Resampling and the Detection of LSB Matching in Colour Bitmaps

Andrew D. Ker

Oxford University Computing Laboratory, Parks Road, Oxford OX1 3QD, England

ABSTRACT

We consider the problem of detecting the presence of hidden data in colour bitmap images. Like straightforward LSB Replacement, LSB Matching (which randomly increments or decrements cover pixels to embed the hidden data in the least significant bits) is attractive because it is extremely simple to implement. It has proved much harder to detect than LSB Replacement because it does not introduce the same asymmetries into the stego image. We expand our recently-developed techniques for the detection of LSB Matching in grayscale images into the full-colour case. Not everything goes through smoothly but the end result is much improved detection, especially for cover images which have been stored as JPEG files, even if subsequently resampled. Evaluation of steganalysis statistics is performed using a distributed steganalysis project. Because evaluation of reliability of detectors for LSB Matching is limited, we begin with a review of the previously-known detectors.

Keywords: Steganography, Steganalysis

1. INTRODUCTION: LSB STEGANOGRAPHY

The art of covert communication, the aim of steganography is to convey messages in secret by concealing their very existence under digital cover media such as images, audio or video files. The competing goal of steganalysis is to detect, as reliably as possible, the presence of hidden data. In this paper we focus on colour bitmap images, taking the role of the steganalyst and developing more reliable detectors for the presence of hidden data.

A popular steganography technique in such media is Least Significant Bit (LSB) Replacement, which combines high capacity with visual imperceptibility and extreme ease of implementation (this last a significant part of its appeal). However there is now substantial literature on LSB replacement,[1–3] describing sensitive statistical methods for its reliable detection. In this paper we consider a minor modification of the LSB Replacement method, which we call *LSB Matching*. Other authors use different titles including *plus/minus 1 embedding*. LSB Matching retains the favourable characteristics of LSB Replacement but is more difficult to detect statistically.

After describing the LSB Matching algorithm, we consider detectors in the literature which do serve to unmask it. Two particular methods are due to Westfeld[4] and Harmsen,[5,6] and these are evaluated in Sects. 2 and 3 respectively. In recent work[7] we built on Harmsen's detector in novel ways, to produce detectors for LSB Matching in grayscale images. In Sect. 4 we extend these techniques to the full-colour case. There are technical obstacles to overcome and further new techniques are developed to deal with them. We perform thorough evaluation of the various detectors using a distributed steganalysis project.[8] In particular, we highlight the difference in detector reliability when steganography is performed on different types of cover image.

1.1. LSB Replacement and LSB Matching

We begin by describing the LSB Replacement and Matching algorithms. The technique of LSB Replacement is steganography folklore; LSB Matching was first described in Ref. 9. In each case the secret data is taken as a stream of bits, and the cover image is considered as a stream of bytes (separating the red, green, and blue components of each pixel in colour images). These bytes are taken in a pseudorandom order, as specified by a secret key which is presumed to be shared between sender and recipient of the stego image. This serves both to prevent the enemy steganalyst from reading the secret data straight off, and also to spread the secret data over the cover when there is less than the maximal amount.

Further author information: E-mail: adk@comlab.ox.ac.uk, Telephone: +44 1865 283530

In the case of embedding by LSB Replacement, the cover image bytes have their least significant bits replaced by the secret data, in order. In the LSB Matching embedding algorithm each secret data bit is compared with the least significant bit of the corresponding cover byte: in the case of a match, do nothing; in the case of a mismatch, the cover byte is incremented or decremented *at random* (except that one cannot decrement a zero pixel or increment a fully-saturated pixel).

The decoding algorithm is the same in either case: the recipient traverses the stego image in the order specified by the secret key and simply reads off the secret bits. The original cover image is not required and should be discarded by the sender. We emphasise that, in LSB Matching just as in LSB Replacement, it is *not* whether the cover pixel is incremented or decremented which encodes the hidden data. Rather, it is the stream of LSBs.

There is now much evidence that LSB Replacement is fairly easy to detect.[1–3] In the case of LSB Replacement in grayscale images, the most sensitive detectors of Ref. 3 can reliably detect data hidden at a rate of more than 0.05 bits per cover pixel, and in favourable cover images at rates as low as 0.005 bits per cover pixel. (The steganalyst cannot detect *all* covert communication, but they can force the steganographer either to communicate slowly or to take large risks.) Despite the similarities between the embedding methods, the success in finding reliable and sensitive detectors for LSB Replacement steganography has not been equalled for LSB Matching. It is not immediately obvious why LSB Matching should be any harder to detect, as both types of embedding add noise at the same level into the covers. The distinction is in the asymmetries inherent in LSB Replacement – a cover pixel with an even value might be left alone or might be incremented by one, but never decremented; conversely for odd-valued cover pixels – which are not present in LSB Matching. The LSB Replacement detectors of Refs. 1–3 are exploiting this asymmetry, and so are totally ineffective against LSB Matching.

Because they work on individual pixels in the spatial domain, LSB Replacement and Matching can only be used on bitmap images, either grayscale or colour. This may be thought a disadvantage, given the prevalence of the JPEG image format; however, spatial domain steganography remains of interest because of the extreme simplicity of embedding methods. This is not merely a matter of laziness on the part of the encoder: consider one situation in which steganography might be used, where the steganographer is attempting to smuggle data out of a secure environment. Because the environment is secure and, presumably, monitored, even the presence of steganography software is enough to convict the steganographer of wrongdoing. The steganographer dare not download and use a JPEG steganography program such as Outguess[10] because it would draw attention to them.

With this scenario in mind, in Ref. 3 we gave a program only 80 characters long, which could be typed directly into a Unix command line, to perform the very simplest LSB Replacement steganography. That particular program is not particularly secure because it does not use a secret key or spread the effects of steganography around the cover image. Here is a longer, but very much more secure, program:

```
perl -n0777 <cover-image >stego-image
     -e'split/(\s+)/,<STDIN>,5;@z=map ord,split"",pop@_;srand key;
        for(0..$#z){@p[$k,$_]=($_,$p[$k=int rand$_]);}
        map{$z[$q=shift@p]+=($z[$q]-ord()&1)*(rand 2<=>1)}
        split"",unpack"B*",$_;print@_,map chr,@z;' secret-text-file
```

Entered on a Unix command line (assuming that the Perl intpreter is present), this performs LSB Matching steganography in both grayscale and colour bitmaps. The first line decodes the image data from the cover image, which must be in the PGM/PPM format (and most Unix installations include tools to convert such files to and from other formats) and initialises the pseudorandom number generator. The second line generates a permutation of the cover. The third applies the LSB Matching algorithm to each pixel, and the last combines everything into a PGM/PPM output file. At 200 characters of code, this is still (just about) short enough to memorise, especially if the steganographer is familiar with Perl and can remember the program as something more than a sequence of gibberish*. Using this program, the steganographer can act without any steganography

*As one would expect from a 200 character program, errors are not handled gracefully. In particular, the program does not exclude the possibility of decrementing zero pixels or incrementing fully-saturated pixels, although a warning will be

software at all! Such short programs are only likely to be possible for spatial domain steganography, simply because of the complexities of the JPEG file structures. Hence our interest in spatial domain steganography.

In this paper, therefore, we consider only bitmap images. Although our main aim is a detector for LSB Matching in colour bitmaps, we will also refer to the simpler case of grayscale bitmaps along the way. Implicit are the assumptions that the resolution of the bitmap is 8 bits per component (most of the material can be carried over to other situations) and that hidden messages consist of a random-looking bitstream. The latter is likely to be the case if the messages are either compressed or encrypted.

1.2. Detectors for LSB Matching

We consider some possible detectors for LSB Matching. First, there is an elementary detector which looks for peaks at sample values of 1 and 254. It is easy to see that over- or underexposed cover images would give rise to such a peak. But in practice this artifact does not happen often enough under steganography, and happens relatively often when no steganography is present, that we can discard it as a viable method for detection.

A potentially powerful technique for the detection of (any type of) spatial domain steganography – but only for stego images whose cover image was a stored as a JPEG file – is called JPEG Compatability Analysis.[11] This makes use of the fact that there is a relatively small number of possible outputs from a JPEG decoder. Take an image to test for steganography. By guessing the JPEG quantization matrix, or trying every possibility, and simulating the quantization, we can find the nearest JPEG to that image. If there is a small difference between the image and its nearest JPEG, we suspect spatial-domain steganography, and the length and position of the hidden message may be deduced. For details we refer the reader to Ref. 11.

However, there are difficulties with JPEG Compatability Analysis. Firstly, according to Ref. 11, the technique becomes computationally infeasible for mildly-compressed JPEG images (and such images are typical outputs of digital cameras). And obviously the technique does not work if the cover was not a JPEG file, or if it undergoes even the mildest resampling or other image processing operations, before embedding. A more serious problem is that there is more than one way to decompress a JPEG file. Therefore one cannot say that an bitmap image is suspicious because it is almost, but not quite, the output of one particular JPEG decoder – the difference might be due to the JPEG decoder itself. For example, see Fig. 1. A small JPEG image has been decoded using the free decoder provided by the Independent JPEG Group[12] (IJG) and also using Adobe Photoshop CS. The difference between the two decoded images, with contrast enhanced so that the differences become visible, is also displayed. This difference is largely, but not entirely, due to the way Photoshop handles chroma subsampling. The same experiment is repeated on the same JPEG file encoded without subsampling. In this case the Photoshop-decoded image looks particularly suspicious, because it differs from the output of the free IJG decoder mostly in isolated pixels altered by plus or minus one – mimicking the effects of LSB Matching steganography.

We performed similar tests on other image processing software. Many programs make use of the same algorithm as the IJG decoder, but there are others, in addition to Photoshop, which differ. Furthermore, we even observed differences in JPEG decoding between different versions of Photoshop. Perhaps a human running the JPEG Compatability Analysis method can take account of these problems, but they do cast doubt on the practical applicability of the method, at least to automatic scanning for steganography.

Other detectors for steganography, applicable to LSB Matching, include those of Lyu and Farid,[13] Westfeld,[4] and Harmsen *et al.*[5,6] The first of these is a "blind" detector, and as such we expect its performance to be worse than specialised detectors aimed at a particular type of steganography. It has not been tested against a spatial-domain LSB Matching scheme and we do not consider it further in this paper. Westfeld's and Harmsen's detectors we turn to in Sects. 2 and 3. It is the latter which we build on, to generate novel and more sensitive detectors, in Sect. 4.

---

generated if this does happen. There may also be problems if the filesystem tries to translate carriage return/linefeed combinations automatically, but this can be avoided easily enough.

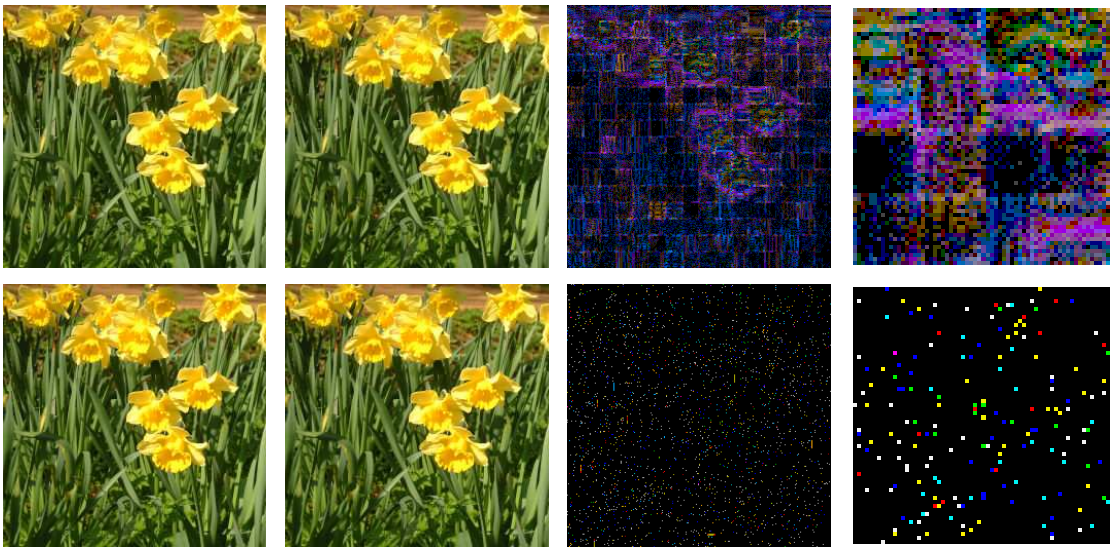

Figure 1. From left to right, a small (256×256) JPEG file decoded using the free IJG decoder and Adobe Photoshop CS, the contrast-enhanced difference, and a zoom of the contrast-enhanced difference. Above, image compressed using default JPEG encoding. Below, using JPEG encoding without chroma subsampling.

1.3. Evaluation Methodology

It is important to have confidence in steganography detectors. A "detector" is a discriminating statistic, a function of images which takes certain values in the case of stego images and other values in the case of innocent cover images. To quantify the reliability of detectors we use a distributed steganalysis project, first described in Ref. 8. We have obtained a number of large sets of digital images, totalling tens of thousands, which we use as covers. For a given length of hidden message, steganography is performed and the steganalysis statistics are computed on each image. By comparing the distributions of a steganalysis statistic in the cases of hidden data (for a particular amount of hidden data) and no hidden data, the reliability of that detector can be computed. Assuming that a detector aims only to give a binary diagnosis of steganography or no steganography (and the primacy of such an aim is discussed in Ref. 8), and that the detection statistic is a one-dimensional[†], the reliability is given by the *Receiver Operating Characteristic (ROC) curve*, which shows how the false positives and false negatives vary as the detection threshold is adjusted.

Because there are a number of steganalysis algorithms we wish to test, each with a number of possible variations, a number of hidden message lengths, and tens of thousands of cover images, there are millions of calculations to perform. To do so quickly, we use a small distributed network to undertake the computations; each node runs a highly-optimised program dedicated to the simulation of steganographic embedding and the computation of many different types of detection statistic; the calculations are queued, and results recorded, in a database from which ROC curves can be extracted and graphed. This distributed system has been used to analyse the detection of both LSB Replacement and LSB Matching steganography. At the time of writing 48 million rows of data have been produced, of which about 11 million are relevant to the detectors considered in this paper.

In practice, the performance of steganalysis methods is highly dependent on the types of cover images used. We will see a dramatic example in Sect. 2, where a detector is almost perfect on JPEG covers and almost useless on other types of cover. It is vital to take such factors into account when testing any steganalysis method; to test against only a set of images from a single source, no matter how large, is to risk an inaccurate and misleading result. Furthermore there is no single "representative" set of natural images for the purposes of

[†]if not, it must be converted into a one-dimensional statistic, using a multidimensional classifier – usually determined by training the detector on a test set of images. See Sect. 3 for an example of this.

testing steganalysis because one cannot specify a balance between images of different types. Our methodology is therefore to use a number of sets of cover images from different sources, and test each set separately. We identify three particular classes of cover images:

(i) JPEGs which have not been altered after decompression,

(ii) JPEGs which have been resized, or perhaps blurred or subject to other manipulations,

(iii) bitmaps which have never been stored as JPEGs.

Of course, the level of JPEG compression will matter, and for (ii) the severity of the resampling (or other manipulation) will be important. We will find that LSB Matching in images of type (i) can be detected quite easily, in a number of different ways. Images of type (ii) are harder to deal with, and it is on this type of image which the novel detectors presented here are particularly effective. It should be no surprise that LSB Matching is very difficult to detect in images of type (iii), even when very long messages are hidden, because such cover images can be expected to have very high entropy already; in this case our new detectors improve the state-of-the-art, but reliable detection of short messages remains difficult. We might expect that JPEG images very substantially reduced in size would have similar properties to uncompressed images; this is examined in Ref. 8, where it is shown that statistically significant differences can remain even when the dimensions are reduced by a factor of 3 or more.

In experiments, our primary set of cover images will be 3000 uncompressed full-colour bitmaps, from the *National Resources Conservation Service Photo Gallery*[14] which provides a large number of free downloadable photographs. The entirety of the library was downloaded, and some grayscale and anomalous images removed. These images are full-colour high-resolution (mostly 2100×1500) pictures, and most appear to be scanned from film. After downloading, each picture was reduced in size to 640 pixels wide (most becoming 458 pixels high) using a bicubic resampling scheme, with the aim of reducing any effects due to film granularity.

We will investigate the effects of JPEG compression, and subsequent resampling, by applying JPEG encoding to create new cover sets from these same images. The reason for using the same images to start with is to exclude the possibility that the content, colour balance, etc. affects the results: this way, the only differing factor will be JPEG compression. We apply a range of different JPEG compression "quality factors", although not all experimental results are reported (in the interests of brevity). Also, we will subsequently resample these JPEG images by factors of between 0.2 and 0.95, to see what happens when JPEG images are reduced in size.

To avoid the possibility of anomalous results due to the image content, we also test against four other sets of cover images, from royalty-free photo libraries purchased for this purpose. All four sets contain only JPEG images (high-resolution bitmap images being very difficult to acquire) but are otherwise varied in their type and size. They total around 50000 images. By and large, however, we have found that the set of 3000 bitmaps, additionally subject to subsequent JPEG compression and resampling, serves as a good representative cover image set, and for reasons of space we will only report the results for the larger sets of JPEGs when the results are particularly interesting.

2. DETECTORS BASED ON CLOSE COLOUR PAIRS

One of the earliest detectors suggested for LSB Matching is due to Westfeld.[4] It is not very thoroughly evaluated in the literature, so we briefly examine it here.

2.1. Westfeld's Detector

The detector is only applicable to colour images. It is founded on the assumption that cover images contain a relatively small number of different colours, in a very similar way to an early detector for LSB Replacement due to Fridrich *et al.*[15] The LSB Matching algorithm will turn a large number occurrences of a single colour into a cluster of closely-related colours. Consider a pixel colour as a triple (r, g, b), specifying the red, green, and blue components. In Ref. 15 the following definition is made: two colours (r_1, g_1, b_1) and (r_2, g_2, b_2) are a *close pair* if $|r_1 - r_2| \leq 1$, $|g_1 - g_2| \leq 1$, and $|b_1 - b_2| \leq 1$. Westfeld calls these pairs *neighbours*. For each colour occurring in an image, Westfeld's technique is to count the number of neighbour colours which also occur.

Each colour can have up to 26 neighbours (excluding itself). In cover images, the assumption is that most colours have few neighbours which occur in the image: Westfeld states that "a colour in a carrier medium has

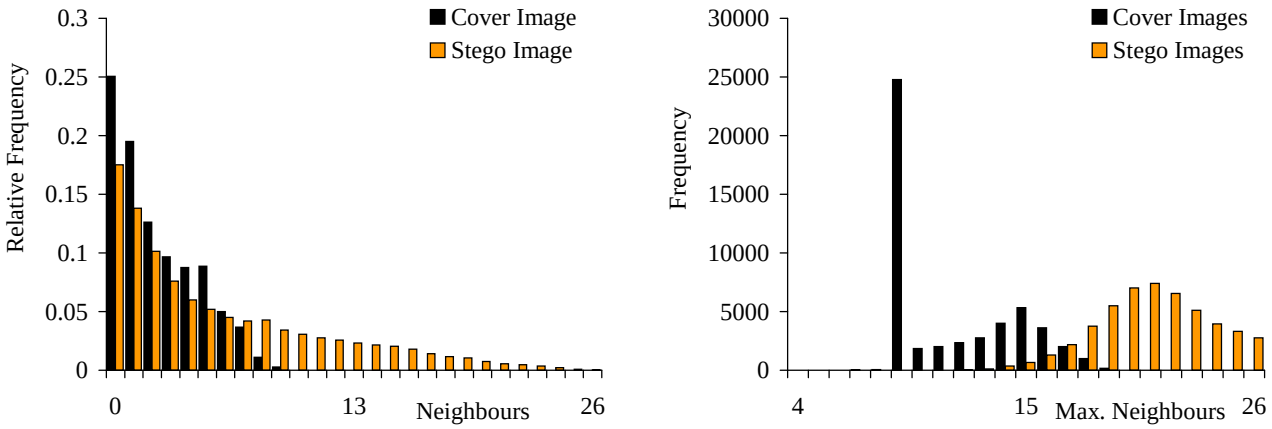

Figure 2. Left, relative frequency of the number of neighbours for each colour in a single image, before and after embedding a maximal-length message by LSB Matching. Right, histogram of the maximum neighbours statistic, for over 50000 JPEG images from various sources, before and after embedding a message 1% of maximal length.

only 4 or 5 neighbours on average" and that, in JPEG images, no colour has more than 9 neighbours. On the other hand, after embedding a message using LSB Matching (even when the message is quite small) enough new colours are created that the average number of neighbours is substantially increased, and many colours even have the full complement of 26 neighbours. For example, see Fig. 2, left. The number of neighbours of each colour in a JPEG image (the same image as in Fig. 1) has been computed, and the histogram displayed. The average number of neighbours for each colour is 2.20. This is repeated after embedding a maximal-length random message (3 bits per cover pixel) by LSB Matching; the average is now 5.58.

2.2. Performance Analysis

It is not entirely clear, from Ref. 4, exactly which discrimination statistic should be used to separate cover images from stego images. It should certainly *not* be the average number of neighbours for each colour occurring in the image: this statistic varies as much between cover images as it does when LSB Matching steganography is performed. A better statistic is the *maximum* number of neighbours that any pixel colour, which occurs in the image, has. We refer to this as the "maximum neighbours" statistic. For the JPEG used in Fig. 1, this statistic is 10 before, and 26 after, steganographic embedding. We also tested slightly more sophisticated alternatives, which take a weighted average number of neighbours, with most weight on the largest, but the improvements were at best incremental.

It should be no surprise that the cover image assumption, that each colour has only a small number of its possible neighbours occurring in the image, is key to the performance of this detector. It turns out that this assumption is true for JPEG images, but false for other images. In particular, it is false for JPEG images which have been even slightly modified by image processing operations such as re-sizing.

We give some results on the distribution of the discriminating statistic. Firstly, we tested the set of 3000 bitmaps after JPEG compression using quality factor 90. When no data is embedded, the most common maximum neighbours statistic is 10; the highest is 19. When a maximal-length hidden message is embedded, the maximum neighbours statistic is *always* 26 – there is always at least one colour with a full set of neighbours present. Therefore the detector is perfect, for this set of images with full embedding. The detector remains perfect when the hidden message is only 10% of the maximum. Even when a message only 1% of the maximum is embedded the detector still functions very well – it is at least 18 about 90% of the time, so there would be 10% false negative detections when setting a threshold of 18, and this leads to only 2.2% false positives.

We have perfomed experiments for a very large number of sets of cover images. Rather than include all the results in detail, we give a short summary. Roughly the same performance is observed no matter what level of JPEG compression is used, even when the "quality factor" is set to 100 (the mildest possible JPEG compression). It makes little or no difference to the results when chroma subsampling is turned off. Figure 2, right, shows the

histogram of the maximum neighbours statistic before and after embedding a message just 1% of the maximal length, for the 50000 JPEG images of widely varied type. Good discrimination is observed. But the story is quite different for cover images which are not JPEGs. In either the original set of 3000 bitmaps, or JPEG images subsequently resampled to 95% of the original size, over 99% of the cover images have 26 as the maximal neighbours statistic already (also, all reasonable JPEG quality factors and reduction ratios give the same result). We have a similarly hopeless situation if we try different statistics based on the number-of-neighbours counts.

Recall that we identified three classes of cover image: (i) JPEG images, (ii) resampled JPEG images, (iii) images which have never been subject to JPEG compression. We conclude that this detector is almost completely useless in cases (ii) or (iii). But in case (i) it is near-perfect, unless the hidden message is very short indeed. In this respect it is applicable in the same situations as JPEG Compatability Analysis. It is not so sensitive (JPEG Compatability Analysis can, in principle, detect the alteration of even a single pixel) but neither is it subject to the awkward problems caused by different JPEG decoders. It also works equally well when the JPEG compression is very mild. When given an image of unknown origin the detector can give a definite negative diagnosis (that no data is hidden) when the maximum neighbours statistic is low, but we can have no confidence in a positive diagnosis of steganography.

3. HISTOGRAM CHARACTERISTIC FUNCTION DETECTORS

The other detector for LSB Matching in the literature is due to Harmsen *et al.*[5,6] In fact, Harmsen does not mention LSB Matching at all – the detector is designed to work on any type of steganography which can be modelled as additive noise. But it is clear that LSB Matching is one such type. We begin by outlining Harmsen's method, on which we will build new detectors later.

3.1. Harmsen's HCF COM Detectors

Although Harmsen only uses the detector on colour bitmaps, its exposition begins with the simpler case of signals with a one-dimensional range (e.g. grayscale images). Consider a cover image, made up of pixels with intensity in the range $0,\ldots,N-1$ (usually N=256). Write $p_c(i,j)$ for the intensity of the cover image at location (i,j). We imagine that a maximal-length hidden message (one secret bit per cover pixel) is embedded using LSB Matching to form a stego image $p_s(i,j)$. Harmsen's detector uses only the relative frequency of occurrence of each intensity, i.e. the histogram. Write $h_c(n) = |\{(i,j)\,|\,p_c(i,j)=n\}|$ for the histogram of the cover image and similarly $h_s(n)$ for the histogram of the stego image after embedding. We model steganographic embedding as independent additive noise (mod N) and write f_Δ for the mass function of the noise random variable, which takes value n with probability proportional to $|\{(i,j)\,|\,p_s(i,j)-p_c(i,j)=n \pmod N\}|$.

Because the addition of integer random variables corresponds to the convolution of their mass functions, $h_s = h_c * f_\Delta$. Let $H_c[k]$, $H_s[k]$, and $F_\Delta[k]$ be the N-element DFTs of $h_c(n)$, $h_s(n)$ and $f_\Delta(n)$ respectively; then we have $H_s[k] = H_c[k]F_\Delta[k]$.

Harmsen calls $H_s[k]$ the *Histogram Characteristic Function (HCF)* of the stego image. In view of the symmetry of DFTs of real signals we only need consider $k = 0,\ldots,N/2$. The distribution of the added noise in the case of LSB Matching, when the hidden message is of maximal length, is just $f_\Delta(0) = 0.5$, $f_\Delta(\pm1) = 0.25$. Elementary calculation gives that $F_\Delta[k] = \cos^2\left(\frac{\pi k}{N}\right)$. More generally, if LSB Matching is used to embed a hidden message of proportion p of the maximal length, we have

$$H_s[k] = pH_c[k]\cos^2\left(\frac{\pi k}{N}\right) + (1-p)H_c[k] = H_c[k]\left(1 - p\sin^2\left(\frac{\pi k}{N}\right)\right). \tag{1}$$

Since this function will recur often we will write $\phi_p(k) = 1 - p\sin^2(\pi k/N)$, so that $H_s[k] = H_c[k]\phi_p(k)$. To diagnose the presence of steganography Harmsen uses the *centre of mass (COM)* of the HCF,

$$\mathcal{C}_1(H[k]) = \frac{\sum_{i=0}^{n} i|H[i]|}{\sum_{i=0}^{n}|H[i]|},$$

where $n = N/2$ to avoid redundant parts of the DFT. The key lemma of Ref. 5, stated here in simplified form, is the following.

LEMMA 3.1 (HARMSEN). *If α_k is a sequence with decreasing magnitudes then $\mathcal{C}_1(\alpha_k H[k]) < \mathcal{C}_1(H[k])$.*

Harmsen considers a general class of steganographic embedding methods, but they all have the property that the modulus of their HCF is monotone decreasing (as is $\phi_p(k)$, for $p > 0$), so that Lemma 3.1 applies, and hence $\mathcal{C}_1(H_s[k]) < \mathcal{C}_1(H_c[k])$. Steganography is diagnosed by a low HCF COM.

Note that Harmsen does not actually use the detector on grayscale images, and in Ref. 7 we have shown that it is in fact quite ineffective. In the case of colour images, a generalised COM $\mathcal{C}_m(H[k_1, \ldots, k_m])$ is used, a vector of m components of which the j^{th} is

$$\frac{\sum_{i_1,\ldots,i_m=0}^{n} i_j |H[\mathbf{i}]|}{\sum_{i_1,\ldots,i_m=0}^{n} |H[\mathbf{i}]|}.$$

In colour images each pixel takes a 3-dimensional value. The histogram is still defined as $h(\mathbf{n}) = |\{(i,j) \,|\, p(i,j) = \mathbf{n}\}|$ but here $\mathbf{n}$ is a vector of length 3. Similarly, the distribution of stego noise is 3-dimensional, $f_\Delta(\mathbf{n})$. We set $H[\mathbf{k}]$ to be the 3-dimensional DFT of the 3-dimensional array $h(\mathbf{n})$, and use the 3-dimensional COM $\mathcal{C}_3(H[\mathbf{k}])$. Note that Harmsen only considers the first octant of the DFT in the COM calculation; strictly speaking, the m-dimensional DFT is only symmetrical about one axis and so we are throwing away non-redundant data. Nonetheless, it makes sense to do so here, given the symmetry between the components (and to avoid aliasing effects in the high frequencies). Since the effect of LSB Matching is independent on each colour component, in this case we have $F_\Delta[k_1, k_2, k_3] = \phi_p(k_1)\phi_p(k_2)\phi_p(k_3)$ and so[‡] we have

$$\mathcal{C}_3(H_s[\mathbf{k}]) < \mathcal{C}_3(H_c[\mathbf{k}]) \tag{2}$$

(we interpret the inequality of vectors componentwise).

The three dimensional quantity $\mathcal{C}_3$ is used to discriminate between the presence and absence of steganography. In Ref. 5, the Bayesian multivariate classifier[16] is used: this is the likelihood ratio statistic for discriminating between two multivariate Gaussian sources. If $\boldsymbol{\mu}_1$ is the mean vector, and Σ_1 the variance-covariance matrix, of the first source (estimated, in the standard way, from a training set of cover images) and $\boldsymbol{\mu}_2$ and Σ_2 the corresponding parameters of the second source (estimated by simulating maxmimal-length steganography into the training set) then the log-likelihood ratio statistic of the vector $\mathbf{k}$ is

$$l(\mathbf{k}) = -\frac{1}{2}((\mathbf{k}-\boldsymbol{\mu}_1)^t \Sigma_1^{-1}(\mathbf{k}-\boldsymbol{\mu}_1) - (\mathbf{k}-\boldsymbol{\mu}_2)^t \Sigma_2^{-1}(\mathbf{k}-\boldsymbol{\mu}_2)) - \frac{1}{2}(\ln|\Sigma_1| - \ln\Sigma_2|).$$

Harmsen only considers the binary detector $l(\mathbf{k}) < 0$, but we can trade false positives for false negatives as usual, by setting different thresholds.

A disadvantage of this HCF COM discriminator is that it involves computing the DFT on an array of size N^3 (i.e. typically 256^3); the time complexity of the DFT is $O(N^3 \log N)$. In a sequel paper,[6] Harmsen suggests two alternative methods to deal with 3-dimensional colour signals with the aim of reducing the time and space complexity (we simplify the presentation a little in what follows). One method is to consider the red, green, and blue channels separately, computing individual 1-dimensional HCFs for each, $H_r[k]$, $H_g[k]$, $H_b[k]$. Then a 3-dimensional classifier is formed from the 1-dimensional COMs of each HCF,

$$\mathcal{C}^{3\times 1\mathrm{D}}(H_r[k], H_g[k], H_b[k]) = (\mathcal{C}_1(H_r[k]), \mathcal{C}_1(H_g[k]), \mathcal{C}_1(H_b[k]))\,.$$

Alternatively, consider the red, green, and blue channels in pairs, computing three sets of 2-dimensional HCFs ($H_{r,g}[k,l]$ for the 2-dimensional DFT of the joint histogram of red and green values, $H_{g,b}[k,l]$ and $H_{b,r}[k,l]$ similarly) and combining three 2-dimensional COMs into a 6-dimensional classifier by concatenating the pairs $\mathcal{C}_2(H_{r,g}[k,l])$, $\mathcal{C}_2(H_{g,b}[k,l])$, and $\mathcal{C}_2(H_{b,r}[k,l])$. The latter classifier will be referred to as $\mathcal{C}^{3\times 2\mathrm{D}}$.

The time complexity of $\mathcal{C}^{3\times 1\mathrm{D}}$ and $\mathcal{C}^{3\times 2\mathrm{D}}$ are $O(N \log N)$ and $O(N^2 \log N)$ respectively. In practice (when $N = 256$), the former is very roughly 10000 times faster to compute than the original classifier, and the latter roughly 100 times faster; some experiments are performed in Ref. 6.

[‡]here Harmsen implicitly uses a generalised version of Lemma 3.1 to multidimensional sequences which are decreasing in every dimension.

Table 1. Summary of performance, for the three variants of Harmsen's HCF COM. In each case, the set of 3000 uncompressed bitmaps is used, possibly JPEG compressed prior to embedding. The message length is the proportion of the maximum. A Bayesian multivariate classifier is trained using half of the images (repeating with no steganography and maximal-length steganography) and tested against the other half. The "summed" 3×2D-COM statistic needs no training.

Image Set	Message Size	False positive rate for 50% false negatives			
		1×3D COM	3×2D COM	3×1D COM	3×2D-COM summed
unaltered bitmaps	1.0	4.9%	8.0%	33.8%	8.5%
unaltered bitmaps	0.5	26.1%	20.5%	41.8%	22.0%
JPEG q.f. 90	1.0	0.0%	0.1%	34.9%	0.0%
JPEG q.f. 90	0.5	5.2%	2.4%	40.8%	1.6%
JPEG q.f. 90	0.2	25.5%	25.4%	46.3%	19.8%
JPEG q.f. 75	1.0	0.0%	0.1%	27.2%	0.0%
JPEG q.f. 75	0.5	0.8%	1.5%	37.3%	1.1%
JPEG q.f. 75	0.2	22.4%	20.9%	44.2%	18.7%

3.2. Performance Analysis

We evaluate the performance of Harmsen's original detector, and the quicker versions which use lower-dimensional DFTs. As part of our testing, we observed that the Bayesian multivariate classifier is somewhat of an unnecessary complication: if there is symmetry between the properties of red, green, and blue channels then there should be symmetry between the components of the COM, and so a suitable 1-dimensional statistic would be simply to sum the components of the 3- or 6-dimensional vector. This avoids the need for training.

While we have performed experiments on many sets of covers, with many different message lengths, we must condense our results in the interests of readability. We will just report the results from the set of 3000 bitmaps, unaltered and also subject to JPEG compression and subsequent resampling. Also, we only report results for Harmsen's three detectors and the single 1-dimensional "summed" statistic $\sum_i \mathcal{C}_i^{3\times 2\mathrm{D}}$ (the other summed statistics perform analogously). Finally, instead of displaying full ROC curves, we report just the false positive rate when the detector's false negative rate is set to 50%. In Ref. 3 we argue that this is a reasonable 1-dimensional measure of performance (motivated by the belief that false positives are more serious errors than false negatives, and observations of the shapes of ROC curves). Nonetheless, we must admit that the results reported here are only a fraction of those obtained, albeit a reasonable summary (in our view).

In Tab. 1 we show the results for each of the three HCF COM classifiers, plus the "summed" statistic $\sum_i \mathcal{C}_i^{3\times 2\mathrm{D}}$, for various cover sets and three different message lengths. We observe, as did Harmsen in Ref. 6, that $\mathcal{C}^{3\times 1\mathrm{D}}$ is a very poor detector, but that $\mathcal{C}^{3\times 2\mathrm{D}}$ is nearly as reliable as the original "1×3D" detector (in rare cases it is even better); the difference between these last two is greater for uncompressed bitmaps than for JPEGs. The "summed" classifier is nearly as reliable as the trained Bayesian multivariate classifier. Unlike in Ref. 6 we also test against shorter hidden messages, and observe that the performance of the COM detectors falls off sharply (in experimented not detailed here we found that the "tipping point" for hidden message length is around 30% in JPEG covers). We can conclude that these detectors form a reliable detector for LSB Matching steganography in JPEG ("type (i)") cover images, when the hidden message is fairly long. In uncompressed bitmaps ("type (iii)") they are only partially reliable, and only for near-maximal hidden messages.

In Tab. 2 we show the detectors' performance for cover images of "type (ii)", resampled JPEGs, for a hidden message length of 50% of maximum. The detectors retain their performance until the reduction factor becomes as small as 30%[§]. We conclude that, unlike Westfeld's detector, the COM detectors are effective in this case.

4. NOVEL DETECTORS

In recent work[7] we gave two methods to improve HCF COM detectors for grayscale images. In what follows, we will try to apply them to the colour case. The two new techniques address two potential weaknesses in HCF

[§]When the covers are reduced to 20% of their original size they are only 128 × 92 pixels. We should expect this to cause some of the performance drop-off: Ref. 3 notes that steganography appears to be harder to detect in small images.

Table 2. Summary of detector performance for the set of 3000 bitmaps, first JPEG compressed at quality factor 75 and then reduced in size (using a bicubic resampling algorithm) before embedding.

Resampling ratio	Message Size	False positive rate for 50% false negatives			
		1×3D COM	3×2D COM	3×1D COM	3×2D-COM summed
90%	0.5	0.4%	2.1%	37.7%	1.6%
70%	0.5	0.6%	1.8%	37.4%	2.2%
50%	0.5	0.7%	2.3%	36.2%	2.7%
30%	0.5	3.2%	8.8%	38.9%	9.2%
20%	0.5	5.0%	9.0%	45.6%	22.0%

COM detectors. The first is that the value of the HCF COM is essentially without context – it is difficult to say whether a particular value is "low" or "high" as it may depend as much on the type of cover image than the presence or absence of steganography. The second is that the HCF COM depends only on the histogram of the image and so is throwing away a great deal of structure. The two ways to address these weaknesses are:

Calibration: In the case of grayscale images, we made use of certain properties of the HCF COM when an image is divided by 2, i.e. resampled to half its original size. Just as we previously wrote p_c, h_c, and H_c, for the pixel values, histogram, and HCF of the cover image (and p_s, etc., for the stego image) we now write $p_{\frac{1}{2}c}$, $h_{\frac{1}{2}c}$, and $H_{\frac{1}{2}c}$ for the same functions of the halved image (and $p_{\frac{1}{2}s}$, etc., for the halved stego image). To halve the image, we use an averaging filter, so

$$p_{\frac{1}{2}c}(i,j) = \left\lfloor \sum_{u=0}^{1}\sum_{v=0}^{1} p_c(2i+u, 2j+v)/4 \right\rfloor . \tag{3}$$

In grayscale images, we observed that $\mathcal{C}_1(H_c[k]) \approx \mathcal{C}_1(H_{\frac{1}{2}c})$, but that $\mathcal{C}_1(H_s[k]) < \mathcal{C}_1(H_{\frac{1}{2}s}[k])$. The first of these is simply an observation of natural images; the second is not proved in Ref. 7 but we will give a proof here. The result is a detector *calibrated* by downsampling the image: by combining the two previous observations we deduce that $\mathcal{C}_1(H[k]) \approx \mathcal{C}_1(H_{\frac{1}{2}}[k])$ in cover images, but $\mathcal{C}_1(H[k]) < \mathcal{C}_1(H_{\frac{1}{2}}[k])$ in stego images.

Adjacency Histogram: Instead of considering just the histogram, use the adjacency histogram (also called the *co-occurrence matrix*), which determines how often each pair of intensities occur in adjacent pixels:

$$ah_c(m,n) = |\{(i,j)\,|\,p_c(i,j) = m, p_c(i,j+1) = n\}|.$$

Because adjacent pixels tend to have close intensities, this histogram is sparse away from the diagonal. Using a 2-dimensional DFT we form the adjacency HCF $AH[k,l]$ and take a 2-dimensional COM. In this case symmetry between the two components is assured, so we will certainly just want to sum the two components of the COM, leading to a 1-dimensional discriminator.

Neither of these methods seems to apply well to the case of full-colour images. In the case of calibration, we will see that it tends to make the detectors worse, not better, when the length of embedded message is not maximal. And we cannot make use of the adjacency histogram because it would be a histogram of size N^6 (typically 256^6) which cannot even be stored in the memory of today's computers, let alone have a DFT performed on it in a feasible amount of time! Nonetheless, these techniques will be seen to inspire better detectors for the full-colour case.

4.1. HCF COM Detectors Calibrated by Resampling

We begin by considering calibration, by downsampling the image according to (3) on each colour component. Firstly, we test the assumption that the COM for cover images is generally unchanged by downsampling in this way:

$$\mathcal{C}_3(H_c[\mathbf{k}]) \approx \mathcal{C}_3(H_{\frac{1}{2}c}[\mathbf{k}]). \tag{4}$$

This is probably unprovable, but we justify it by experimental evidence, as in Ref. 7. We cannot plot a scattergram of the COMs, because they are 3-dimensional, but we can illustrate the extent to which it holds by, for

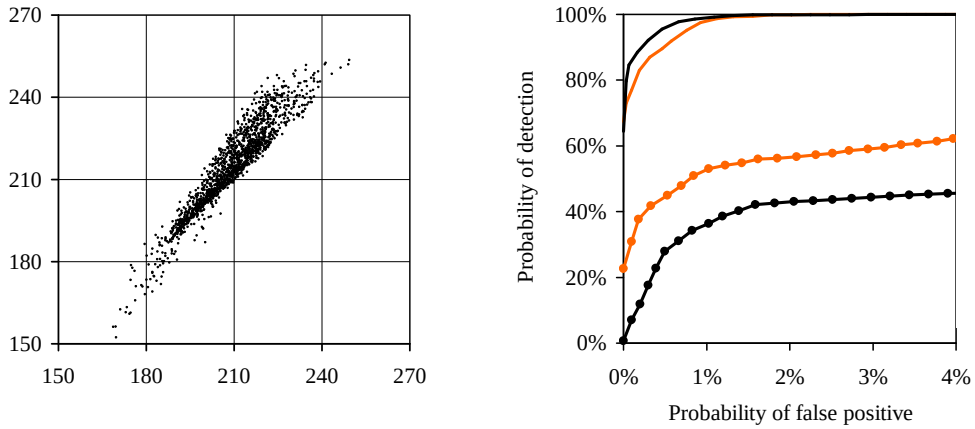

Figure 3. Left, scattergram of the sum of the components of $\mathcal{C}_3(H_c[k])$ (*x-axis*) against $\mathcal{C}_3(H_{\frac{1}{2}c}[k])$ (*y-axis*), for the set of 3000 bitmaps first subject to JPEG compression at quality factor 75. Right, ROC curves for the same set of images, with uncalibrated (*light lines*) and calibrated (*black lines*) HCF COM detectors, when LSB Matching steganography is used to embed a message of length 100% (*unmarked lines*) or 50% (*marked lines*) of maximum.

example, plotting a scattergram of the sum of the components of $\mathcal{C}_3(H_c[k])$ against the sum of the components of $\mathcal{C}_3(H_{\frac{1}{2}c}[k])$. One such scattergram is shown in Fig. 3, left; similar results are observed in other image sets, including the uncompressed bitmaps. There is a reasonable correlation between the two quantities, but it is not as strong as for the grayscale images in Ref. 7. This is the first sign that the grayscale techniques will not carry forward, so well, into the colour case.

We now prove the inequality necessary for the calibration method to be applied to detection of steganography. In Ref. 7 we reasoned informally that it was true because of the adding and rounding during the resampling operation, which tends to cancel out the effects of steganography. Here we give a proof.

LEMMA 4.1. *Interpreting inequality of vector componentwise,* $\mathcal{C}_3(H_s[\mathbf{k}]) < \mathcal{C}_3(H_{\frac{1}{2}s}[\mathbf{k}])$.

Proof. We must examine the effect, on the downsampled image, of performing LSB Matching steganography on the full-sized cover. Each colour component can be considered individually so that the signals involved are all 1-dimensional. Take a single pixel of the downsampled image, and suppose it is the result of adding and averaging pixels from the cover which has value a_1, a_2, a_3, and a_4. Call its value z before embedding, so that $z = \lfloor (a_1+a_2+a_3+a_4)/4 \rfloor$. After embedding suppose that a_i is now a_i' and $z' = \lfloor (a_1'+a_2'+a_3'+a_4')/4 \rfloor$. If a message of length p times the maximum is embedded by LSB Matching, we know that $a_i' - a_i$ takes the values ± 1 with probability $p/4$ and the value 0 with probability $1 - p/2$. Let us write $X = \sum_i a_i \bmod 4$ and $Y = \sum_i (a_i' - a_i)$. Then,

$$\begin{aligned}
P(z' = z+1) &= P(X=0 \wedge Y=4) + P(X=1 \wedge Y \geq 3) + P(X=2 \wedge Y \geq 2) + P(X=3 \wedge Y \geq 1) \\
&\quad \{\text{assuming that } X \text{ is uniformly distributed}\} \\
&= \tfrac{1}{4}\left(P(Y=1) + 2P(Y=2) + 3P(Y=3) + 4P(Y=4)\right) \\
&= \tfrac{1}{4}\left(4\left(1-\tfrac{p}{2}\right)^3\left(\tfrac{p}{4}\right) + 12\left(1-\tfrac{p}{2}\right)\left(\tfrac{p}{4}\right)^3 + 12\left(1-\tfrac{p}{2}\right)^2\left(\tfrac{p}{4}\right)^2 + 8\left(\tfrac{p}{4}\right)^4 + 12\left(1-\tfrac{p}{2}\right)\left(\tfrac{p}{4}\right)^3 + 4\left(\tfrac{p}{4}\right)^4\right) \\
&= \tfrac{1}{4}\left(p - \tfrac{3}{4}p^2 + \tfrac{3}{8}p^3 - \tfrac{5}{64}p^4\right) = \tfrac{q}{4}, \text{ say.}
\end{aligned}$$

By symmetry, $P(z' = z-1) = q/4$ also. It is routine to check that $q < p$ (for $0 < p \leq 1$). Therefore we can conclude that the effect of LSB Matching on the full-size cover image is noise of a similar type in the half-size image, but strictly less of it.

Recall (1), which in the case of colour images becomes $H_s[k_1,k_2,k_3] = H_c[k_1,k_2,k_3]\prod_{i=1}^3 \phi_p(k_i)$. Similarly, $H_{\frac{1}{2}s}[k_1,k_2,k_3] = H_{\frac{1}{2}c}[k_1,k_2,k_3]\prod_{i=1}^3 \phi_q(k_i)$. Now

$$\frac{\partial}{\partial k}\left(\frac{\phi_p(k)}{\phi_q(k)}\right) = \frac{(q-p)\sin\left(\frac{2\pi k}{N}\right)\left(\frac{\pi}{N}\right)}{\phi_q(k)^2} < 0.$$

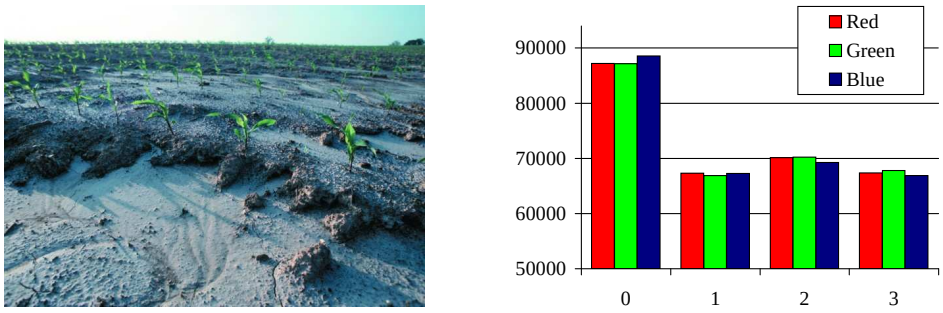

Figure 4. Left, a cover image with poor behaviour under the resampling-calibrated HCF COM. Right, histograms of the random variable X for this image, for each colour component.

We must slightly strengthen assumption (4) to $\mathcal{C}_3(H_c[\mathbf{k}]\prod_i \phi_p(k_i)) \approx \mathcal{C}_3(H_{\frac{1}{2}c}[\mathbf{k}]\prod_i \phi_p(k_i))$ (plausible because we expect $H_c[\mathbf{k}]$ to have roughly the same shape, as well as COM, to $H_{\frac{1}{2}c}[\mathbf{k}]$), to deduce

$$\begin{aligned}\mathcal{C}_3(H_s[\mathbf{k}]) &= \mathcal{C}_3(H_c[\mathbf{k}]\textstyle\prod_i \phi_p(k_i)) \\ &\approx \mathcal{C}_3(H_{\frac{1}{2}c}[\mathbf{k}]\textstyle\prod_i \phi_p(k_i)) \\ &= \mathcal{C}_3\left(H_{\frac{1}{2}s}[\mathbf{k}]\textstyle\prod_i \left(\frac{\phi_p(k_i)}{\phi_q(k_i)}\right)\right) \\ &< \mathcal{C}_3(H_{\frac{1}{2}s}[\mathbf{k}]),\end{aligned}$$

the final inequality by the multidimensional generalisation of Lemma 3.1. □

Combining (4) and Lemma 4.1, we have that $\mathcal{C}_3(H[\mathbf{k}]) \approx \mathcal{C}_3(H_{\frac{1}{2}}[\mathbf{k}])$ in cover images and $\mathcal{C}_3(H[\mathbf{k}]) < \mathcal{C}_3(H_{\frac{1}{2}}[\mathbf{k}])$ in stego images. We therefore want to divide one COM by the other for a dimensionless "calibrated" discriminator. Since we have 3-dimensional quantities, we can either divide pointwise or simply sum the components of each COM and divide one sum by the other (in practice it makes little difference to the reliability of the detector which of these is chosen; the second is preferable because it avoids the need for a multidimensional classifier).

The performance of this classifier is, however, disappointing. Two ROCs are displayed in Fig. 3, right. When a maximal-length hidden message is embedded in this particular set of covers (which are JPEGs) the calibrated detector performs a little better than the standard 3D COM. But when a hidden message only 50% of the maximum is embedded, the calibrated detector performs worse than the standard detector. Similar effects are seem whenever the covers are JPEGs or resampled JPEGs and the hidden message is less than maximal.

We investigate this effect with a particular example, the cover image in Fig. 4. The COM is $(68.9, 68.8, 68.7)$; that of the same image reduced by a factor of two is $(68.9, 68.8, 68.9)$ (so (4) holds in this case). When 50% steganography is embedded in the full-size cover, the COM is reduced to $(57.8, 57.3, 56.5)$ and that of the half-size image to $(53.5, 53.5, 53.7)$. This directly contradicts Lemma 4.1! The resolution to this apparent paradox is that the proof of this lemma includes the assumption that the random variable X, the sum of adjacent groups of 4 pixel intensities (mod 4), is uniform. Whilst this assumption is reasonable for scanned images, it often fails for JPEGs because of areas of flat colour or straight gradients, both of which tend to produce groups where $X = 0$. On the right of Fig. 4 the histograms for X are displayed. The lack of uniformity is evident. Lemma 4.1 must be disregarded – the calibration technique derived from it introduces error – in such cases, which are quite common.

4.2. Alternative Methods of Calibration

It is the rounding operator in (3) which causes Lemma 4.1 and the method of calibration to fail. The obvious alternative is not to do any dividing or rounding; in this case we are not downsampling and so we might as well consider pixels in pairs rather than groups of 4. Therefore we consider the following image:

$$p_{c'}(i,j) = p_c(2i,j) + p_c(2i+1,j) \tag{5}$$

(with this formula applied separately to each colour component). This image is a "squeezed" version of the original, with horizontal pairs of pixels added together. It is an operation conceptually very close to downsampling,

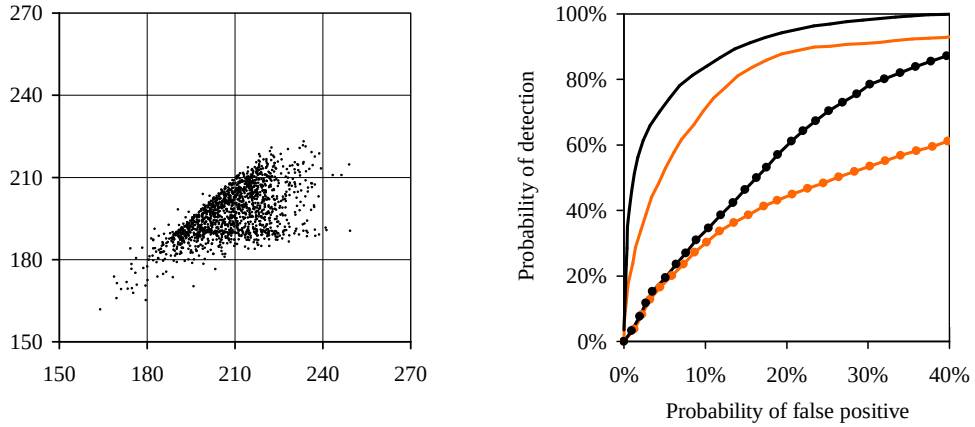

Figure 5. Left, scattergram of the sum of the components of $\mathcal{C}_3(H_c[k])$ (*x-axis*) against $\mathcal{C}_3(H_{c'}[k])$ (*y-axis*), for the set of 3000 bitmaps first subject to JPEG compression at quality factor 75. Right, ROC curves for the uncalibrated (*light lines*) and calibrated (*black lines*) HCF COM detectors, when LSB Matching steganography is used to embed a message of length 100% (*unmarked lines*) or 50% (*marked lines*) of maximum in 3000 uncompressd bitmaps.

but with a significant difference in that the possible range of intensities is now doubled. We use $p_{c'}$, $h_{c'}$, and $H_{c'}$, $p_{s'}$, etc, for the pixel values, histogram, and HCF of the squeezed images. Note that, in this case, the domain of the histogram is $0, \ldots, 2N-1$ (actually $2N-2$), so we use a $2N$-point DFT to produce $H_{c'}$.

We compute the COM of $H_{c'}[\mathbf{k}]$ based only on the first $(N/2)$ frequencies in each dimension, despite the fact that there are now twice as many non-redundant frequencies, in each dimension, as before. This ensures that the COM of $H_c[\mathbf{k}]$ is calibrated by a similar quantity, and also avoids any high-frequency noise in $h_{c'}[\mathbf{k}]$ which would be the analogy of the effects observed in Fig. 4. Just as before, we assume that the squeezing operation does not affect the HCF COM of cover images:

$$\mathcal{C}_3(H_c[\mathbf{k}]) \approx \mathcal{C}_3(H_{c'}[\mathbf{k}]). \tag{6}$$

Modelling cover images as a Markhov source it is possible to find sufficient conditions for this to be true, but we omit the details here. A scattergram is shown in Fig. 5, left; similar results are observed in other image sets, including the uncompressed bitmaps. The correlation appears weaker than between $\mathcal{C}_3(H_c[\mathbf{k}])$ and $\mathcal{C}_3(H_{\frac{1}{2}c}[\mathbf{k}])$ (it happens to be stronger for grayscale images), but nonetheless we will see that it forms a useful calibration.

We consider the effect, on the squeezed image, of LSB steganography on the parent image. Since each pixel of the squeezed image is the sum of two pixels in the cover, it undergoes the same noise as LSB matching summed twice. Therefore, $H_{s'}[\mathbf{k}] = H_{c'}[\mathbf{k}] \prod_{i=1}^{3} (\psi_p(k_i))^2$, where $\psi_p(k) = 1 - p\sin^2(\pi k/2N)$. Note that ψ_p differs from ϕ_p because the DFT involved is of a different length. Now we need the analogue of Lemma 4.1:

$$\mathcal{C}_3(H_s[\mathbf{k}]) < \mathcal{C}_3(H_{s'}[\mathbf{k}]). \tag{7}$$

The proof method of Lemma 4.1 does *not* work, because $\phi_p(k)/(\psi_p(k))^2$ is not always decreasing. Indeed, (7) is not guaranteed to hold for all signals. Nonetheless, it does seem to hold for images, and indeed it is possible to give very relaxed conditions for (7) to be true, although the details are messy and we omit them here.

The calibrated detector is $\mathcal{C}_3(H[\mathbf{k}])/\mathcal{C}_3(H'[\mathbf{k}])$. Despite the weaker correlation seen in Fig. 5, and the wooliness of (7), in practice this method of calibration is superior. It does not suffer from the same problem as the detector calibrated by resampling; in the case of the image displayed in Fig. 4, the "squeezed" stego image has COM $(60.9, 60.6, 59.8)$ so the inequality (7) does hold. In Fig. 5, right, we demonstrate that it also provides an improvement over the standard HCF COM detector on uncompressed covers, although the performance is only moderate in this, difficult, case (note the scale of the x-axis).

A major drawback of this technique, however, is that we are required to perform a DFT on $(2N)^3$ points. While not completely infeasible, it is slow and we encountered difficulty fitting the resulting 512^3 complex floats into the 512MB of memory in the nodes of our distributed steganalysis cluster. The space problem can be overcome, with some minor modifications to the standard FFT algorithm, but we do not report the details here.

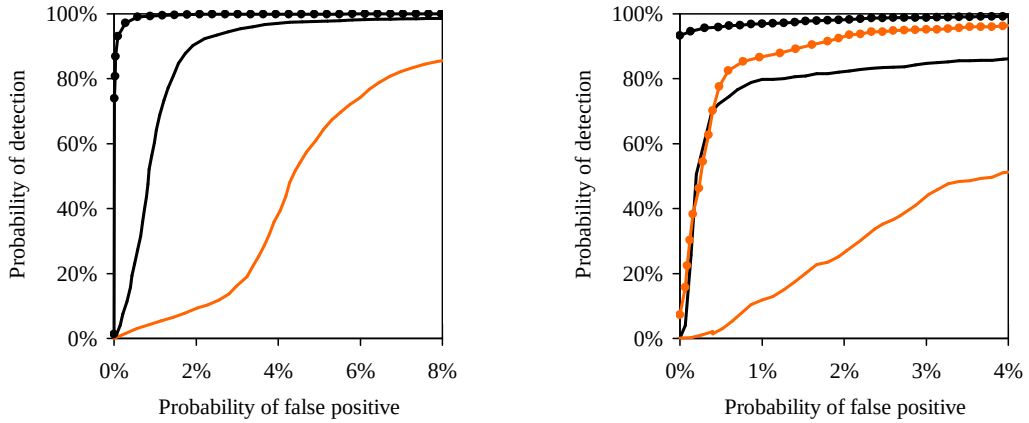

Figure 6. Left, ROCs demonstrating the sensitivity of the final detector. Over a set of 10000 JPEG cover images, steganography is performed with messages of length 20% (*marked line*), 10% (*black line*) and 5% (*light line*). The detector maintains good reliability even for these short messages. Right, demonstrating that this detector maintains superior performance when the same JPEGs are resampled down to 70% of their original size prior to embedding. The standard HCF COM detector (*light lines*) and the final detector (*dark lines*) are compared when the hidden message is 50% (*marked lines*) and 20% (*unmarked lines*) of the maximum.

4.3. Totalling Colour Components

Aiming to reduce the time and space complexity of the preceding method, and having already considered signals with a greater range than $0, \ldots, N-1$, we examined the effect of adding up the three colour channels in the image to make a 1-dimensional signal. Let us call this the "totalled" signal, writing $p_{c^t}(i,j)$ for the total of red, green, and blue intensities at point (i,j) of the cover, and h_{c^t}, and H_{c^t} for the corresponding histogram and HCF (obtained using a $3N$-point DFT).

This makes sense, because under LSB Matching each component receives the same, independent, level of noise; each pixel in this "totalled" signal receives the same level of noise three times over, independently. Therefore, after a message proportion p of the maximum is embedded, $H_{s^t}[k] = H_{c^t}[k]\chi_p(k)^3$ where $\chi_p(k) = 1 - p\sin^2(\pi k/3N)$.

The most obvious advantage is that the DFT involved is only 1-dimensional (of size $3N$), which quick to compute. But there is another, substantial, advantage: now we can apply the method of the *adjacency histogram* we used for grayscale image, described earlier in Sect. 4, which was previously denied to us for reasons of space. By taking the joint distribution of adjacent values in the totalled signal, we have to perform a 2-dimensional DFT but gain substantially on performance. Finally, we can squeeze down the totalled image to apply the calibration technique of Sect. 4.2 as well. This calibrated classifier, the final one we present here, is therefore

$$\left(\frac{\sum_{i_1,i_2=0}^{n}(i_1+i_2)|AH_t[i_1,i_2]|}{\sum_{i_1,i_2=0}^{n}|AH_t[i_1,i_2]|}\right)\Bigg/\left(\frac{\sum_{i_1,i_2=0}^{n}(i_1+i_2)|AH'_t[i_1,i_2]|}{\sum_{i_1,i_2=0}^{n}|AH'_t[i_1,i_2]|}\right)$$

where $n = 3N/2$, AH_t is the adjacency HCF of the "totalled" signal (formed using a $(3N)^2$-point DFT) and AH'_t is the adjacency HCF of the "totalled", "squeezed" signal ((formed using a $(6N)^2$-point DFT). As it is already a 1-dimensional classifier, no training is required.

This detector is, in most cases, a large step up in sensitivity from the others discussed here. Figure 6 shows the extent of this, in the case of one set of JPEG images, even after the JPEGs are resampled. We can detect steganography with reasonable reliability even when the hidden message is only 5% of the maximum. As more evidence, we performed the same experiment as reported in Tab. 2, embedding a 50% length message in the set of 3000 bitmaps, which had been subject to JPEG compression and then reduced in size before embedding. When the false negative rates are 50%, this detector has false positive rates of 0.0%, 0.2% and 2.2% for resampling ratios of 70%, 50%, and 30%, respectively. This compares very favourably with the entries in Tab. 2. (On the other hand, this detector is not so good with bitmap images which have never been JPEGs: the calibrated detector of the previous section is superior in this case. Understanding why will be the subject of future research.)

5. CONCLUSIONS

Spatial-domain LSB Matching is a very simple steganography technique, simple enough to perform with command line utilities, but generally harder to detect than the more widely studied LSB Replacement method. We have investigated ways to detect LSB Matching steganography, and developed new detectors. We combine a number of different ideas: summing the colour channels of a colour image, using the technology of the HCF COM developed by Harmsen, modifying the method to use an adjacency histogram (or co-occurrence matrix) instead of the usual histogram, and calibrating the resulting COM by summing adjacent pixels in the image.

The result is a much more sensitive detector, especially for steganography in images which have been JPEG compressed, including when they have been resampled prior to embedding. Nonetheless, this paper has introduced the idea of "calibration" in the spatial domain in a very basic way, and we expect that more sophisticated methods (perhaps based on suitable denoising algorithms) will supercede the detectors given here.

ACKNOWLEDGMENTS

The author is a Royal Society University Research Fellow.

REFERENCES

1. J. Fridrich, M. Goljan, and R. Du, "Reliable detection of LSB steganography in color and grayscale images," *Proc. ACM Workshop on Multimedia and Security*, pp. 27–30, 2001.
2. S. Dumitrescu, X. Wu, and Z. Wang, "Detection of LSB steganography via sample pair analysis," in *Proc. Information Hiding Workshop*, *Springer LNCS* **2578**, pp. 355–372, 2002.
3. A. Ker, "Improved detection of LSB steganography in grayscale images," in *Proc. Information Hiding Workshop*, *Springer LNCS* **3200**, pp. 97–115, 2004.
4. A. Westfeld, "Detecting low embedding rates," in *Proc. Information Hiding Workshop*, *Springer LNCS* **2578**, pp. 324–339, 2002.
5. J. Harmsen and W. Pearlman, "Higher-order statistical steganalysis of palette images," in *Security and Watermarking of Multimedia Contents V*, E. J. Delp III and P. W. Wong, eds., *Proc. SPIE* **5020**, pp. 131–142, 2003.
6. J. Harmsen, K. Bowers, and W. Pearlman, "Fast additive noise steganalysis," in *Security, Steganography, and Watermarking of Multimedia Contents VI*, E. J. Delp III and P. W. Wong, eds., *Proc. SPIE* **5306**, pp. 489–495, 2004.
7. A. Ker, "Steganalysis of LSB matching in grayscale images." To appear in *IEEE Signal Processing Letters*, 2005.
8. A. Ker, "Quantitive evaluation of pairs and RS steganalysis," in *Security, Steganography, and Watermarking of Multimedia Contents VI*, E. J. Delp III and P. W. Wong, eds., *Proc. SPIE* **5306**, pp. 83–97, 2004.
9. T. Sharp, "An implementation of key-based digital signal steganography," in *Proc. Information Hiding Workshop*, *Springer LNCS* **2137**, pp. 13–26, 2001.
10. Outguess Steganography Software. `http://www.outguess.org/`.
11. J. Fridrich, M. Goljan, and R. Du, "Steganalysis based on JPEG compatability," in *Multimedia Systems and Applications IV*, A. G. Tescher, B. Vasudev, and V. M. Bove, Jr, eds., *Proc. SPIE* **4518**, pp. 275–280, 2002.
12. Independent JPEG Group. `http://www.ijg.org/`.
13. S. Lyu and H. Farid, "Steganalysis using colour wavelet statistics and one-class vector support machines," in *Security, Steganography, and Watermarking of Multimedia Contents VI*, E. J. Delp III and P. W. Wong, eds., *Proc. SPIE* **5306**, pp. 35–45, 2004.
14. NRCS Photo Gallery. `http://photogallery.nrcs.usda.gov/`.
15. J. Fridrich, R. Du, and L. Meng, "Steganalysis of LSB encoding in color images," in *Proc. IEEE International Conference on Multimedia and Expo*, pp. 1279–1282, 2000.
16. R. O. Duda, P. E. Hart, and H. G. Stork, *Pattern Classification*, Wiley-Interscience, New York, 2nd ed., 2000.