information. Our model will consist of a parametric description of the marginal DCT coefficient densities. Because the DC coefficients (which represent the mean luminance within a block) are not well characterized by a parametric model, and because modifications to these coefficients are more likely to result in perceptible blocking artifacts, we will use only the AC coefficients during the encoding. Zero valued coefficients are also skipped for the encoding, because these often occur in featureless areas of the image where changes are most likely create visible artifacts. The remaining AC coefficients are modeled using the following parametric density function, which is a specific form of a Generalized Cauchy distribution:

$$P(u) = \frac{p-1}{2s}(|u/s|+1)^{-p} \tag{2}$$

where u is the coefficient value and $p > 1, s > 0$. The corresponding cumulative density function is

$$D(u) = \begin{cases} \frac{1}{2}(1+|u/s|)^{1-p} & \text{if } u \leq 0, \\ 1 - \frac{1}{2}(1+|u/s|)^{1-p} & \text{if } u \geq 0 \end{cases} \tag{3}$$

Other probability distributions, such as the generalized Laplacian distribution [10], have also been used to describe coefficient histograms that are peaked at zero. The distribution used here was chosen because it appeared to provide a better fit to the AC coefficient histograms, particularly in the tails of the distribution, and also because there is a closed form solution for its cumulative density function. This allows more precise fitting of the distribution to coefficient histograms and provides an efficient means of computing the probabilities for each histogram bin.

The first step in the embedding algorithm is to compute low precision histograms (with bin size > 1) of *each type* of AC coefficient for a cover image x. We will call the bin size of the low precision histogram the embedding step size. Each coefficient value is represented by a histogram bin index and a symbol which indicates its offset within the bin. If the embedding step size is 2, for instance, there will be two possible offsets within each nonzero bin. The zero bin is restricted to a width of 1 because we are skipping zero valued coefficients. The bin indices for all the coefficients comprise x_α, which will remain unchanged, and the bin offsets will comprise x_β which will be changed to encode our message.

For each image, the model parameters s and p are fit to these low precision histograms we have computed. The distributions are fit to only the most significant information in the coefficients because it is critical that both the encoder and the decoder compute the same estimated probabilities. The steganography decoder cannot know the least significant portions of the original coefficients as these may have been altered by the encoder. We fit the model parameters s and p to a histogram h of the coefficients by maximizing the likelihood $P(h|p, s)$ that the coefficients were generated from the model. During embedding, the coefficients are altered only within these low precision histogram bins (only the bin offsets are changed) so that the same estimates for p and s for each coefficient type may be obtained by the decoder. Figure 3 shows the histogram of the (2,2) DCT coefficients for a sample image measured in log probability and the model density after being fit to the histogram using the maximum likelihood approach.

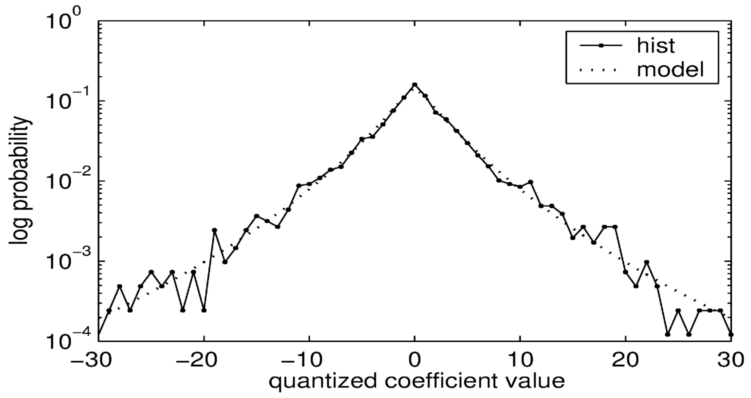

Fig. 3. Measured histogram (in log probability) of DCT coefficient (2,2) for the goldhill image, and the model pdf with parameters $s = 18.28$, $p = 6.92$.

3.2 Embedding Method

Once the model is fit to the histograms for an image, it is used to compute the probability of each possible offset symbol for a coefficient given its bin index. These offset symbols, and their respective probabilities are passed to a non-adaptive arithmetic entropy decoder [4] along with the message we wish to embed in the cover image. The offset symbols returned by the entropy decoder comprise x'_β which are combined with the bin indices to compute the coefficient values of the steganogram x'. To avoid visual attacks caused by changing coefficients only in part of the image, the order in which coefficients are used for encoding the message is determined by computing a pseudo-random permutation seeded by a key. This technique is known as permutative straddling [12]. If we run out of symbol probabilities before running out of message bits, we have reached our maximum message length for this image. In order to anticipate when this will happen, we can obtain the average maximum message length by computing the entropy of our symbol frequencies. If the message is shorter than the maximum message length, any remaining symbols are assigned according to the original coefficient offsets so that these coefficients remain unchanged.

A similar process is used to decode the message from the steganogram, except that the bin offset symbols x'_β in the steganogram are passed along with the symbol probabilities to an arithmetic encoder. Assuming the message length is encoded into the message, we can stop once the end of the message is reached. The algorithms for embedding and retrieving the message are outlined below:

Outline of the embedding algorithm

1. Given a cover image in JPEG format, and an encrypted message, generate low precision (bin size > 1) histograms of coefficient values. This information comprises x_α.
2. Fit the p and s parameters of our parametric model to each histogram by maximum likelihood.
3. Assign symbols to represent the offset of each coefficient within its respective histogram bin. These symbols comprise x_β. Compute the probability of each possible symbol for each coefficient using the model cdf.
4. Choose a pseudo-random permutation to determine the ordering of the coefficients.
5. Pass the message, and the symbol probabilities computed in step 3 in the order specified by step 4 to a non-adaptive arithmetic decoder in order to obtain symbols specifying the new bin offsets for each coefficient. The resulting symbols comprise x'_β.
6. Compute the new coefficients from the histogram bin indices (x_α) of the symbol offsets (x'_β).

Outline of the decoding algorithm

1-4. Same as embedding algorithm steps 1-4.

5. Pass the symbols and symbol frequencies obtained in steps 1-4 to the non-adaptive arithmetic encoder to obtain the original message.

Embedding step sizes. An embedding step size of 2 roughly corresponds to LSB encoding since each nonzero AC coefficient can take on one of two new values. Larger embedding step sizes will increase the message capacity (still without altering the marginal statistics) by sacrificing some image quality. If the cover media is not highly quantized, a higher embedding step size can be used before image quality becomes noticeably diminished. This provides a convenient means of increasing message capacity. However, transmitting images that are not very highly compressed may arouse suspicion in some situations.

Arithmetic encoding. The model-based approach described here requires an entropy codec. We used a non-adaptive arithmetic encoding method altered from the arithmetic encoding algorithm published in [13] to accept frequencies passed for each symbol rather than estimating them adaptively. For another example of non-adaptive entropy coding see [2], or refer to [4,13] for details on arithmetic encoding methods.

Embedding efficiency. One way to demonstrate the effectiveness of the model-based approach is to calculate the embedding efficiency. Embedding efficiency is the average number of message bits embedded per change to the coefficients [12]. It is generally assumed that the more changes that are made to the coefficients, the easier on average it will be to detect the steganography. Thus, we would like to minimize the number of these changes for a particular message length. In the model-based approach, the embedding efficiency will be determined by the entropy of the symbol distributions. Let us assume for now that we are using an embedding step size of 2, since that is most comparable to other steganography methods. We can show that the embedding efficiency of the model-based method described here will always achieve an embedding efficiency greater than or equal to 2, regardless of the message length.

Let k represent the probability of one of the two offset symbols for a given coefficient. The average number of bits we will encode, or the embedding rate, is equal to the entropy of the channel: $H = -(k \log_2 k + (1-k)\log_2(1-k))$. The probability that the value of the coefficient will be changed by encoding a different symbol than the original one, the rate of change, is $k(1-k) + (1-k)k = 2k(1-k)$. The expected embedding efficiency is the ratio of these two rates,

$$E[\texttt{efficiency}] = \frac{-(k \log_2 k + (1-k)\log_2(1-k))}{2k(1-k)} \tag{4}$$

which is never smaller than 2 for $0 < k < 1$. If $k = \frac{1}{2}$, the embedding efficiency will be exactly 2 because we will encode our message at a rate of 1 bit per coefficient and will be changing a coefficient from its original state half of the time. Otherwise, the embedding efficiency will always be greater than 2 and will be the largest (and the capacity the smallest) when the symbol probabilities are furthest apart. The F5 algorithm uses a technique known as matrix encoding to obtain an arbitrarily high embedding efficiency by reducing the message capacity.

However, for its maximum message capacity which is around 13%, the embedding efficiency is only 1.5 [12]. The OutGuess algorithm, since it must change about two coefficients on average for every other bit it embeds, has an embedding efficiency close to 1. In practice, we found that OutGuess provided an embedding efficiency of about 0.96 and a maximum message capacity of about 6.5%.

3.3 Results

Table 1 gives the results obtained from encoding maximal length messages in several grayscale test images using the proposed model-based method. While we tested our method on grayscale images, nothing prevents its application to color images. The images were first compressed to a JPEG quality factor of 80. This method does not double compress, which would leave a detectable signature on the coefficient histograms [6]. Instead the least significant bits of the coefficients are simply replaced, so the result steganogram maintains the same quantization tables as the original. The steganogram file size, message length and embedding efficiency for an embedding step size of 2 are shown for each image. Figure 4 shows the coefficient histograms of the DCT coefficient (2,2) before and after different steganography methods have been applied to the goldhill image. As can be seen, the F5 method greatly increases the number of zeros while the model-based method described here retains the shape of the original histogram. Note that they aren't expected to be identical to the original, since we are sampling from a model to select our coefficient values. The maximum allowable message length was used for each method during the comparison.

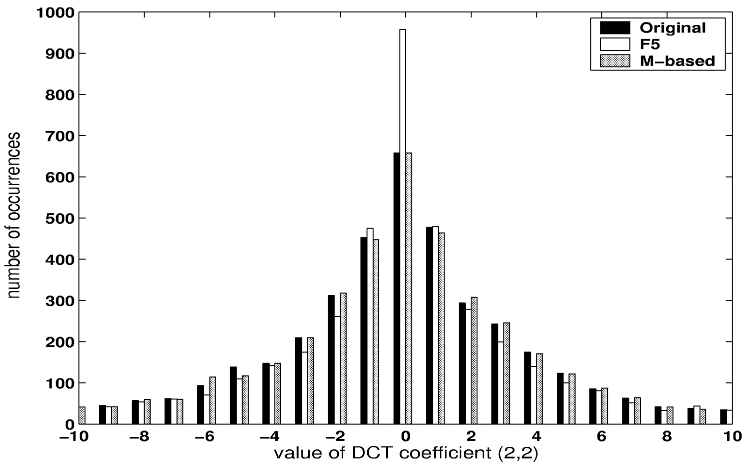

Fig. 4. A comparison of the coefficient histograms after different embedding methods. Each bar represents the number of occurrences for each value of the DCT coefficient (2,2). Shown are histograms for the original image, an image with 4984 bytes embedded using F5, and an image with 6544 bytes embedded using the model-based method.

Table 1. Results from embedding maximal length messages into several 512x512 grayscale JPEG images with an embedding step size of 2. Files were compressed using JPEG quality factor of 80 and optimized Huffman tables.

Image name	File size (bytes)	Message size (bytes)	Capacity	Embedding Efficiency
barb	48,459	6,573	13.56%	2.06
boat	41,192	5,185	12.59%	2.03
bridge	55,698	7,022	12.61%	2.07
goldhill	48,169	6,607	13.72%	2.11
lena	37,678	4,707	12.49%	2.16
mandrill	78,316	10,902	13.92%	2.07

4 Summary and Conclusions

We have presented a new model-based approach to steganography. This approach provides a unified framework for understanding steganography and steganalysis methods. We have shown that it is possible to determine the maximum length message that can be hidden without detection by a given model, and have described a general methodology by which this maximum message length may be obtained. It is hoped that this will encourage future research to focus on developing and applying advanced statistical models rather than on ad hoc embedding methods. As a proof of concept, we have demonstrated how to apply the model-based methodology to JPEG images using a model which captures marginal statistics of the DCT coefficients. The resulting algorithm achieves higher embedding efficiency than current methods while maximizing message capacity, and is resistant to first order statistical attacks. For example, it can embed twice as long a message as the OutGuess algorithm while changing fewer coefficients, and unlike OutGuess maintains not only global coefficient histograms but individual coefficient histograms as well.

5 Future Work

This line of investigation is open to a number of different directions. For instance, the algorithm described here for JPEG images can be readily applied to many other types of cover media such as MP3 audio files, video, or image formats other than JPEG. Primarily, however, we will focus on making the steganographic method more secure (and the steganalysis more powerful) by improving the statistical model to capture some joint statistical dependencies of the cover coefficients. For instance, we will consider a model conditioned on a local neighborhood of coefficients. Current steganography methods for JPEG images can be detected using attacks which measure increases in "blockiness" which occur during embedding [6]. We will investigate methods for defending against such attacks. A simple way to improve the underlying statistical model is to use a better representation for the cover media. It may be possible to extend this method in order to embed in the wavelet domain, even if the resulting image will be compressed using a different transform. However, this would require that the embedding method is robust against compression. The method presented here is particularly fragile because a change to even one coefficient will cause the arithmetic encoding process to produce a completely different message after that point. It may be possible, however, to use a different entropy encoding method to make the system more robust. One approach is to reset the encoding process at regular intervals and add redundancy to the message for error correction. This approach may also provide a method for defending against active attacks, in which the image may be altered by the adversary in order to prevent the hidden communication from reaching its intended receiver.

Acknowledgments. Supported by NIMH MH57921 (Bruno A. Olshausen) and MH60975 (E.G. Jones / B.A. Olshausen), and DOE-GAANN #P200A980307.

References

1. Anderson, R.J., Petitcolas, F.A.P.: On the Limits of Steganography. IEEE Journal of Selected Areas in Communications: Special Issue on Copyright and Privacy Protection), 16(4) (1998) 474–481
2. Buccigrossi, R.W., Simoncelli, E.P.: Progressive Wavelet Image Coding Based on a Conditional Probability Model *Proceedings ICASSP-97*, Munich Germany (1997)
3. Cachin, C.: An Information-Theoretic Model for Steganography *Proceedings of 2nd Workshop on Information Hiding*, LNCS, Springer (1998)
4. Cover, T., Thomas, J.: *Elements of Information Theory.* Wiley, New York, (1991)
5. Farid, H., Lyu, S.: Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines. In: Petitcolas, F.A.P. (Ed.): *Inf. Hiding: 5th Intl. Workshop.* LNCS 2578. Springer-Verlag, Berlin Heidelberg (2003) 340–354
6. Fridrich, J., Goljan, M., Hogea, D.: Attacking the OutGuess. *Proc. ACM: Special Session on Multimedia Security and Watermarking*, Juan-les-Pins, France (2002)
7. Fridrich, J., Goljan, M., Hogea, D.: Steganalysis of JPEG Images: Breaking the F5 Algorithm. In: Petitcolas, F.A.P. (Ed.): Inf. Hiding: 5th Intl. Workshop. LNCS, 2578. Springer-Verlag, Berlin Heidelberg (2003) 310–323
8. Petitcolas, F.: MP3Stego (1998) `http://www.cl.cam.ac.uk/~fapp2/steganography/mp3stego`
9. Provos, N.: Defending Against Statistical Steganalysis. In: *Proc. 10th USENIX Security Symposium.* Washington, DC (2001)
10. Simoncelli, E.P., Adelson, E.H.: Noise Removal Via Bayesian Wavelet Coring. *3rd IEEE Int'l Conf Image Processing.* Lausanne, Switzerland (1996)
11. Westfeld, A., Pfitzmann, A.: Attacks on Steganographic Systems. In: Pfitzmann A. (Ed.): *Inf. Hiding: 3rd Intl. Workshop.* LNCS 1768. Springer-Verlag, Berlin Heidelberg (2000) 61–75
12. Westfeld, A.: High Capacity Despite Better Steganalysis (F5 - A Steganographic Algorithm). In: Moskowitz, I.S. (Ed.): *Inf. Hiding: 4th Intl. Workshop.* LNCS 2137. Springer-Verlag, Berlin Heidelberg (2001) 289–302
13. Witten, I. H., Neal, R. M., Cleary, J. G.: Arithmetic coding for data compression, *Communications of the ACM*, 30(6) (1987)