Information Hiding and Covert Communication

Andrew Ker

adk@comlab.ox.ac.uk

Royal Society University Research Fellow
Oxford University Computing Laboratory

Foundations of Security Analysis and Design

Bertinoro, Italy, 25-26 August 2008

Information Hiding

and Covert Communication

Steganography

στεγανος + γραφειν
“covered” + “to write”

Digital Watermarking

(fingerprinting, tamperproofing)

Digital Media
Forensics

Information Hiding and Covert Communication

Part 1: Introduction to Steganography & Watermarking

Highlight: examples

Part 2: Steganalysis

Highlight: extremely sensitive detectors for covert communication

Part 3: More Efficient Steganography

Highlight: codes which approach the maximum possible efficiency

Part 4: Steganographic Capacity

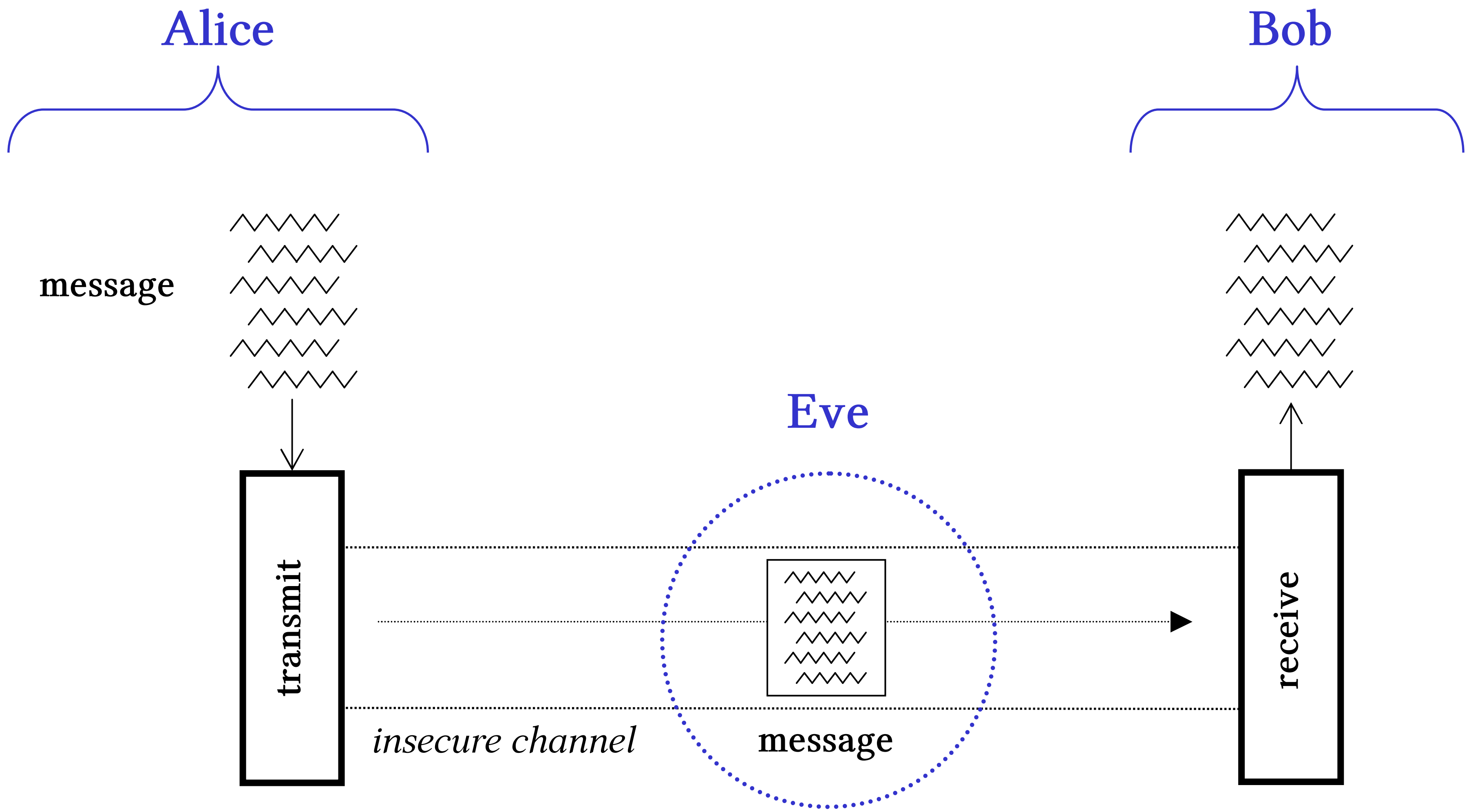
Highlight: the Square Root Law

Information Hiding and Covert Communication

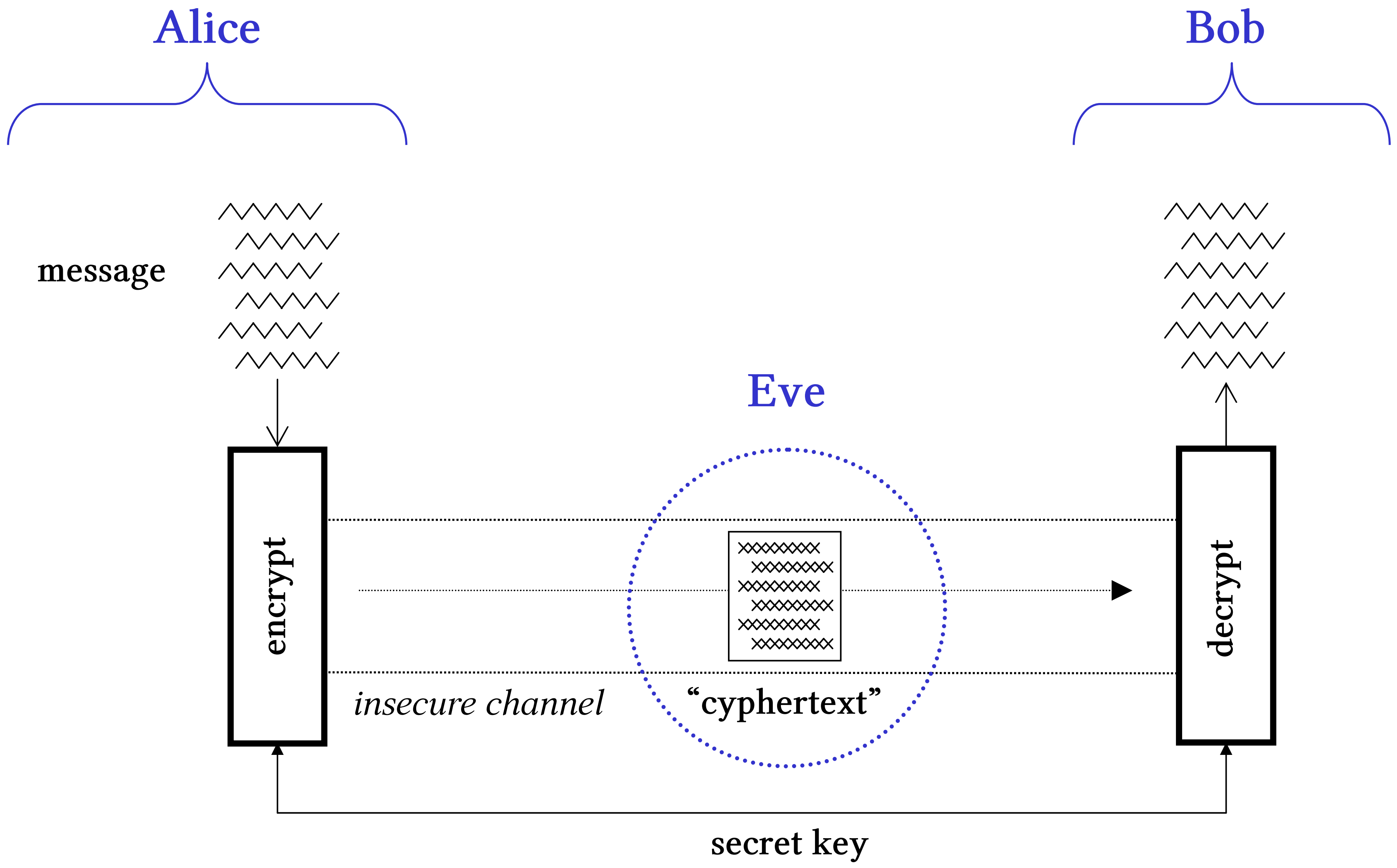
Part 1: Introduction to Steganography & Watermarking

- Steganography
  - Examples & countermeasures (steganalysis)
- Digital watermarking
  - Applications
  - Examples & countermeasures
- Typical embedding domains & operations

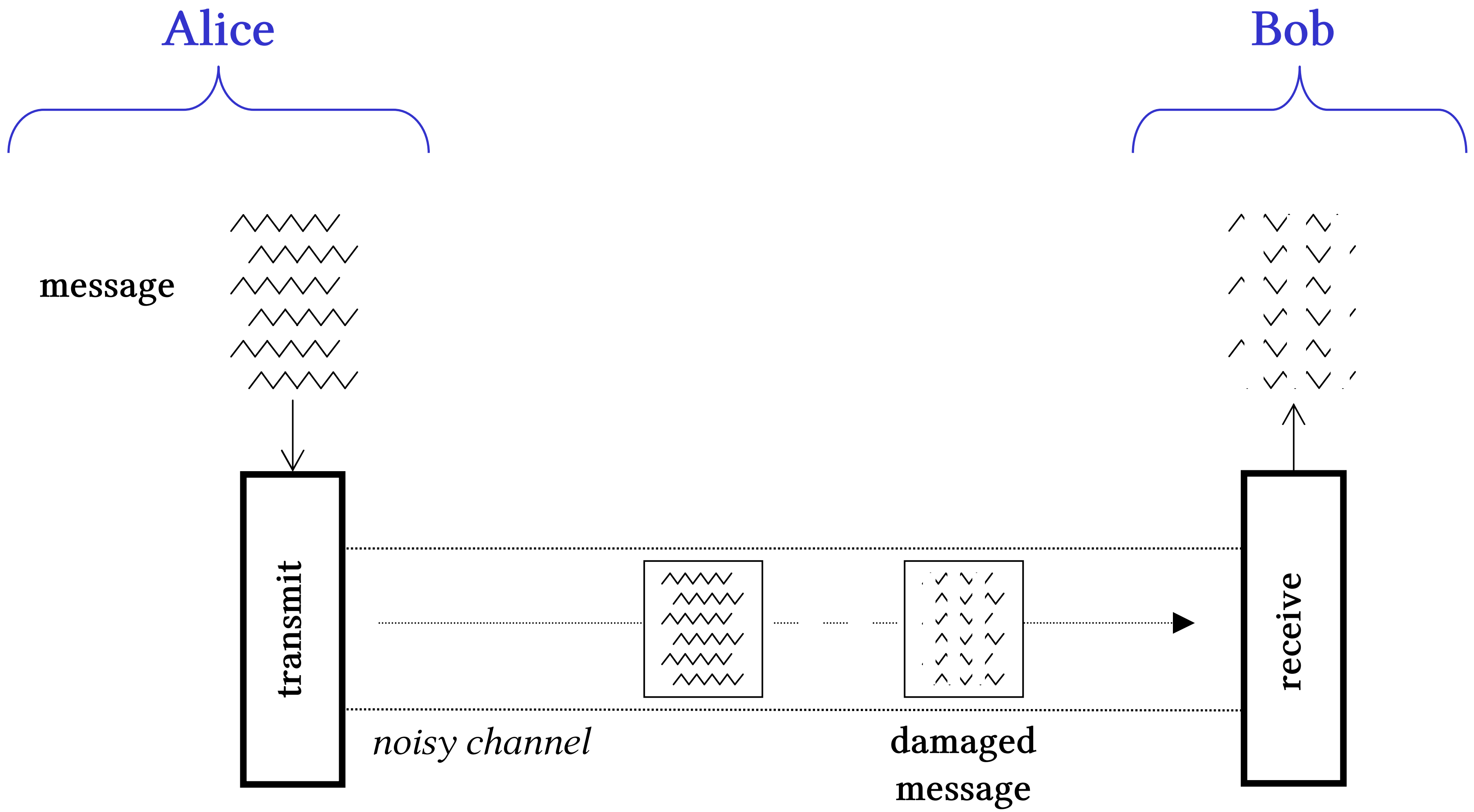
Communication



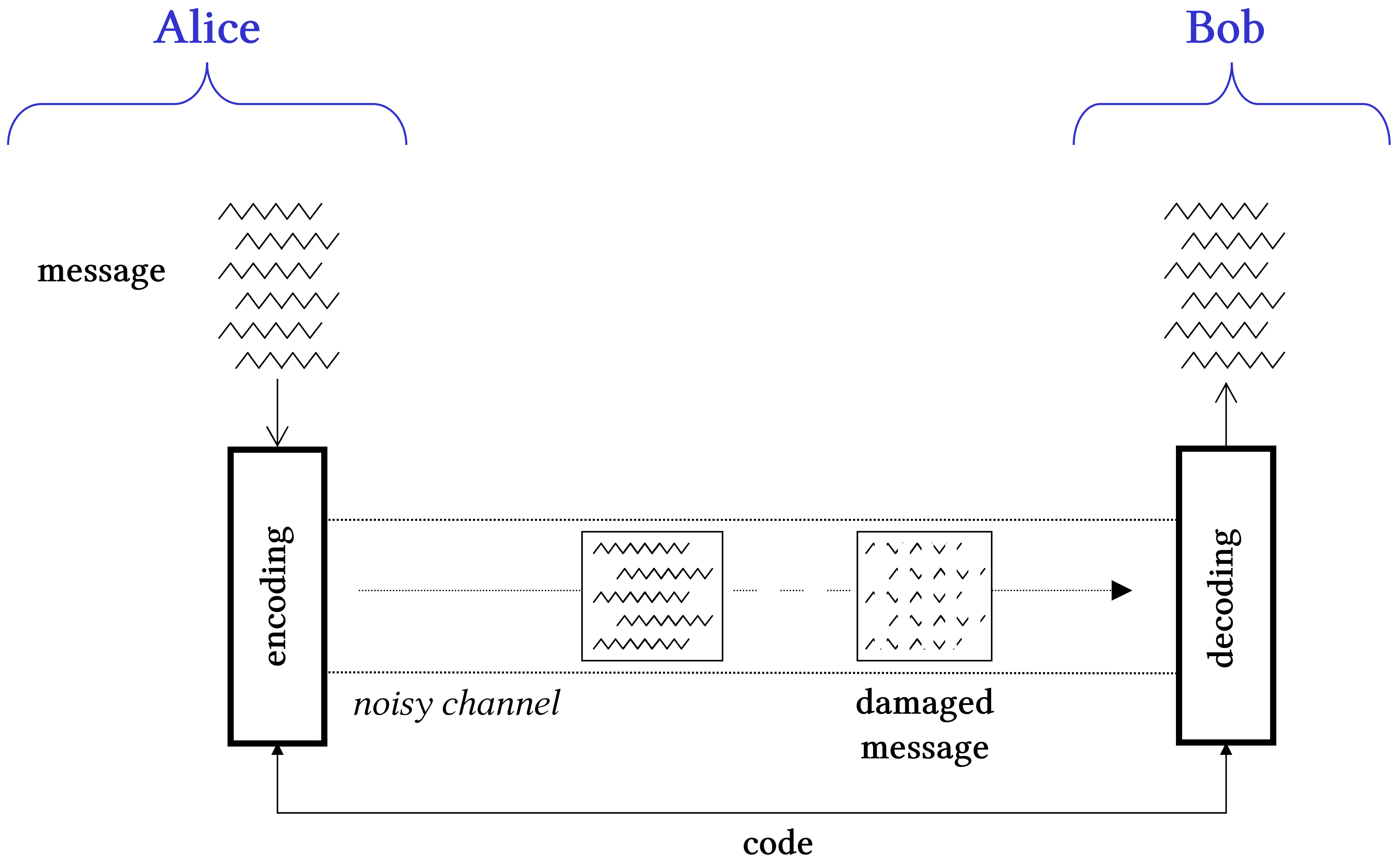
Communication: cryptography



Communication



Communication: coding



Communication

1. Cryptography

   Keep the message confidential

2. Coding

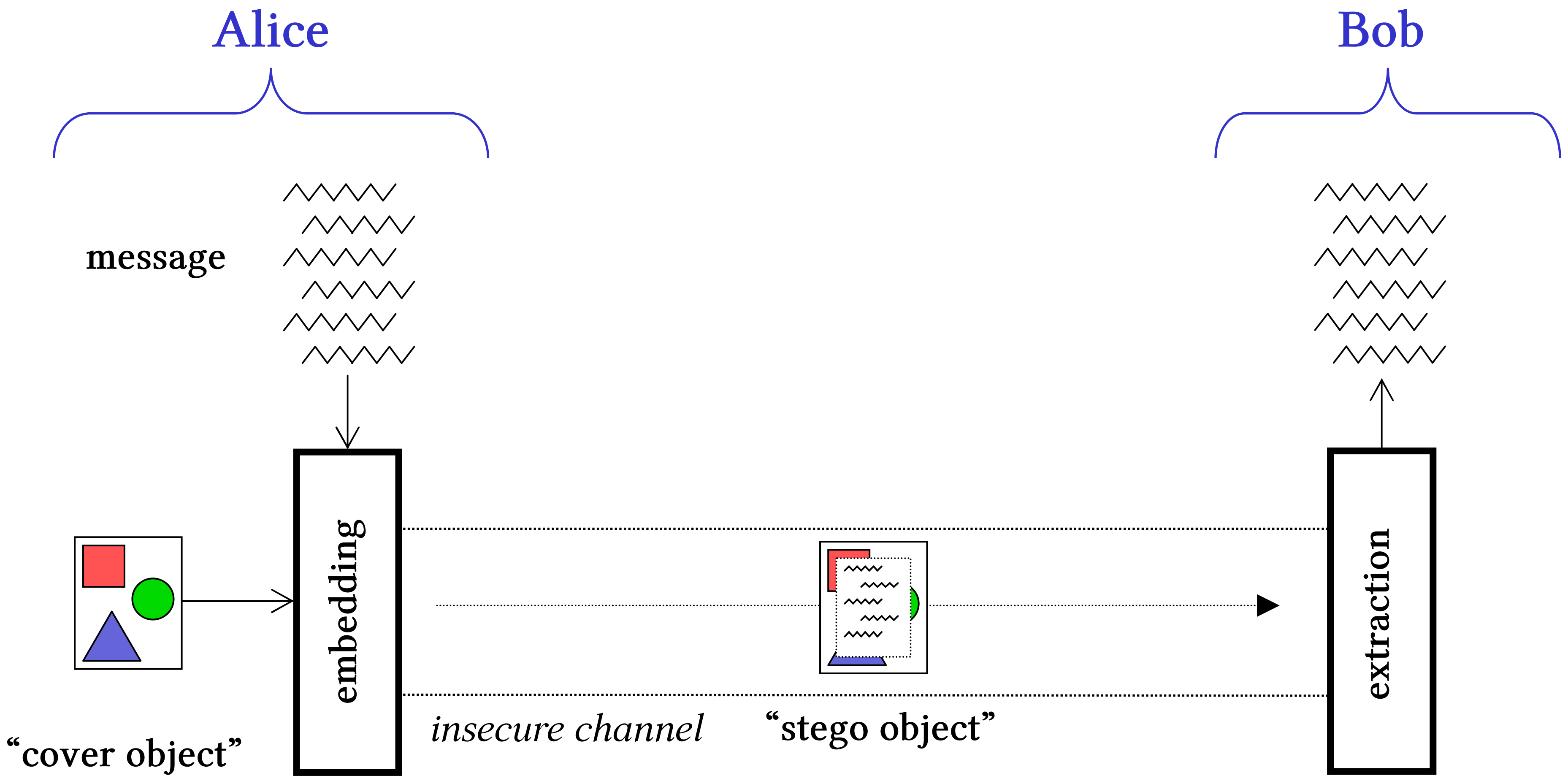
   Keep the message intact

Other challenges:

How can Alice and Bob share the secret key/code?

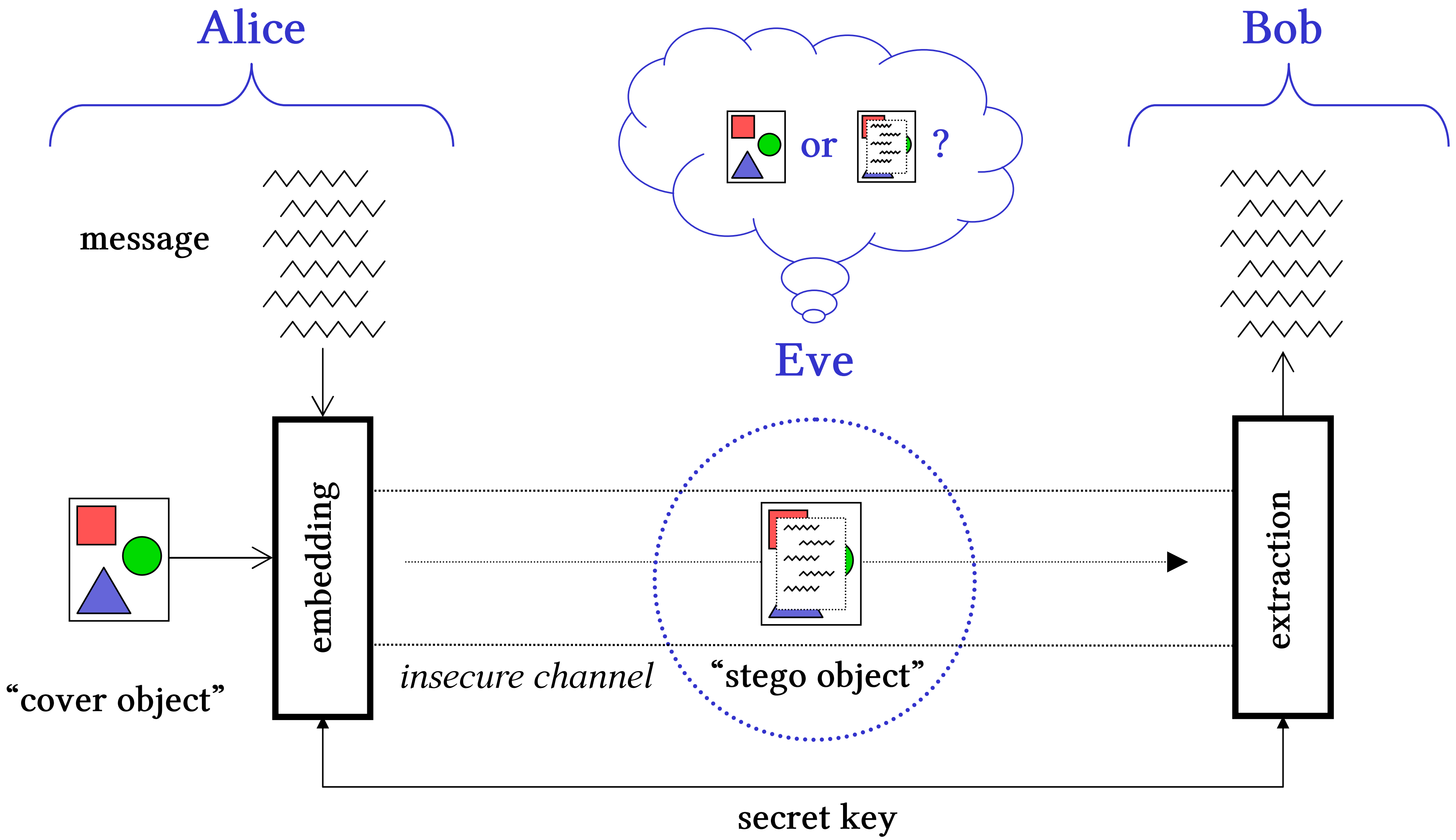
How does Alice know that she is communicating with Bob?

Steganography



G. Simmons. *The Prisoners' Problem and the Subliminal Channel.* In Proc. Crypto '83, Plenum Press, 1984.

Steganography

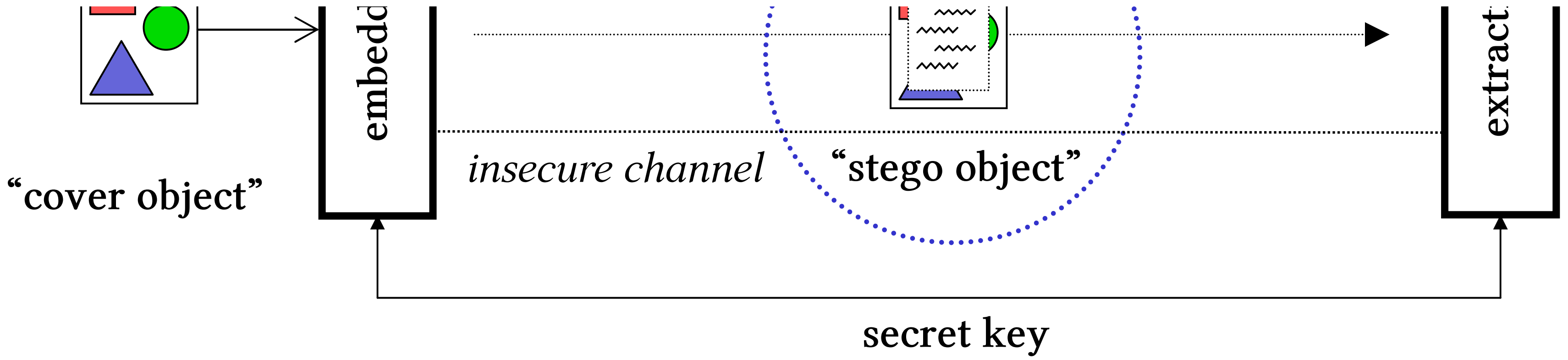


Steganography

Why digital media?

- *We will see that it has plenty of capacity for hidden data.*
- *Over 70% of all internet traffic is audio, picture, or video.*
- *Over 20% of all internet traffic is YouTube video!*
- *Over 90% of peer-to-peer traffic is audio, picture, or video.*

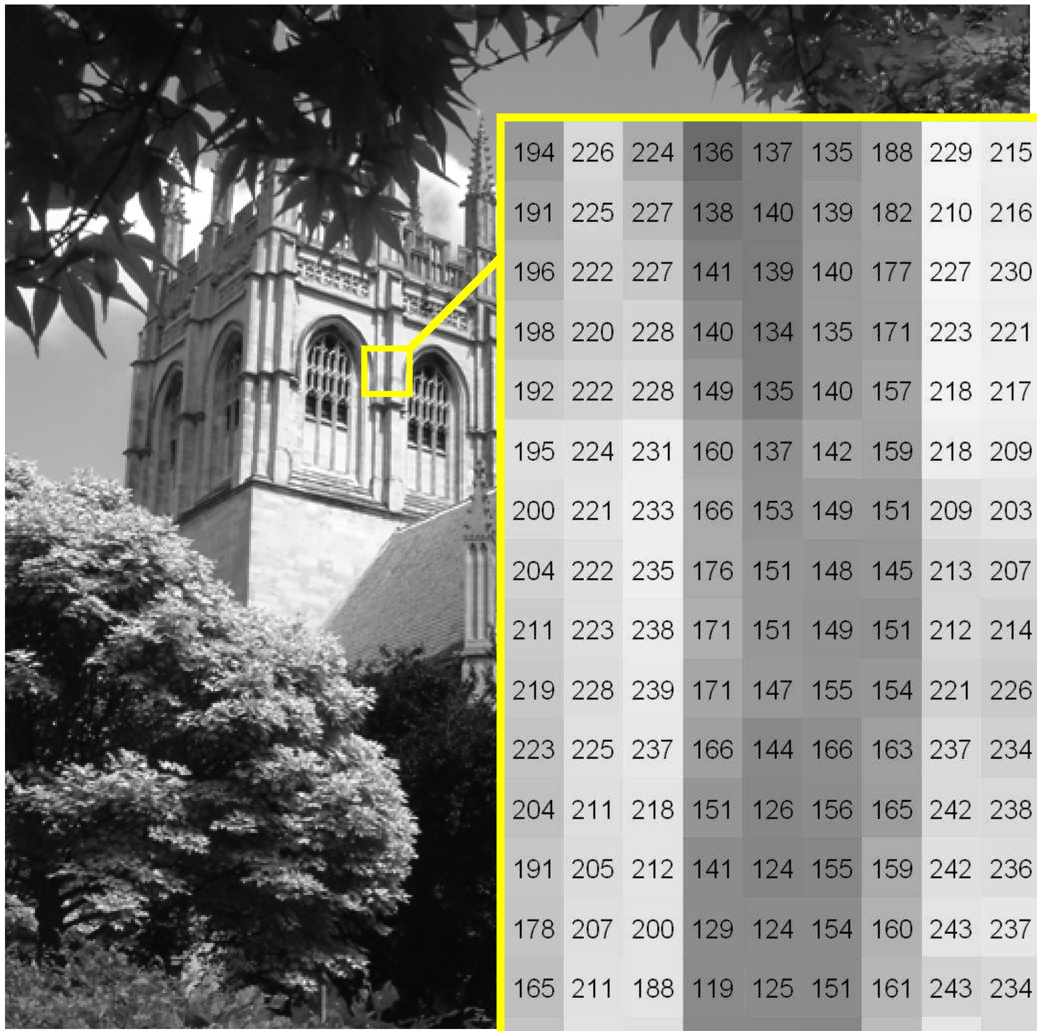
[Ipoque / Ellacoya Networks, 2007]

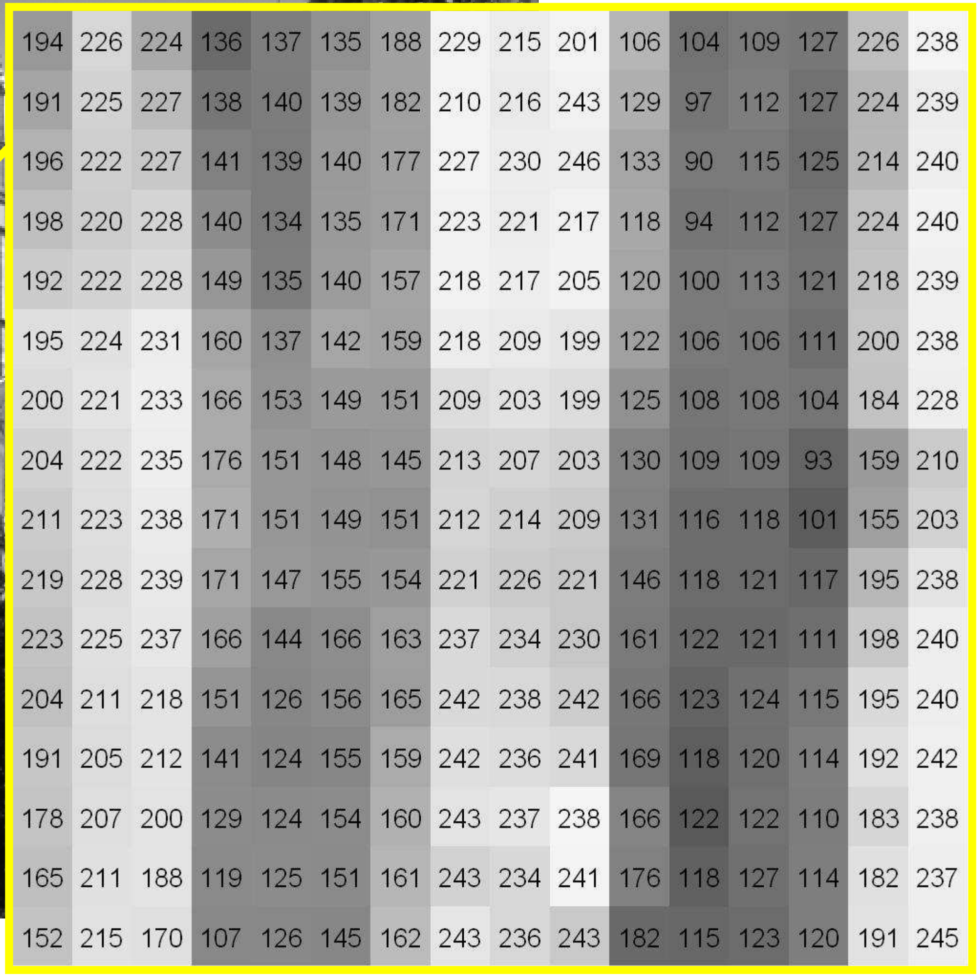


Example

Cover object, 512×512 pixels, grayscale

Cover object, 512×512 pixels, grayscale





194	226	224	136	137	135	188	229	215	201	106	104	109	127	226	238
191	225	227	138	140	139	182	210	216	243	129	97	112	127	224	239
196	222	227	141	139	140	177	227	230	246	133	90	115	125	214	240
198	220	228	140	134	135	171	223	221	217	118	94	112	127	224	240
192	222	228	149	135	140	157	218	217	205	120	100	113	121	218	239
195	224	231	160	137	142	159	218	209	199	122	106	106	111	200	238
200	221	233	166	153	149	151	209	203	199	125	108	108	104	184	228
204	222	235	176	151	148	145	213	207	203	130	109	109	93	159	210
211	223	238	171	151	149	151	212	214	209	131	116	118	101	155	203
219	228	239	171	147	155	154	221	226	221	146	118	121	117	195	238
223	225	237	166	144	166	163	237	234	230	161	122	121	111	198	240
204	211	218	151	126	156	165	242	238	242	166	123	124	115	195	240
191	205	212	141	124	155	159	242	236	241	169	118	120	114	192	242
178	207	200	129	124	154	160	243	237	238	166	122	122	110	183	238
165	211	188	119	125	151	161	243	234	241	176	118	127	114	182	237
152	215	170	107	126	145	162	243	236	243	182	115	123	120	191	245

LSB replacement

The simplest steganographic method for uncompressed ("TIF", "BMP") images.

Embedding

- Form payload as a sequence of bits,
- Take cover as a sequence of bytes,
- Replace Least Significant Bits (LSB) of cover bytes with payload.

Extraction

- Take cover as a sequence of bytes,
- Read LSBs.

Can be performed by an 80-character Perl program on a Unix commandline:

```
perl -n0777e '$_=unpack"b*",$_;split/(\s+)/,<STDIN>,5;
              @_[8]=~s{.}{$&&v254|chop()&v1}ge;print@_'
              <input.pgm >output.pgm stegotext
```

LSB replacement

The simplest steganographic method for uncompressed (“TIF”, “BMP”) images.

<u>Embedding</u>

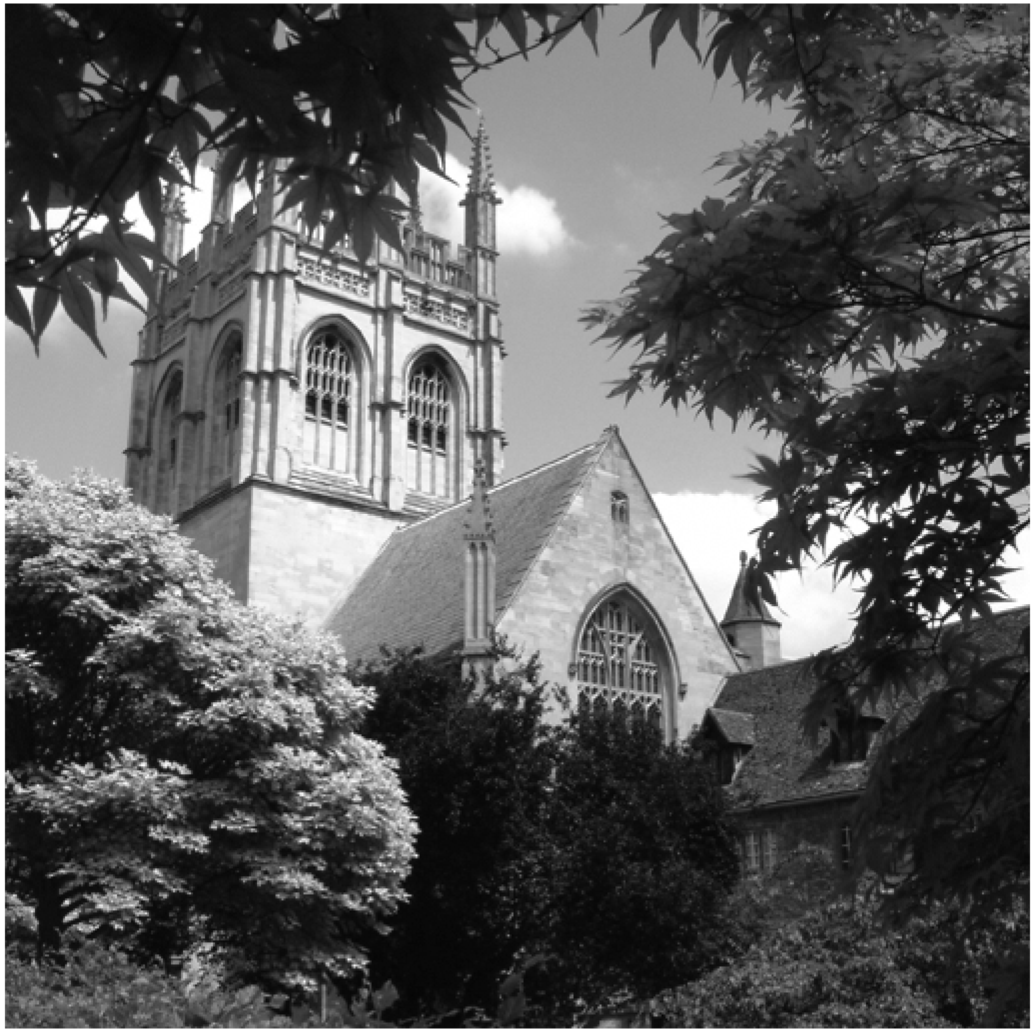
- Form **compressed & encrypted** payload as a sequence of bits,
- Take cover as a sequence of bytes **in pseudorandom order per secret key,**
- Replace Least Significant Bits (LSB) of cover bytes with payload.

<u>Extraction</u>

- Take cover as a sequence of bytes **in pseudorandom order per secret key,**
- Read LSBs, **decrypt and decompress**.

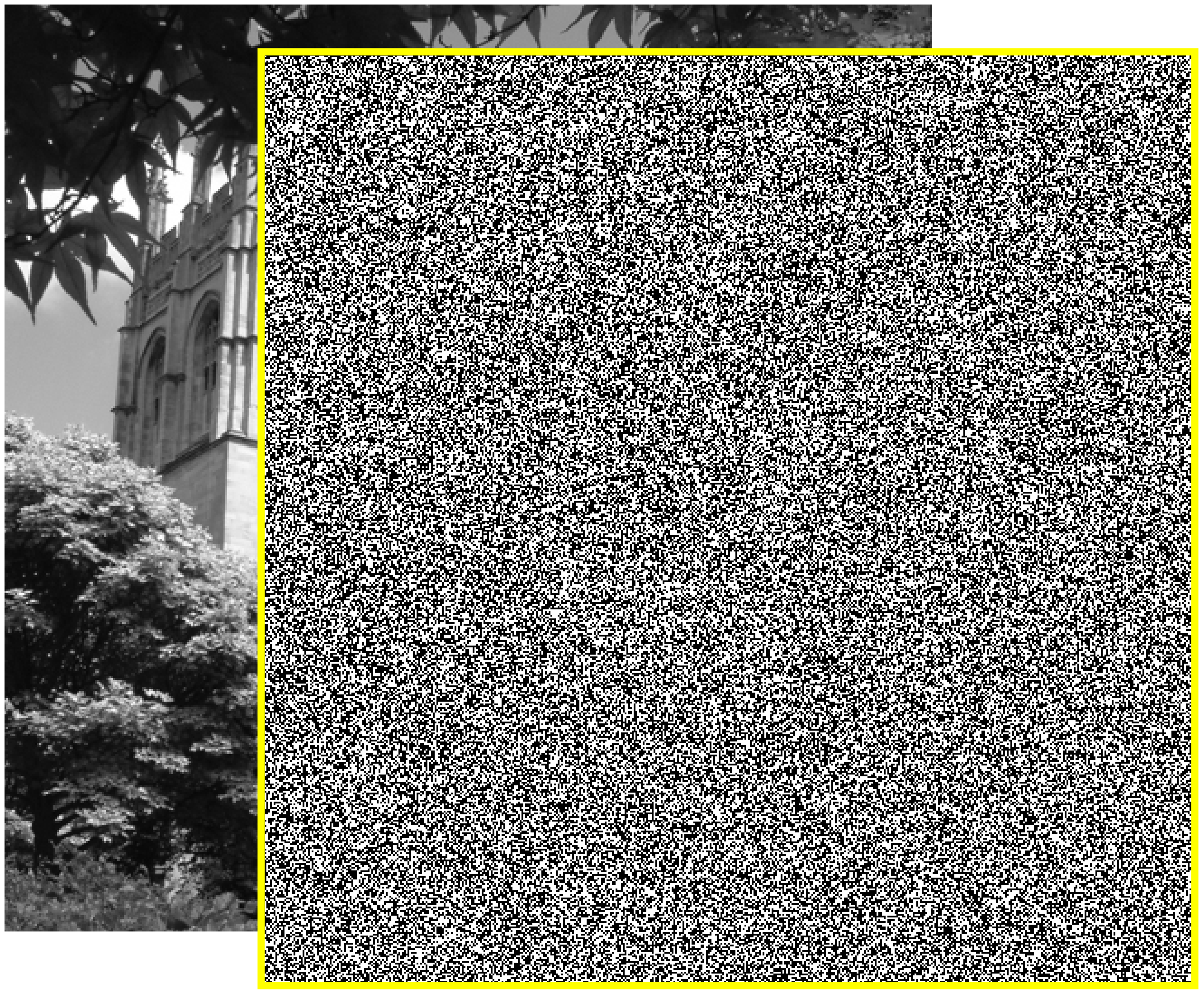
Using pseudorandom order also has the advantage of spreading smaller-than-maximal payloads throughout the cover.

Stego object, 1 secret bit per cover pixel (32KB)



Cover object, 512×512 pixels, grayscale

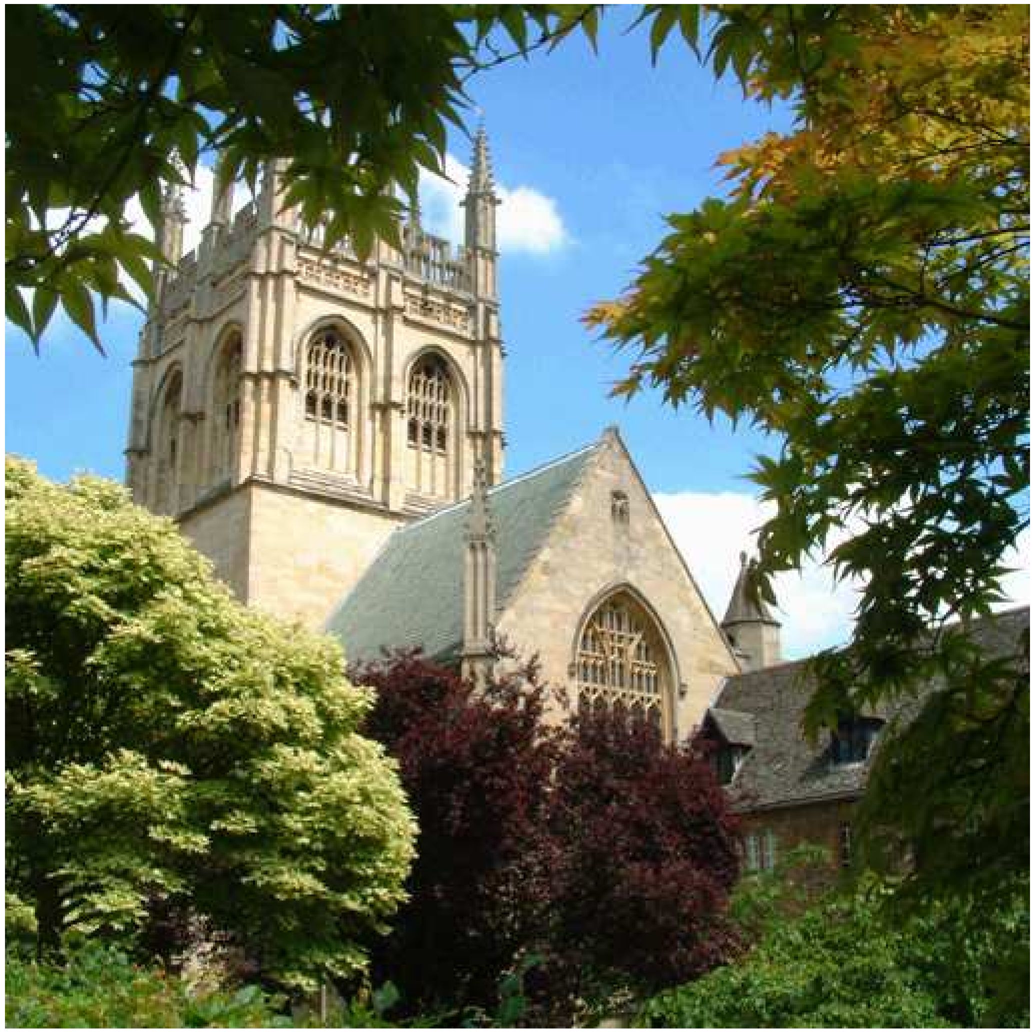
Stego object, 1 secret bit per cover pixel (32KB)



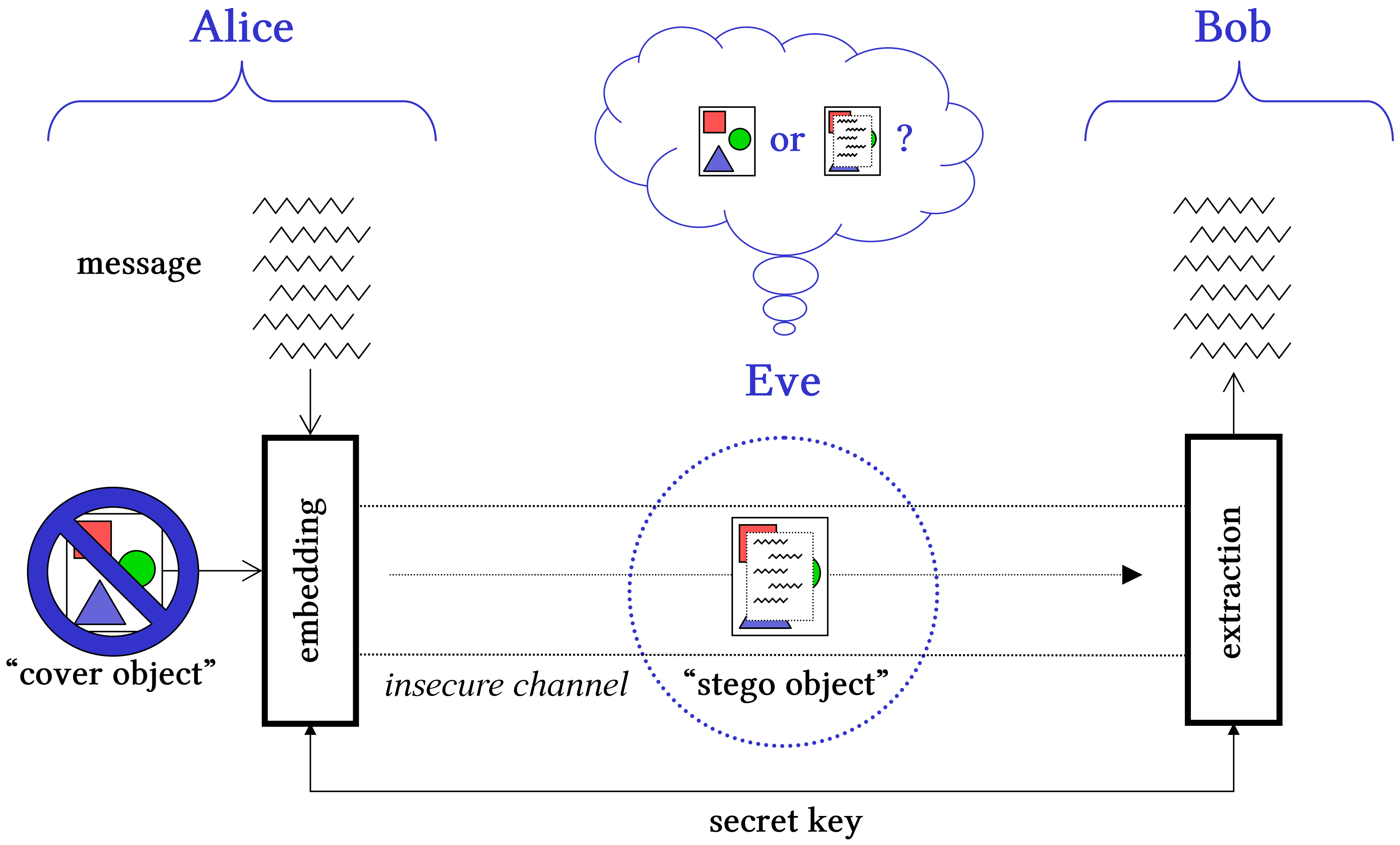
Cover object, 512×512 pixels, grayscale

Cover object, 512£512 pixels, 24 bits per pixel

Stego object, 9 secret bits per cover pixel (288KB)



Steganography



Steganography is not…

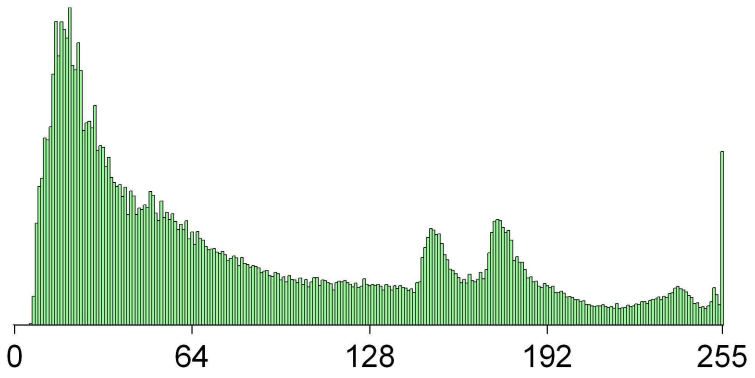
- Hiding data in pseudorandom streams e.g. Truecrypt.
  (too easy)

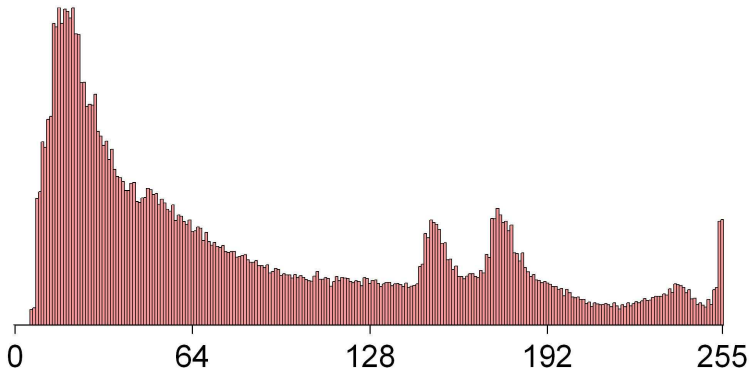
- Hiding data in unused parts of image/video/audio/packet/program headers.
  (not secure)

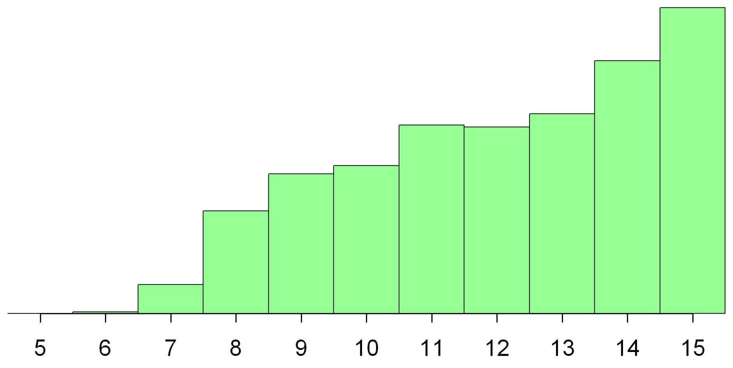
Steganalysis

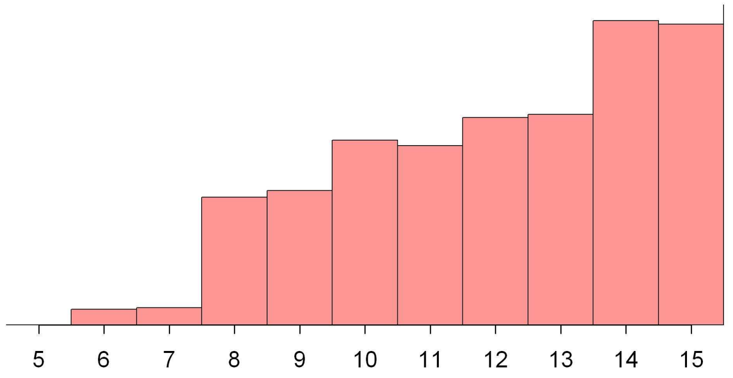
- the counter-discipline to steganography, detecting hidden data.

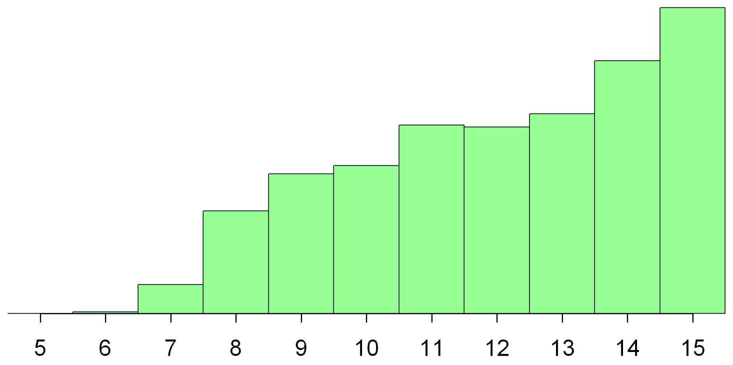
Even when steganography is not perceptible (visually, audibly, ...) it might still be detected by **statistical analysis**.

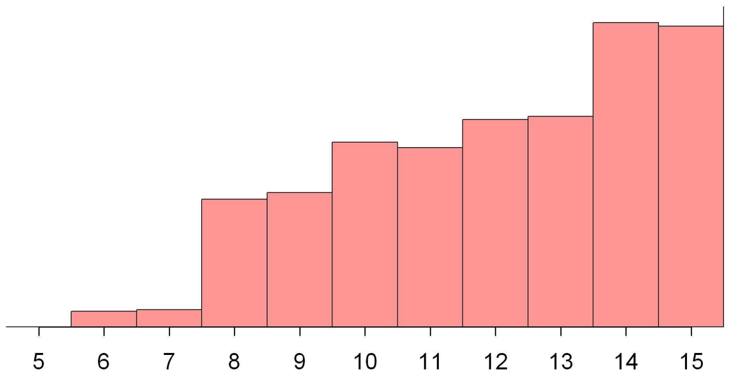

0
64
128
192
255


0
64
128
192
255


5
6
7
8
9
10
11
12
13
14
15


5
6
7
8
9
10
11
12
13
14
15

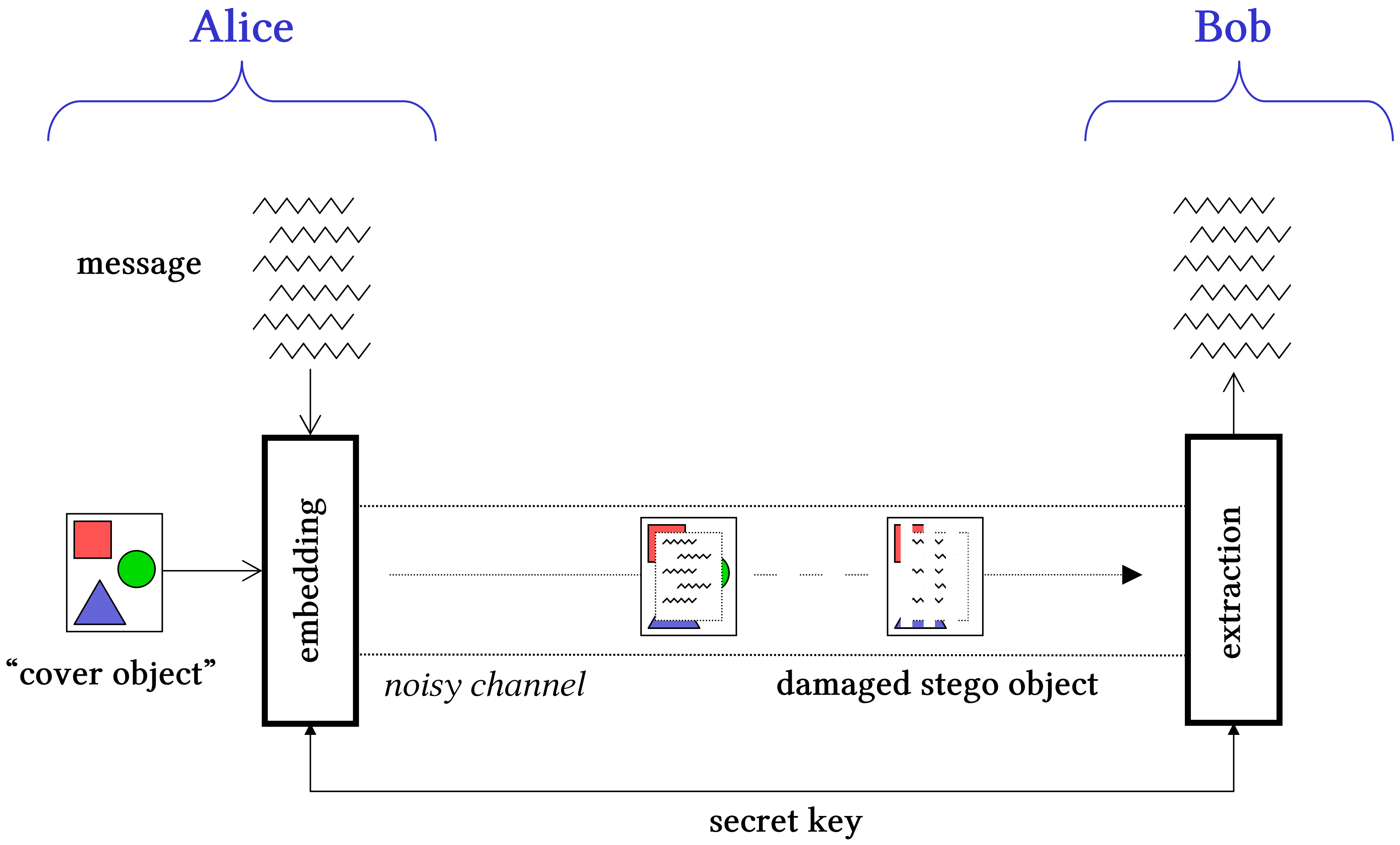




The “pairs of values” effect can be used to make a detector, known as the **“Chi-Square”** detector for LSB replacement

A. Westfeld & A. Pfitzmann. *Attacks on Steganographic Systems*. In Proc. 3rd Information Hiding Workshop, Springer LNCS, 1999.

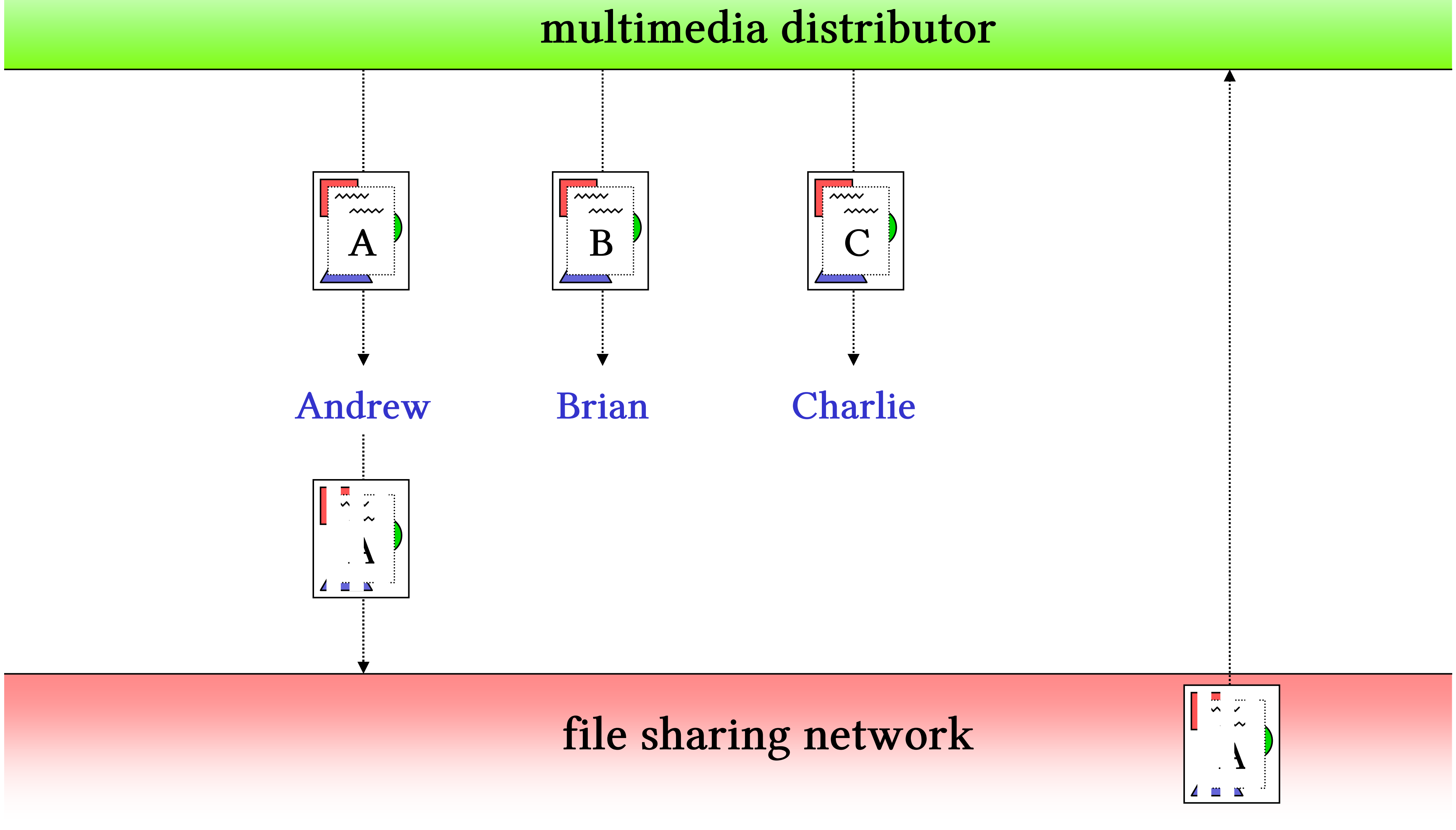
Digital watermarking



Applications

- Broadcast monitoring
  - *The watermark is a machine-readable tag identifying the cover content.*
- Copyright enforcement
  - *The watermark proves ownership of the cover.*
  - *The watermark indicates a license to play the cover medium on specific device.*
- Traitor tracing
  - *The watermark identifies the original recipient of the cover.*

Traitor tracing



Example "DSSS Watermarking"

Form cover as a vector of bytes:

$$\boldsymbol{d} = (d_1, \ldots, d_n)$$

Generate random "watermark" sequence (actually a secret key) of +1 and -1:

$$\boldsymbol{w} = (w_1, \ldots, w_n) \qquad w_i = \pm 1 \quad \text{equiprobably, and independently}$$

Watermarked image is simply $\boldsymbol{d}^* = \boldsymbol{d} + \alpha \boldsymbol{w}$ (α is the watermark strength).

To detect the presence of the watermark, form the "similarity" or "normalized correlation" score

$$nc(\boldsymbol{d}^*, \boldsymbol{w}) = \frac{\boldsymbol{d}^* \cdot \boldsymbol{w}}{\sqrt{\boldsymbol{d}^* \cdot \boldsymbol{d}^*}}.$$

which is high when this particular watermark is present and low otherwise.

I. Cox, J. Kilian, F. Leighton, & T. Shamoon. *Secure Spread Spectrum Watermarking for Multimedia.* IEEE Trans. Image Processing, 6(12), 1997.

Example “DSSS Watermarking”

Form cover as a vector of bytes:

$$\boldsymbol{d} = (d_1, \ldots, d_n)$$

Generate random “watermark” sequence (actually a secret key) of +1 and -1:

$$\boldsymbol{w} = (w_1, \ldots, w_n) \qquad w_i = \pm 1 \quad \text{equiprobably, and independently}$$

Watermarked image is simply $\boldsymbol{d}^* = \boldsymbol{d} + \alpha \boldsymbol{w}$ (α is the watermark strength).

To detect the presence of the watermark, form the “similarity” or “normalized correlation” score

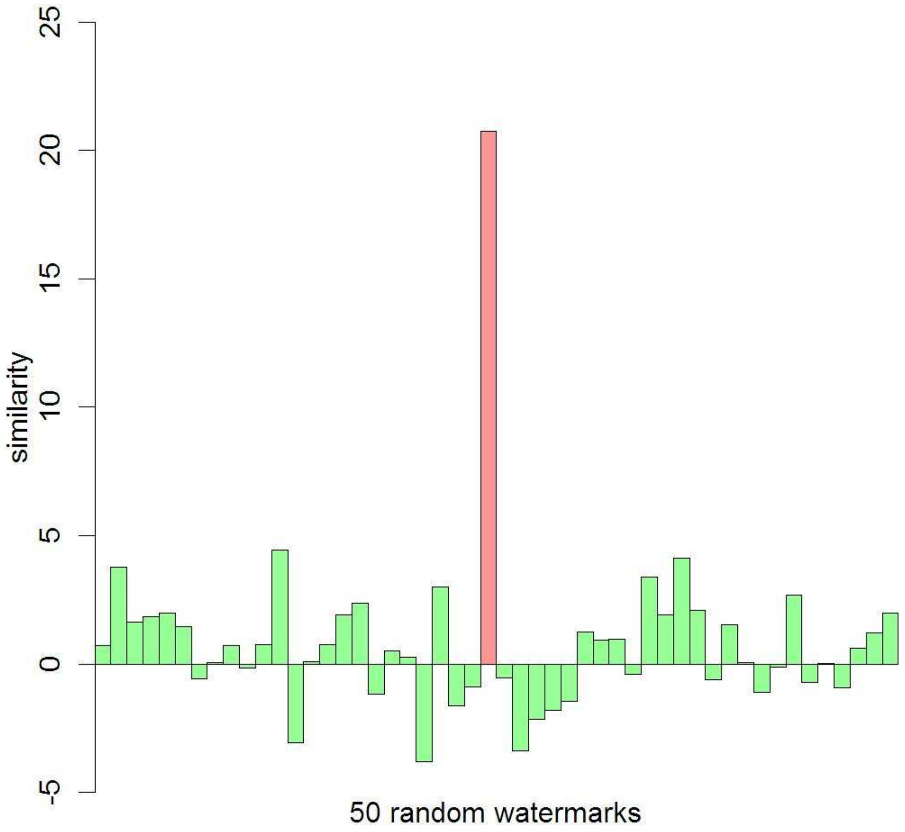
$$nc(\boldsymbol{d}^*, \boldsymbol{w}) = \frac{\boldsymbol{d}^* \cdot \boldsymbol{w}}{\sqrt{\boldsymbol{d}^* \cdot \boldsymbol{d}^*}}.$$

which is high when this particular watermark is present and low otherwise.

This watermark conveys no information except its presence (“zero bit watermark”).

Watermark detection

Watermarked image



Watermark attacks

- the counter-discipline to watermarking, destroying hidden data.

(The opponent wants to remove the watermark without destroying the cover.)

Note that embedding a watermark MUST degrade the cover a bit, otherwise it can be painlessly overwritten.

Common attacks

- Noise attack

  Not targeted: simply aim to reduce watermark fidelity.

- Collusion attack

  Take the average of many copies of the same cover, with different watermarks.

- Desynchronization attack

  Distort the cover spatially so that the watermark is no longer detectable.

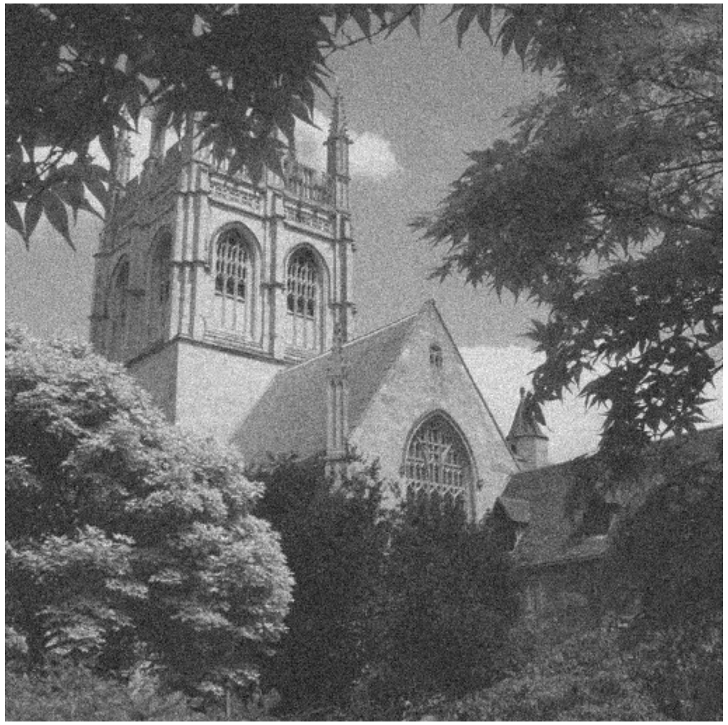
- Watermark estimation

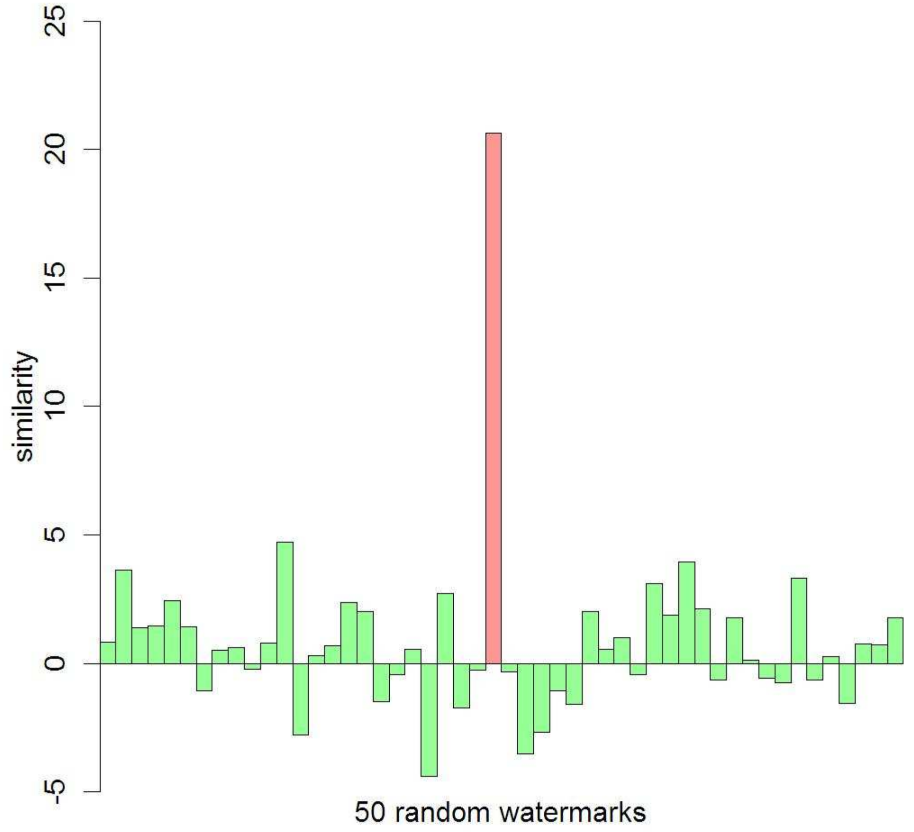
  EITHER use many objects with the same watermark, OR treat the detector as an oracle, to estimate (and then subtract) the watermark.

Noise attack

Added Gaussian noise (standard deviation=20).

Watermarked+attacked image

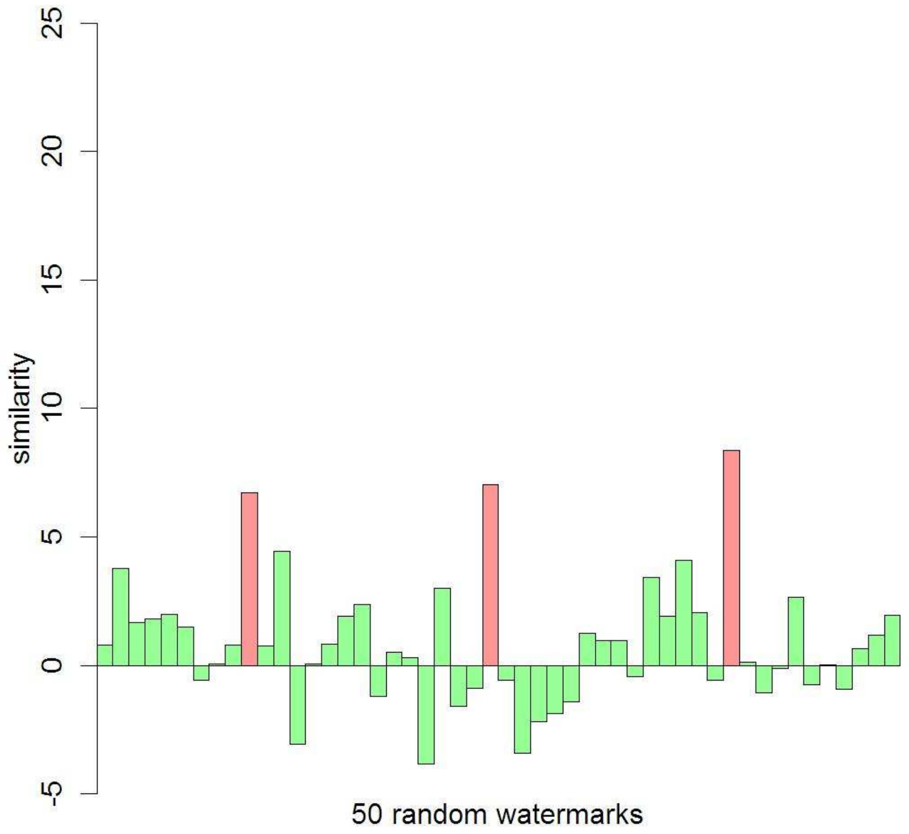




Collusion attack

Formed a new image by averaging three copies of the same cover marked with different watermarks.

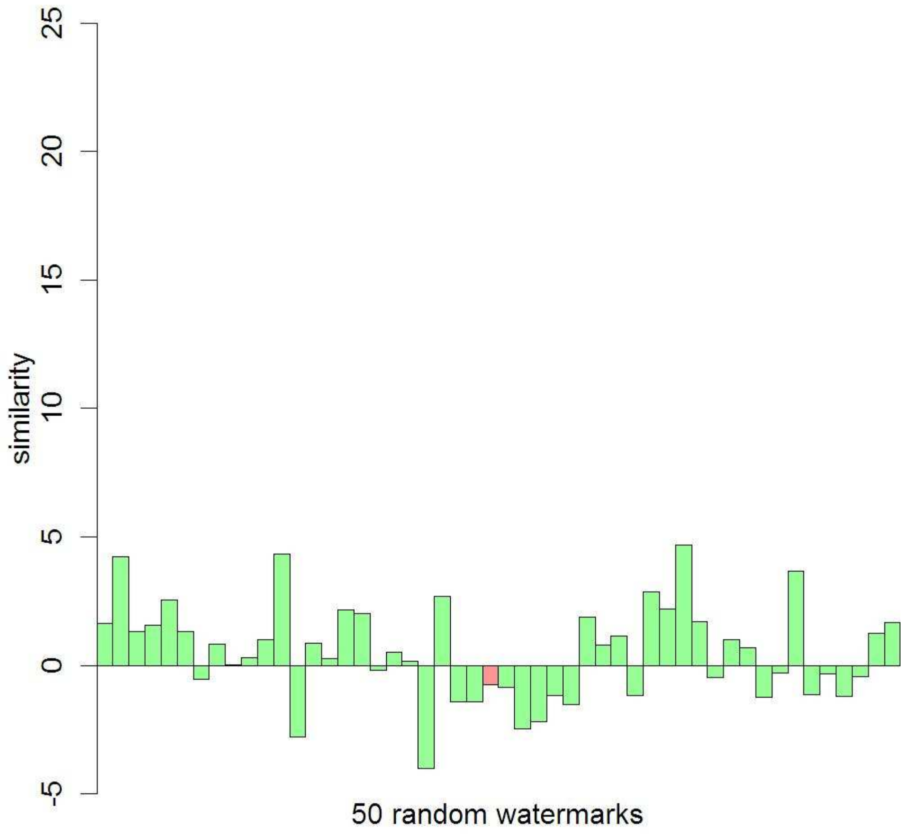
Watermarked+attacked image



Desynchronization attack

Removed pixel column 1 and pixel duplicated column 512.

Watermarked+attacked image



Steganography & watermarking

- Steganography

  Embed message so that it cannot be detected

  Steganalysis

  Detect hidden information

- Digital watermarking

  Embed message so that it cannot be removed

  Watermark attacks

  Remove information (without destroying cover)

JPEG

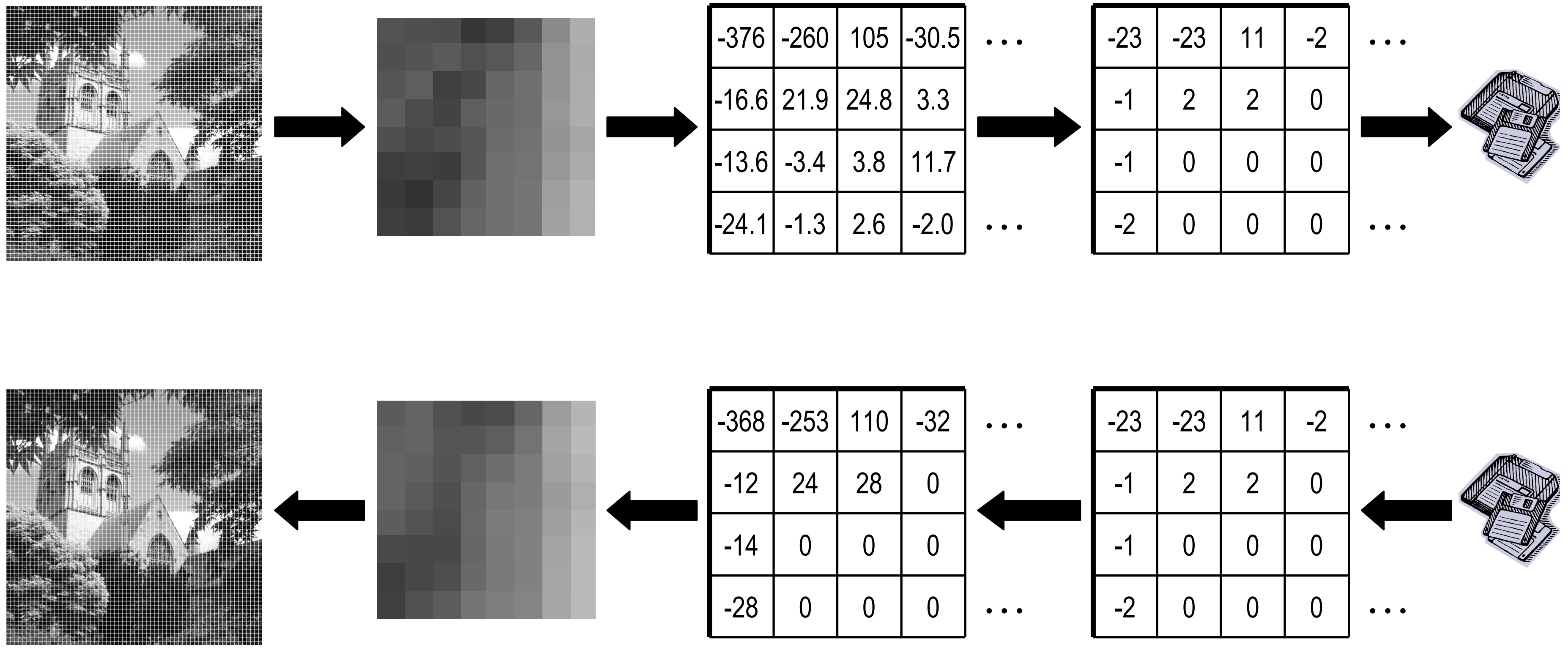
- lossy compression using quantization in the DCT domain.

1. Split into 8×8 blocks.
2. Discrete Cosine Transform.
3. Quantize DCT coefficients.
4. Compress conventionally and store.

JPEG

16	11	10	16	⋯
12	12	14	19	
14	13	16	24	
14	17	22	29	⋯

1. Split into 8×8 blocks.
2. Discrete Cosine Transform (to 8×8 coefficients).
3. Divide coefficients by “quantization table” and round to nearest.
4. Compress quantized coefficients conventionally and store.



-376	-260	105	-30.5	⋯
-16.6	21.9	24.8	3.3	
-13.6	-3.4	3.8	11.7	
-24.1	-1.3	2.6	-2.0	⋯

-23	-23	11	-2	⋯
-1	2	2	0	
-1	0	0	0	
-2	0	0	0	⋯

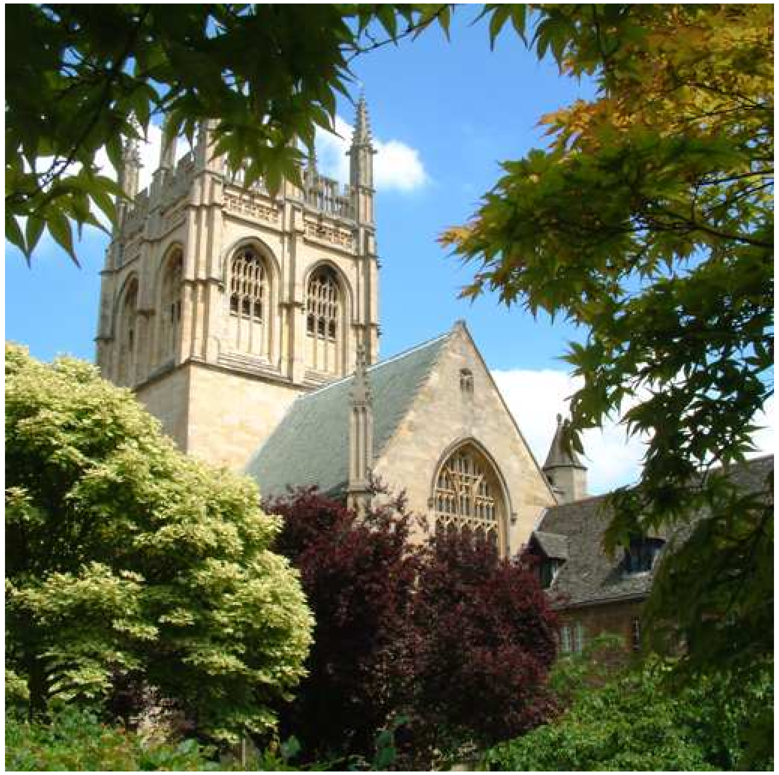
-368	-253	110	-32	⋯
-12	24	28	0	
-14	0	0	0	
-28	0	0	0	⋯

-23	-23	11	-2	⋯
-1	2	2	0	
-1	0	0	0	
-2	0	0	0	⋯

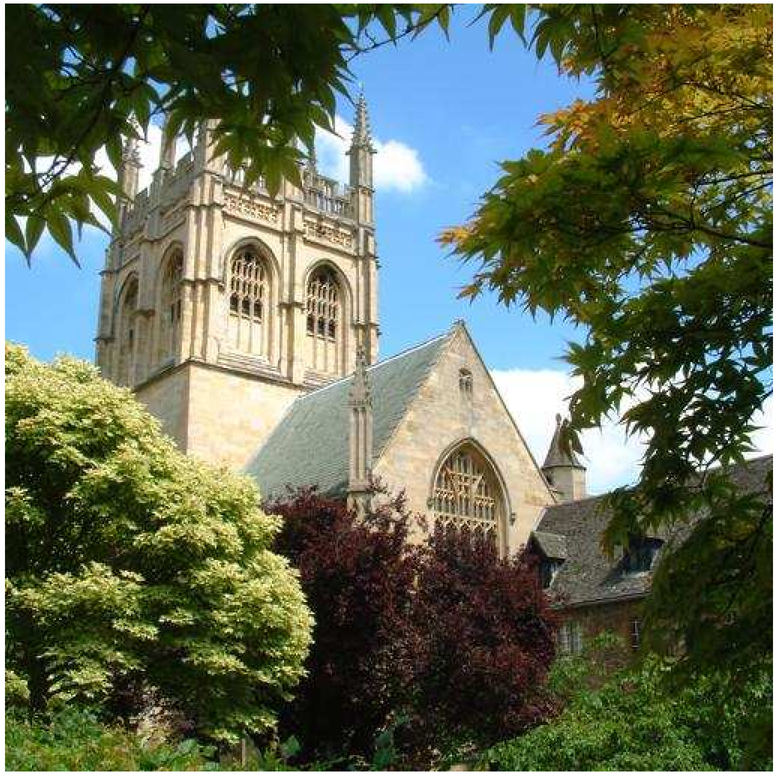
JPEG

In practice, JPEG compression reduces file size with relatively little loss in visual quality.

Bitmap 768KB



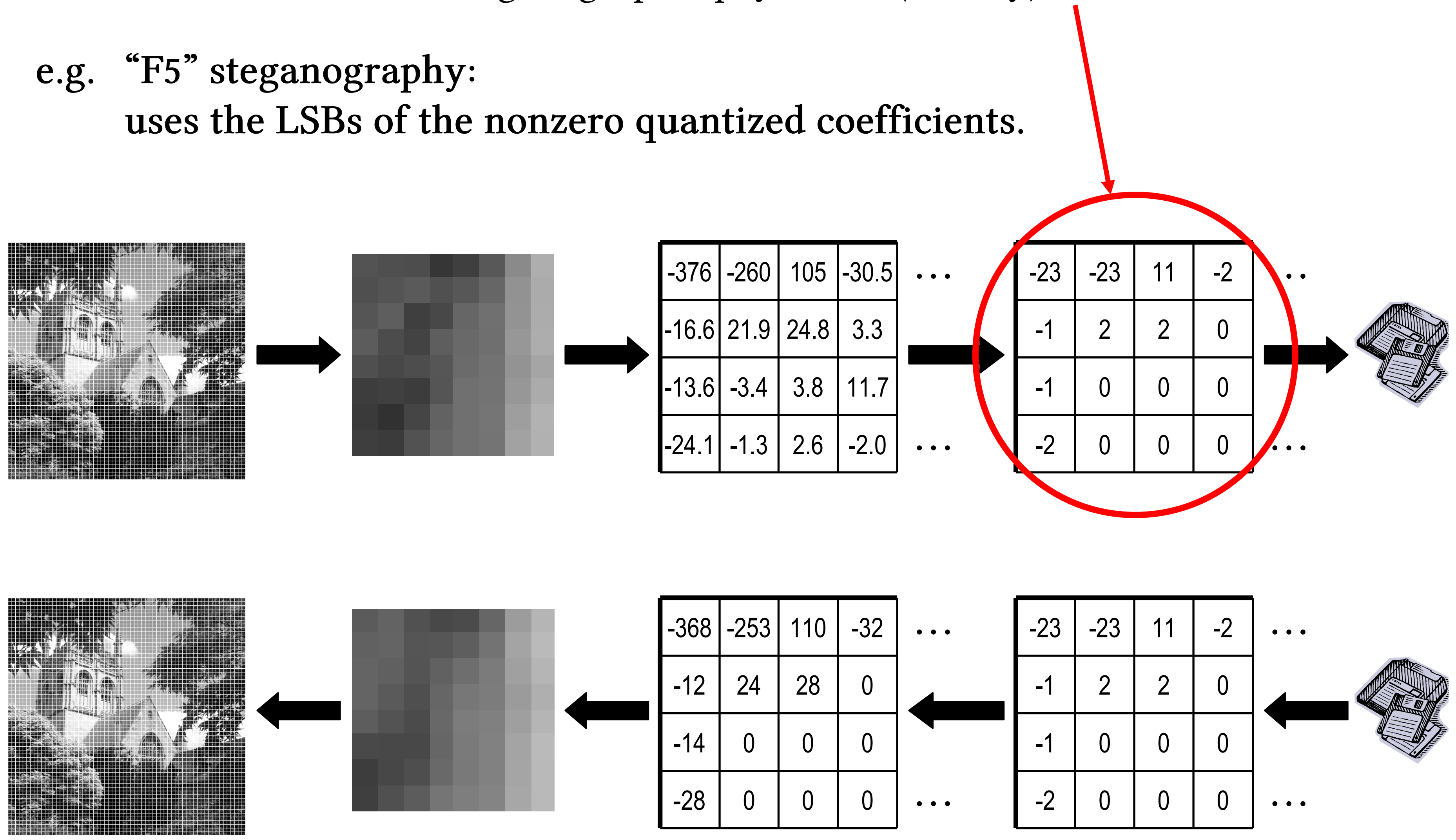
JPEG 66KB



JPEG steganography

It is at this level that the steganographic payload is (usually) embedded.

e.g. “F5” steganography:
uses the LSBs of the nonzero quantized coefficients.



-376	-260	105	-30.5
-16.6	21.9	24.8	3.3
-13.6	-3.4	3.8	11.7
-24.1	-1.3	2.6	-2.0

-23	-23	11	-2
-1	2	2	0
-1	0	0	0
-2	0	0	0

-368	-253	110	-32
-12	24	28	0
-14	0	0	0
-28	0	0	0

-23	-23	11	-2
-1	2	2	0
-1	0	0	0
-2	0	0	0

F5 steganography

Alters LSBs of nonzero quantized coefficients
(there are some slight difficulties with avoiding zeros).

JPEG cover 66KB

Stego object with 8.1KB payload

Can hide information in...

- uncompressed images,
- compressed images,
- audio,
- movies,
- 3D meshes,
- screen savers,
- fonts,
- source code,
- byte code,
- text,
- DNA (!),
- ...

Information Hiding and Covert Communication

Part 2: Steganalysis

- Aims & general overview
  - "Chi-Square" detector for LSB replacement in uncompressed images
- Detection using combinatorial analysis of embedding
  - "Couples" detector for LSB replacement in uncompressed images
- Detection using machine learning
  - "Extended DCT" detector for F5 embedding in JPEG images

Steganalysis

Aim: to detect whether an object contains a covert payload or not.

Steganalysis can be...

- **targeted** at a particular embedding method (most common), or
- **blind**, with potential to unmask even unknown embedding methods (rare, usually weak).

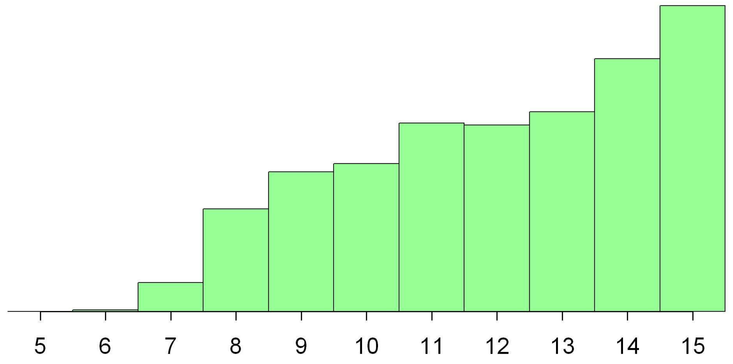
The output can be...

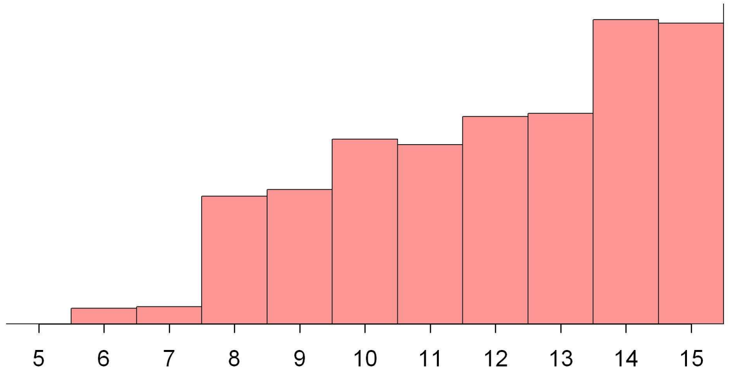
- **simple binary**: yes or no to the presence of payload, or
- **quantitative**, estimating the size of the payload.

Most steganalysers use one of two methodologies:

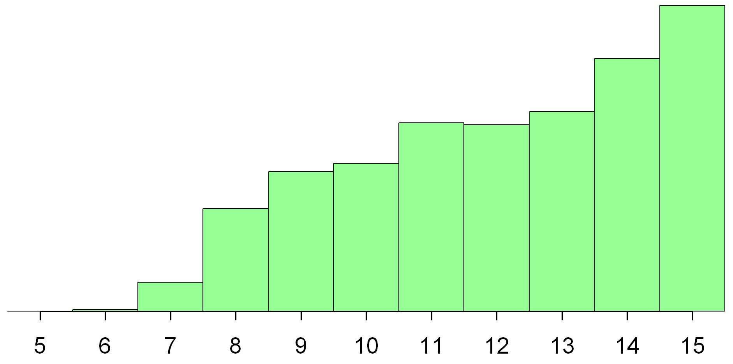
1. Combinatorial analysis of embedding operation (must be targeted).
2. An application of machine learning techniques.

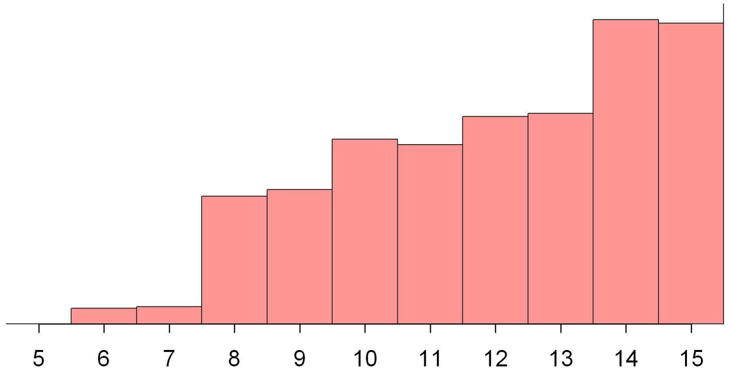
Chi-Square detector





Chi-Square detector





Measure closeness of pairs by the "Chi-Square" statistic:

$$X^2 = \sum_{i=0}^{127} \frac{\big(f[2i] - f[2i+1]\big)^2}{f[2i] + f[2i+1]}$$

(Have to exclude terms with zero or very small denominator.)

High value of $X^2 \longrightarrow$ no payload

Low value of $X^2 \longrightarrow$ suspect payload

A. Westfeld & A. Pfitzmann. *Attacks on Steganographic Systems*. In Proc. 3rd Information Hiding Workshop, Springer LNCS, 1999.

Steganalysis

Aim: to detect whether an object contains a covert payload or not.

Steganalysis can be…

- **targeted** at a particular embedding method (most common), or
- **blind**, with potential to unmask even unknown embedding methods (rare, usually weak).

The output can be…

- **simple binary**: yes or no to the presence of payload, or
- **quantitative**, estimating the size of the payload.

Most steganalysers use one of two methodologies:

1. Combinatorial analysis of embedding operation (must be targeted).
2. An application of machine learning techniques.

Measuring performance

Binary detectors are benchmarked by their false positive / false negative tradeoff "Receiver Operating Characteristic" curve.

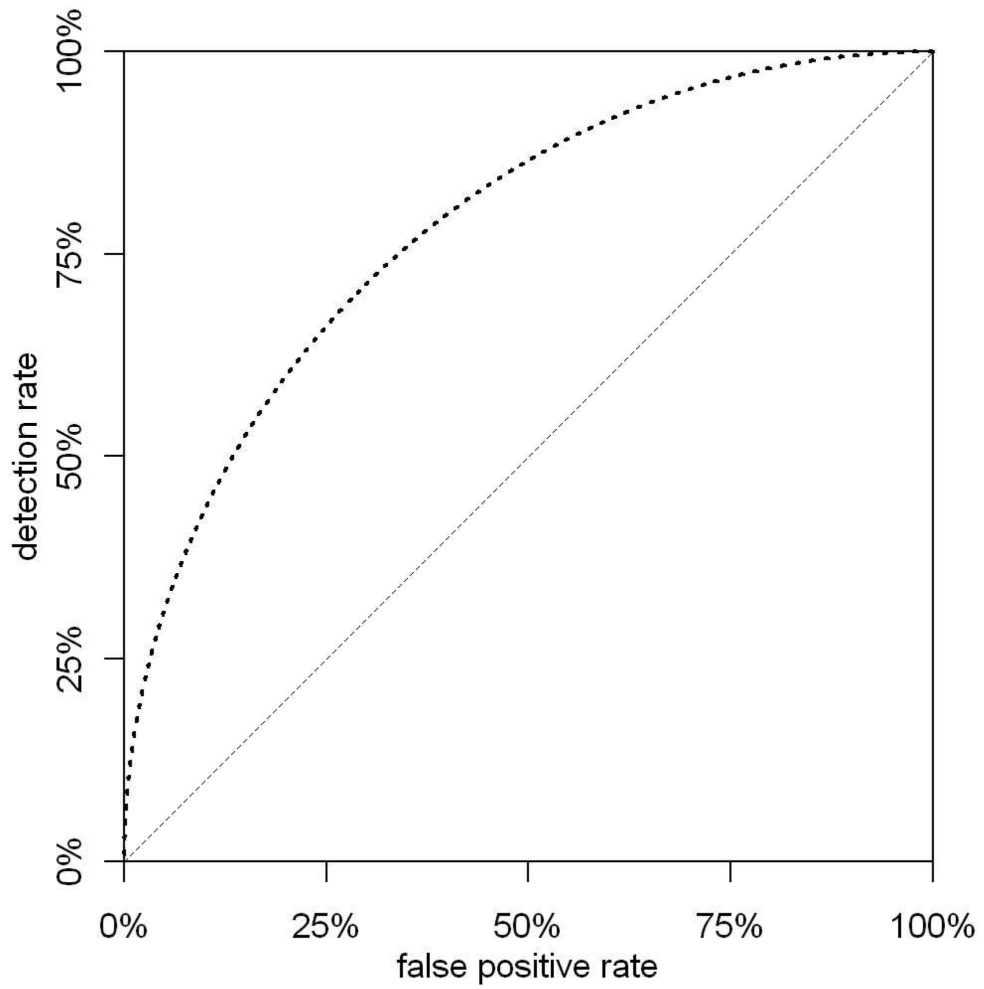


Image library

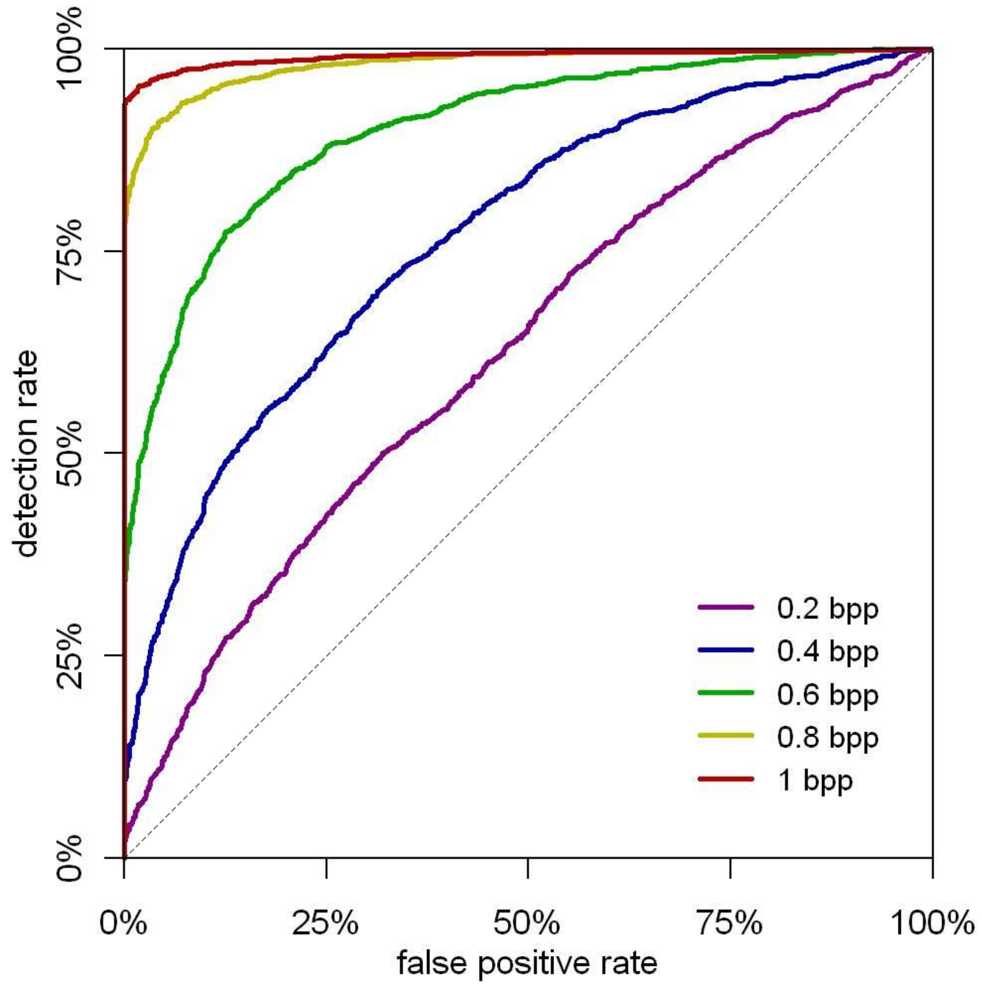
In the absence of a perfect model for covers, we must estimate the ROC empirically.

Here we will use a library of 1600 cover images, each 3 Mpixels, taken in RAW format using a digital camera.

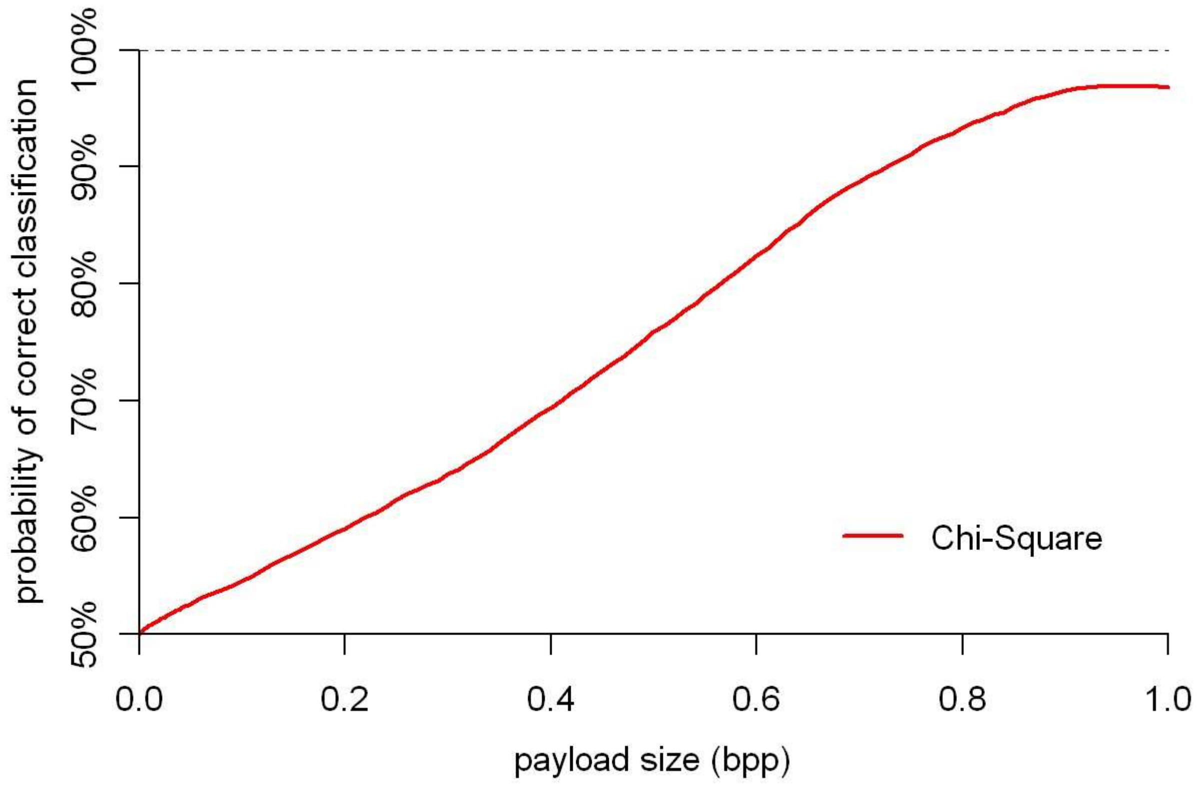
NB: performance might be highly dependent on the characteristics of the covers. Good researchers test on multiple, independent, sets of covers.

R. Böhme & A. Ker. *A Two-Factor Error Model for Quantitative Steganalysis*. In Proc. Electronic Imaging 2006, SPIE.

Performance of Chi-Square



Performance of Chi-Square



Chi-Square is only a **weak** detector for LSB replacement steganography, if

- the payload size is smaller than maximum, and
- the payload is spread pseudorandomly through the cover.

Couples detector

“Couples” is a more recent detector for LSB replacement in uncompressed images. It differs from Chi-Square in that:

1. It has a specific model for certain statistical properties of cover images.
2. It is quantitative (estimates the size of payload).

The detector uses properties of **adjacent pairs** of pixels, to estimate the proportionate payload size.

S. Dumitrescu, X. Wu, & Z. Wang. *Detection of LSB Steganography via Sample Pair Analysis*. In Proc. 5th Information Hiding Workshop, Springer LNCS, 2002.

A. Ker. *A General Framework for the Structural Steganalysis of LSB Replacement*. In Proc. 7th Information Hiding Workshop, Springer LNCS, 2005.

Steganalysis

Aim: to detect whether an object contains a covert payload or not.

Steganalysis can be…

- **targeted** at a particular embedding method **(most common)**, or
- **blind**, with potential to unmask even unknown embedding methods **(rare, usually weak)**.

The output can be…

- **simple binary**: yes or no to the presence of payload, or
- **quantitative**, estimating the size of the payload.

Most steganalysers use one of two methodologies:

1. Combinatorial analysis of embedding operation (must be targeted).
2. An application of machine learning techniques.

Couples detector

We look at ***adjacent pairs of pixels***, and the effects of LSB operations on them.

Definitions (classification of pixel pairs)

$\mathcal{P}$	all adjacent pixel value pairs (x, y).
$\mathcal{C}_m$	pairs with values (x, y) such that $\lfloor x/2 \rfloor - \lfloor y/2 \rfloor = m$.
$\mathcal{E}_m$	pairs with values $(2k, 2k + m)$.
$\mathcal{O}_m$	pairs of the form $(2k + 1, 2k + 1 + m)$.

e.g. if 66 and 72 are the values of two adjacent pixels then this pair is in $\mathcal{P}$, $\mathcal{C}_3$, and $\mathcal{E}_6$.

Trace sets

$\mathcal{P}$	all adjacent pixel value pairs (x, y).
$\mathcal{C}_m$	pairs with values (x, y) such that $\lfloor x/2 \rfloor - \lfloor y/2 \rfloor = m$.
$\mathcal{E}_m$	pairs with values $(2k, 2k + m)$.
$\mathcal{O}_m$	pairs of the form $(2k + 1, 2k + 1 + m)$.

$$\begin{tikzcd}
 & & & & \mathcal{P} \arrow[dlll] \arrow[dl] \arrow[d] \arrow[dr] \arrow[drrr] \arrow[drrrrr] & & & & & \\
\text{Trace sets:} & \mathcal{C}_{-127} & \cdots & \mathcal{C}_{-1} & \mathcal{C}_0 & \mathcal{C}_1 & \cdots & \mathcal{C}_m \arrow[dl] \arrow[d] \arrow[dr] \arrow[drr] & \cdots & \mathcal{C}_{127} \\
 & & & & \text{Trace subsets:} & & \mathcal{O}_{2m-1} & \mathcal{E}_{2m} & \mathcal{O}_{2m} & \mathcal{E}_{2m+1}
\end{tikzcd}$$

Trace sets

Structural Property:

> *LSB replacement moves pairs between trace subsets,*
> *but the trace sets are fixed.*

$$\begin{tikzcd}
 & & & & \mathcal{P} \arrow[dlll] \arrow[dl] \arrow[d] \arrow[dr] \arrow[drrr] \arrow[drrrrr] & & & & & \\
\text{Trace sets:} & \mathcal{C}_{-127} & \cdots & \mathcal{C}_{-1} & \mathcal{C}_0 & \mathcal{C}_1 & \cdots & \mathcal{C}_m \arrow[dll] \arrow[dl] \arrow[d] \arrow[dr] & \cdots & \mathcal{C}_{127} \\
 & & & & \text{Trace subsets:} & \mathcal{O}_{2m-1} & \mathcal{E}_{2m} & \mathcal{O}_{2m} & \mathcal{E}_{2m+1} &
\end{tikzcd}$$

Embedding transitions

Fix m. How are the trace subsets of $\mathcal{C}_m$ affected by LSB operations?

$$\begin{tikzcd}
\mathcal{E}_{2m+1} \arrow[loop left] \arrow[r, leftrightarrow] \arrow[d, leftrightarrow] \arrow[dr, leftrightarrow] & \mathcal{O}_{2m} \arrow[loop right] \arrow[d, leftrightarrow] \arrow[dl, leftrightarrow] \\
\mathcal{E}_{2m} \arrow[loop left] \arrow[r, leftrightarrow] & \mathcal{O}_{2m-1} \arrow[loop right]
\end{tikzcd}$$

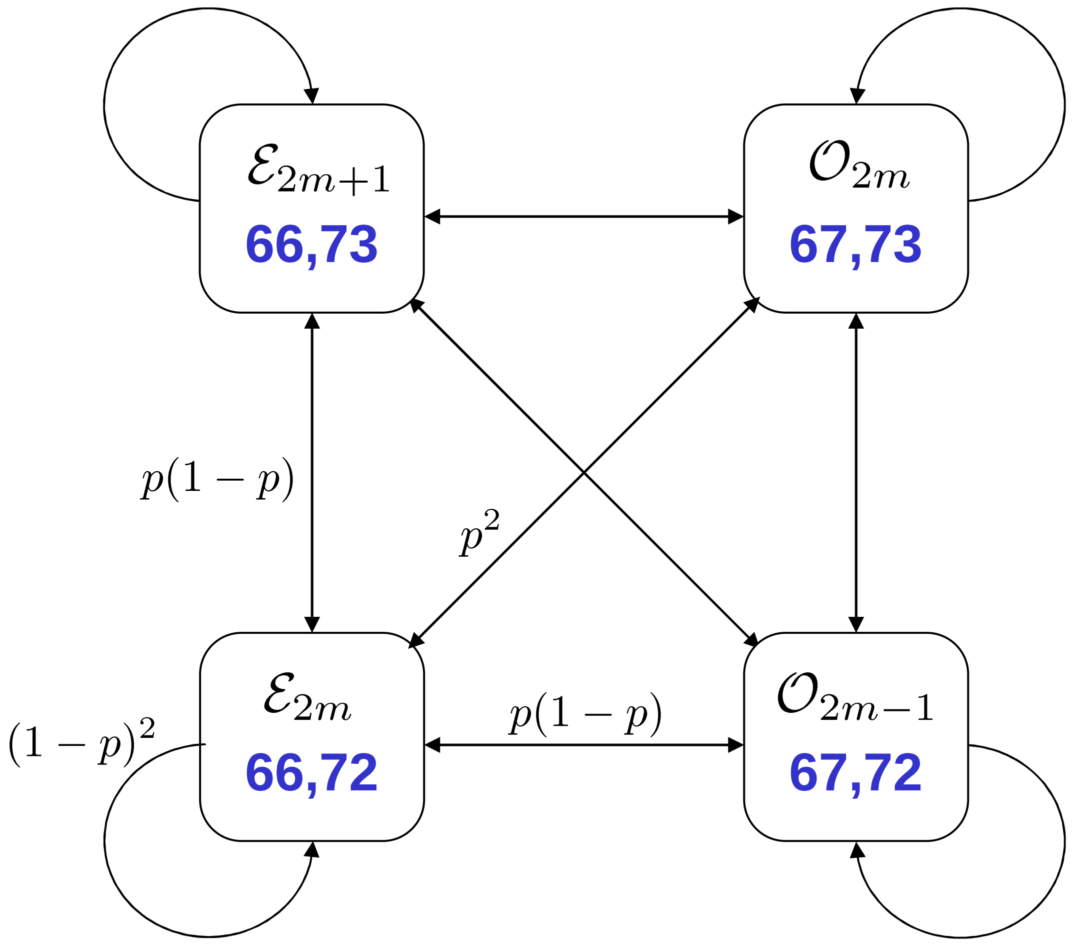
Embedding transitions

Example: some pairs for $m = 3$

$$\begin{tikzcd}
\begin{array}{c}\mathcal{E}_{2m+1}\\ \mathbf{66,73}\end{array} \arrow[loop left] \arrow[r, leftrightarrow] \arrow[d, leftrightarrow] \arrow[dr, leftrightarrow] & \begin{array}{c}\mathcal{O}_{2m}\\ \mathbf{67,73}\end{array} \arrow[loop right] \arrow[d, leftrightarrow] \arrow[dl, leftrightarrow] \\
\begin{array}{c}\mathcal{E}_{2m}\\ \mathbf{66,72}\end{array} \arrow[loop left] \arrow[r, leftrightarrow] & \begin{array}{c}\mathcal{O}_{2m-1}\\ \mathbf{67,72}\end{array} \arrow[loop right]
\end{tikzcd}$$

Embedding transitions

When proportion p LSBs are flipped (at random).


$$\begin{tikzcd}
\boxed{\mathcal{E}_{2m+1}\ \mathbf{66,73}} \arrow[loop left] \arrow[r, leftrightarrow] \arrow[d, leftrightarrow, "p(1-p)"'] \arrow[dr, leftrightarrow] & \boxed{\mathcal{O}_{2m}\ \mathbf{67,73}} \arrow[loop right] \arrow[d, leftrightarrow] \arrow[dl, leftrightarrow, "p^2"'] \\
\boxed{\mathcal{E}_{2m}\ \mathbf{66,72}} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] & \boxed{\mathcal{O}_{2m-1}\ \mathbf{67,72}} \arrow[loop right]
\end{tikzcd}$$

Embedding transitions

Fix a cover of size N. Embed a random message of length $\mathbf{2}pN$.

$$\begin{tikzcd}
\mathcal{E}_{2m+1} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] \arrow[d, leftrightarrow, "p(1-p)"'] \arrow[dr, leftrightarrow, "p^2"] & \mathcal{O}_{2m} \arrow[loop right, "(1-p)^2"] \arrow[d, leftrightarrow, "p(1-p)"] \arrow[dl, leftrightarrow, "p^2"'] \\
\mathcal{E}_{2m} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] & \mathcal{O}_{2m-1} \arrow[loop right, "(1-p)^2"]
\end{tikzcd}$$

Define

$$\begin{aligned}
e_m &= \text{\#pairs in } \mathcal{E}_m \text{ in cover} \\
o_m &= \text{\#pairs in } \mathcal{O}_m \text{ in cover} \\
e'_m &= \text{\#pairs in } \mathcal{E}_m \text{ after embedding} \\
o'_m &= \text{\#pairs in } \mathcal{O}_m \text{ after embedding}
\end{aligned}$$

Then

$$\underset{\substack{\uparrow \\ \text{(this is really the expectation of a random variable)}}}{e'_{2m}} \approx (1-p)^2 e_{2m} + p(1-p)o_{2m-1} + p(1-p)e_{2m+1} + p^2 o_{2m}.$$

Embedding transitions

Fix a cover of size N. Embed a random message of length $2pN$.

$$\begin{tikzcd}
\mathcal{E}_{2m+1} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] \arrow[d, leftrightarrow, "p(1-p)"'] \arrow[dr, leftrightarrow, "p^2"] & \mathcal{O}_{2m} \arrow[loop right, "(1-p)^2"] \arrow[d, leftrightarrow, "p(1-p)"] \arrow[dl, leftrightarrow, "p^2"'] \\
\mathcal{E}_{2m} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] & \mathcal{O}_{2m-1} \arrow[loop right, "(1-p)^2"]
\end{tikzcd}$$

Define

$$\begin{aligned}
e_m &= \#\text{pairs in } \mathcal{E}_m \text{ in cover}\\
o_m &= \#\text{pairs in } \mathcal{O}_m \text{ in cover}\\
e'_m &= \#\text{pairs in } \mathcal{E}_m \text{ after embedding}\\
o'_m &= \#\text{pairs in } \mathcal{O}_m \text{ after embedding}
\end{aligned}$$

Then

$$e'_{2m} \approx (1-p)^2 e_{2m} + p(1-p)o_{2m-1} + p(1-p)e_{2m+1} + p^2 o_{2m}.$$

Embedding transitions

Fix a cover of size N. Embed a random message of length $2pN$.

$$\begin{tikzcd}
\mathcal{E}_{2m+1} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] \arrow[d, leftrightarrow, "p(1-p)"'] \arrow[dr, leftrightarrow, "p^2"] & \mathcal{O}_{2m} \arrow[loop right, "(1-p)^2"] \arrow[d, leftrightarrow, "p(1-p)"] \arrow[dl, leftrightarrow, "p^2"'] \\
\mathcal{E}_{2m} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] & \mathcal{O}_{2m-1} \arrow[loop right, "(1-p)^2"]
\end{tikzcd}$$

Define

$$\begin{aligned}
e_m &= \text{\#pairs in } \mathcal{E}_m \text{ in cover}\\
o_m &= \text{\#pairs in } \mathcal{O}_m \text{ in cover}\\
e'_m &= \text{\#pairs in } \mathcal{E}_m \text{ after embedding}\\
o'_m &= \text{\#pairs in } \mathcal{O}_m \text{ after embedding}
\end{aligned}$$

Then

$$e'_{2m} \approx (1-p)^2 e_{2m} + p(1-p)o_{2m-1} + p(1-p)e_{2m+1} + p^2 o_{2m}.$$

Embedding transitions

Fix a cover of size N. Embed a random message of length $2pN$.

$$\begin{tikzcd}
\mathcal{E}_{2m+1} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] \arrow[d, leftrightarrow, "p(1-p)"'] \arrow[dr, leftrightarrow, "p^2"] & \mathcal{O}_{2m} \arrow[loop right, "(1-p)^2"] \arrow[d, leftrightarrow, "p(1-p)"] \arrow[dl, leftrightarrow, "p^2"'] \\
\mathcal{E}_{2m} \arrow[loop left, "(1-p)^2"] \arrow[r, leftrightarrow, "p(1-p)"] & \mathcal{O}_{2m-1} \arrow[loop right, "(1-p)^2"]
\end{tikzcd}$$

Define

$$\begin{aligned}
e_m &= \text{\#pairs in } \mathcal{E}_m \text{ in cover} \\
o_m &= \text{\#pairs in } \mathcal{O}_m \text{ in cover} \\
e'_m &= \text{\#pairs in } \mathcal{E}_m \text{ after embedding} \\
o'_m &= \text{\#pairs in } \mathcal{O}_m \text{ after embedding}
\end{aligned}$$

Then

$$e'_{2m} \approx (1-p)^2 e_{2m} + p(1-p) o_{2m-1} + p(1-p) e_{2m+1} + p^2 o_{2m}.$$

Inverting the transitions

We derive:

$$\underset{\substack{\uparrow\\ \text{stego}}}{\begin{pmatrix} e'_{2m} \\ o'_{2m-1} \\ e'_{2m+1} \\ o'_{2m} \end{pmatrix}} \approx \begin{pmatrix} (1-p)^2 & p(1-p) & p(1-p) & p^2 \\ p(1-p) & (1-p)^2 & p^2 & p(1-p) \\ p(1-p) & p^2 & (1-p)^2 & p(1-p) \\ p^2 & p(1-p) & p(1-p) & (1-p)^2 \end{pmatrix} \underset{\substack{\uparrow\\ \text{cover}}}{\begin{pmatrix} e_{2m} \\ o_{2m-1} \\ e_{2m+1} \\ o_{2m} \end{pmatrix}}$$

Inverting,

$$\underset{\substack{\uparrow\\ \text{cover}}}{\begin{pmatrix} e_{2m} \\ o_{2m-1} \\ e_{2m+1} \\ o_{2m} \end{pmatrix}} \approx \frac{1}{(1-2p)^2} \begin{pmatrix} (1-p)^2 & -p(1-p) & -p(1-p) & p^2 \\ -p(1-p) & (1-p)^2 & p^2 & -p(1-p) \\ -p(1-p) & p^2 & (1-p)^2 & -p(1-p) \\ p^2 & -p(1-p) & -p(1-p) & (1-p)^2 \end{pmatrix} \underset{\substack{\uparrow\\ \text{stego}}}{\begin{pmatrix} e'_{2m} \\ o'_{2m-1} \\ e'_{2m+1} \\ o'_{2m} \end{pmatrix}}$$

Inverting the transitions

We derive:

$$\underset{\substack{\uparrow\\ \text{stego}}}{\begin{pmatrix} e'_{2m}\\ o'_{2m-1}\\ e'_{2m+1}\\ o'_{2m}\end{pmatrix}} \approx \begin{pmatrix} (1-p)^2 & p(1-p) & p(1-p) & p^2\\ p(1-p) & (1-p)^2 & p^2 & p(1-p)\\ p(1-p) & p^2 & (1-p)^2 & p(1-p)\\ p^2 & p(1-p) & p(1-p) & (1-p)^2 \end{pmatrix} \underset{\substack{\uparrow\\ \text{cover}}}{\begin{pmatrix} e_{2m}\\ o_{2m-1}\\ e_{2m+1}\\ o_{2m}\end{pmatrix}}$$

Inverting,

$$\begin{aligned} e_m &\approx \phi_m(p, \boldsymbol{e}', \boldsymbol{o}') \\ \underset{\substack{\uparrow\\ \text{cover}}}{o_m} &\approx \psi_m(p, \underset{\substack{\uparrow\\ \text{stego}}}{\boldsymbol{e}', \boldsymbol{o}'}) \end{aligned}$$

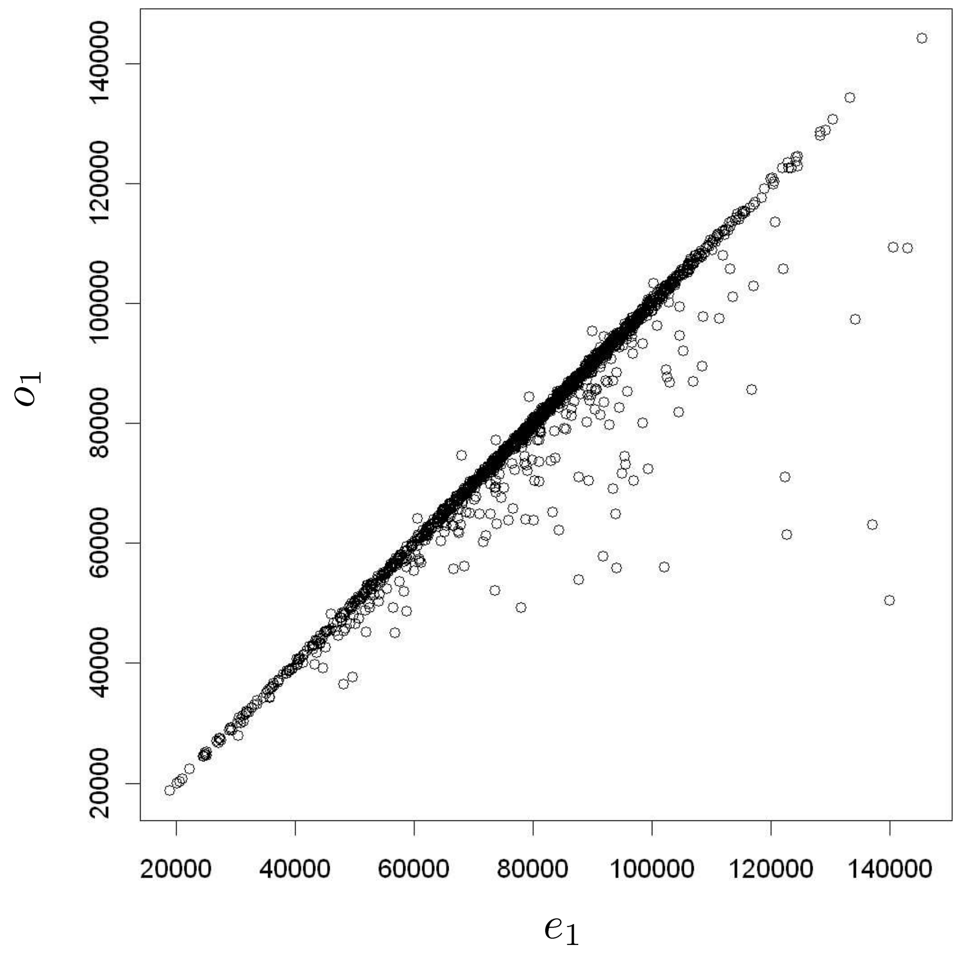
Cover model

In natural images, we believe that $e_m \approx o_m$.

(Why? The difference between the values of each pair should be independent of the parity.)

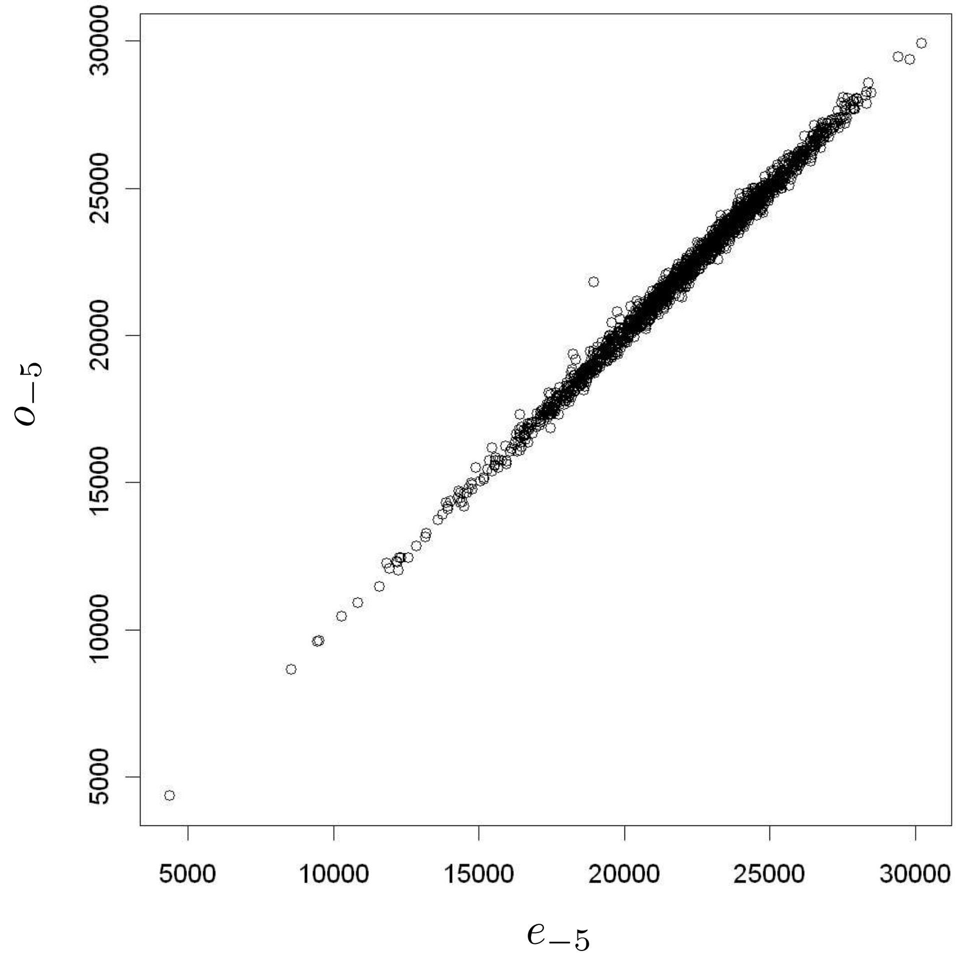
Cover model

In natural images, we believe that $e_m \approx o_m$.



Cover model

In natural images, we believe that $e_m \approx o_m$.



Creating the estimator

For each m, we have

$$\begin{aligned} e_m &\approx o_m \\ e_m &\approx \phi_m(p, \mathbf{e'}, \mathbf{o'}) \\ o_m &\approx \psi_m(p, \mathbf{e'}, \mathbf{o'}) \end{aligned}$$

so

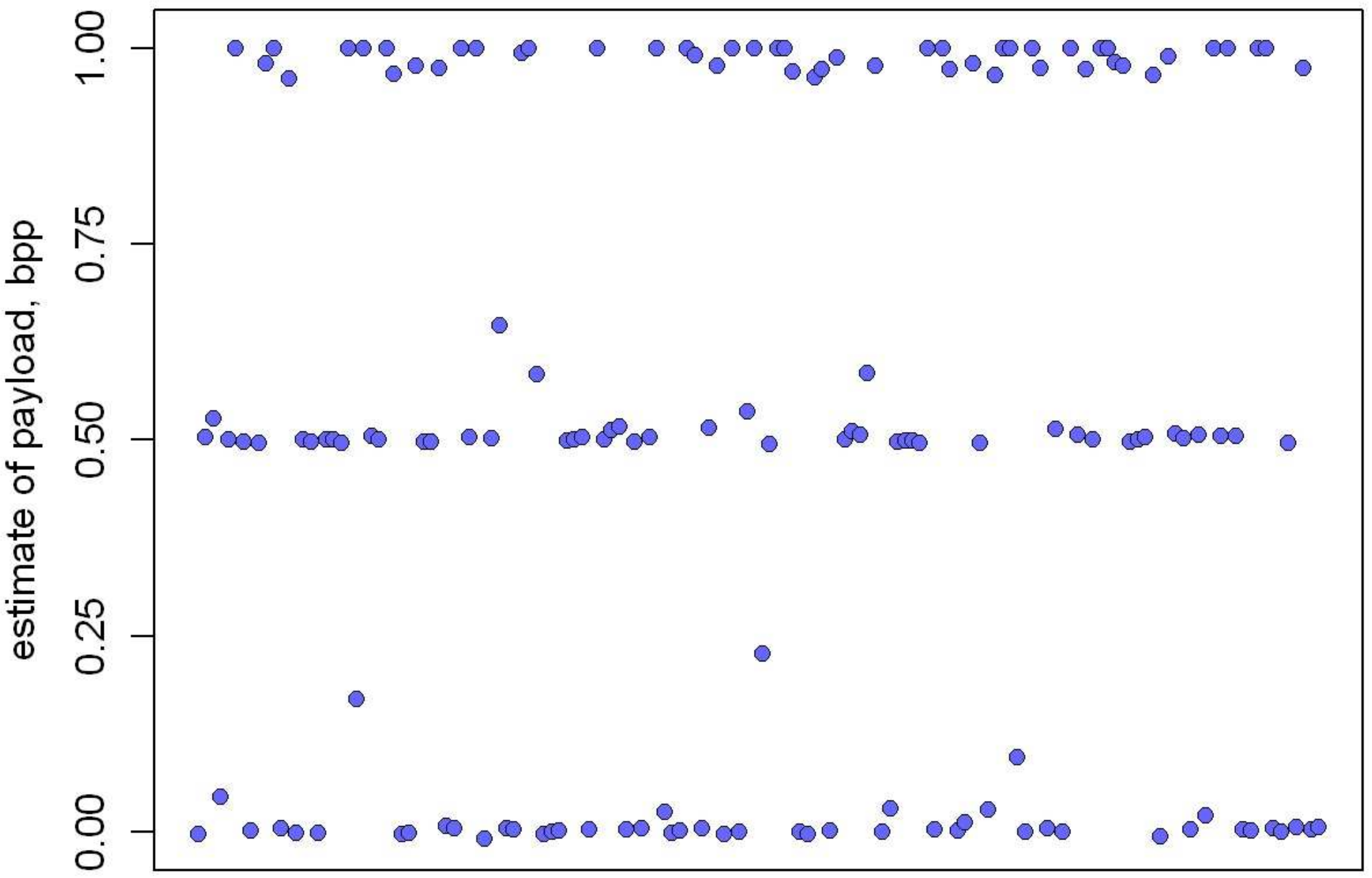
$$0 \quad \approx \quad e_m - o_m \quad \approx \quad \phi_m(p, \mathbf{e'}, \mathbf{o'}) - \psi_m(p, \mathbf{e'}, \mathbf{o'})$$

The Couples estimator for p is the (lower) root of the equation

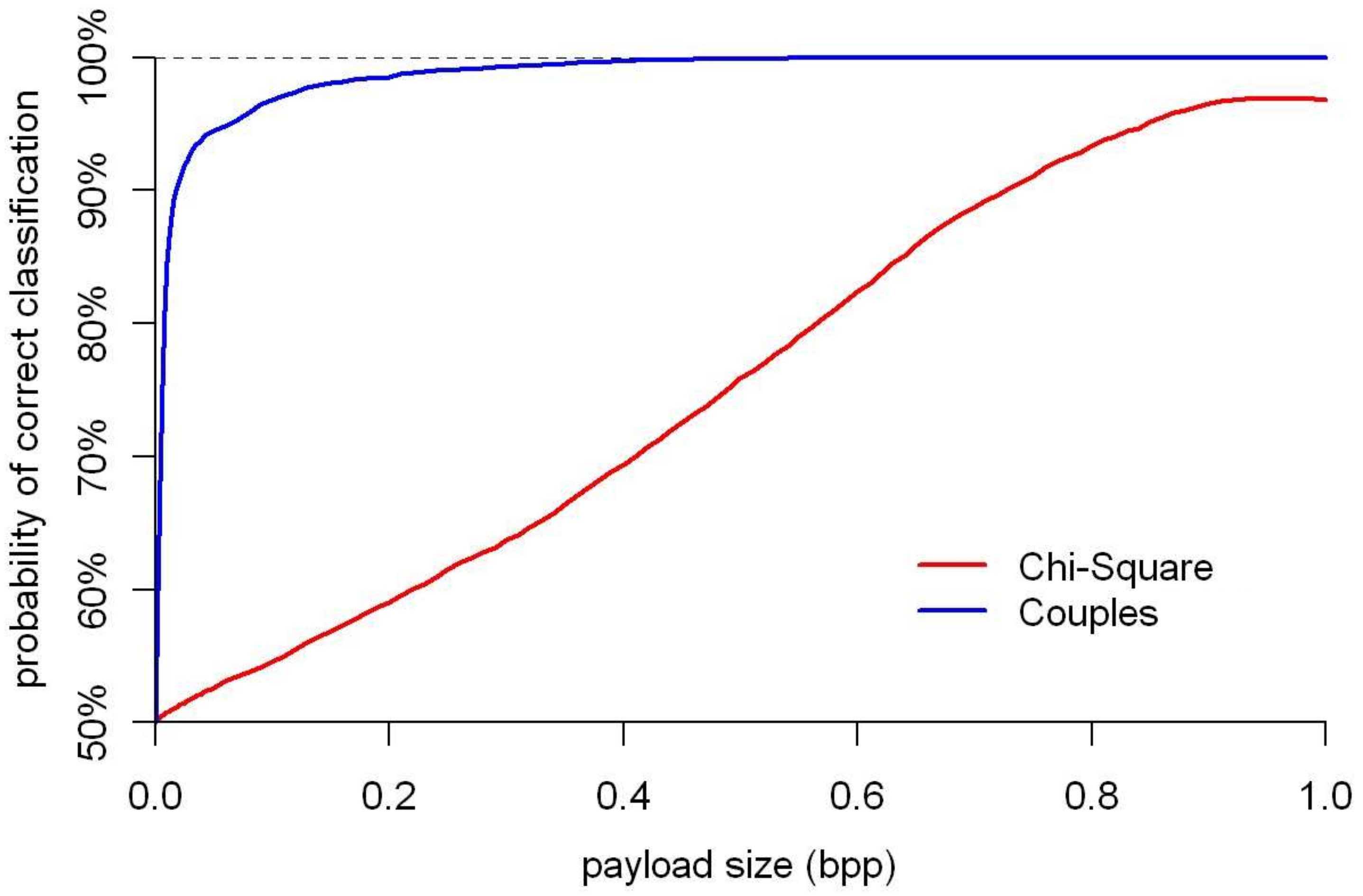
$$0 = \sum_m \phi_m(p, e', o') - \psi_m(p, e', o')$$

Estimator performance

Estimates from 150 images: some had zero LSB payload, some 0.5bpp, some 1bpp.



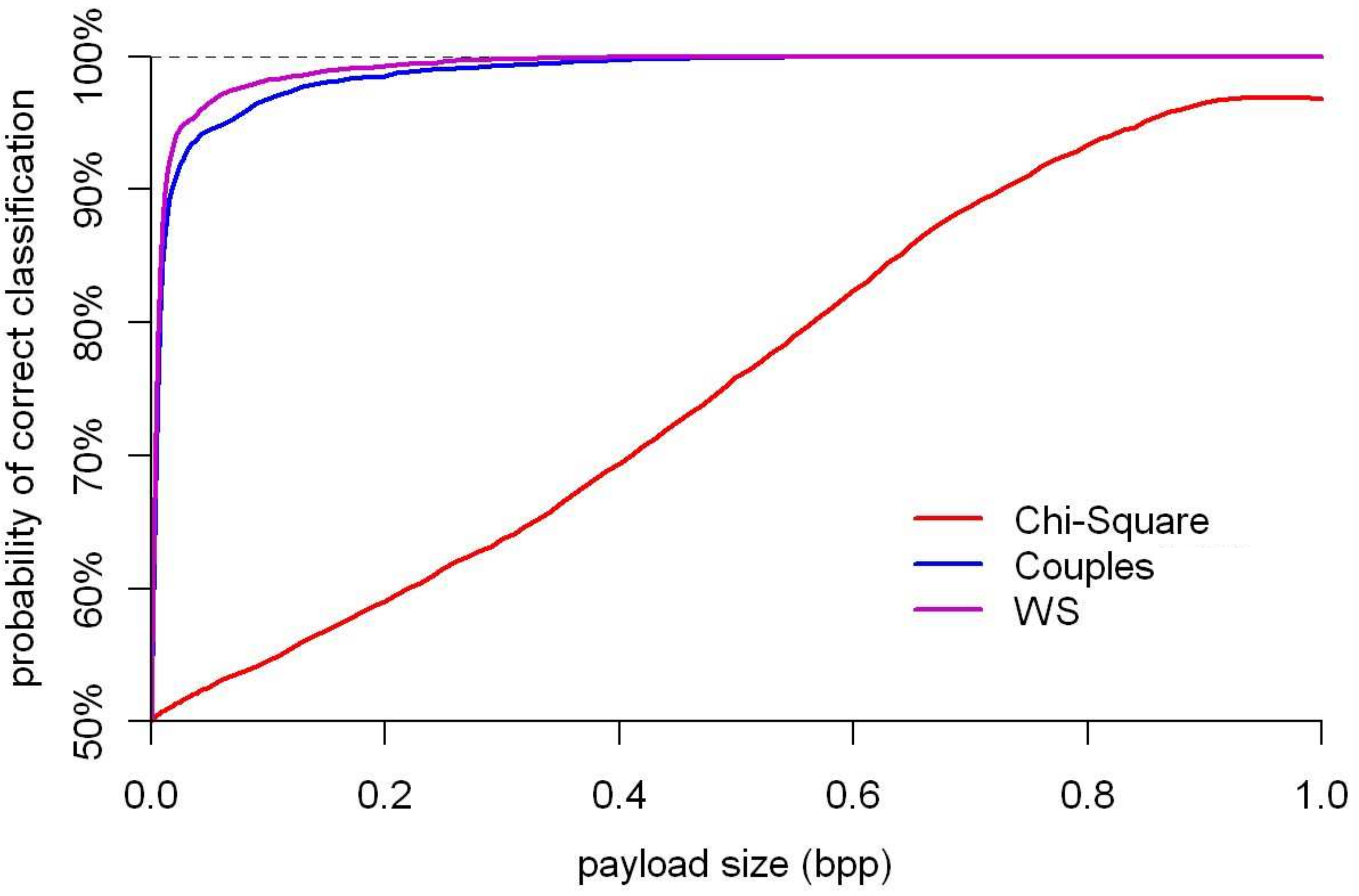
Detector performance



Other detectors for LSB embedding

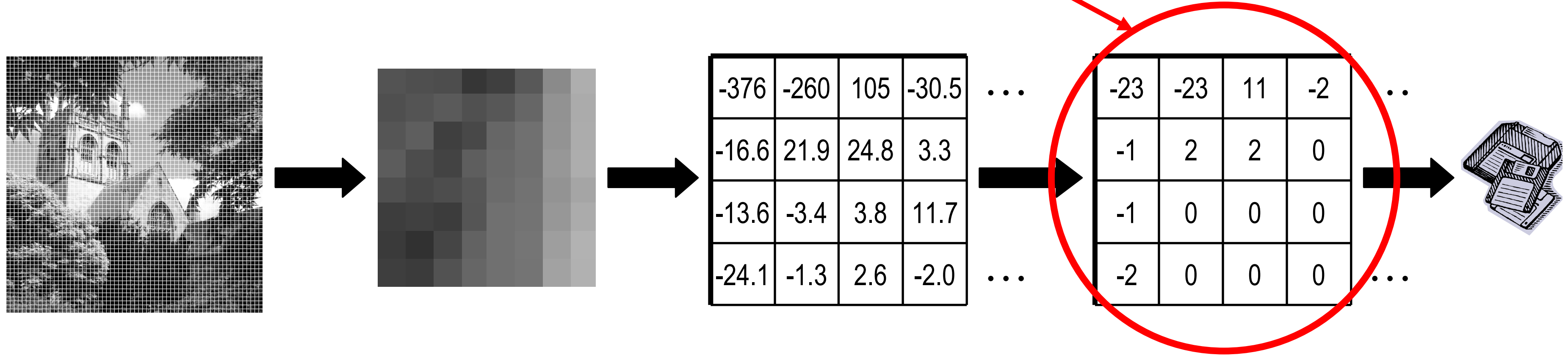
“Histogram Characteristic Function”	Harmsen, 2002; Ker, 2005; …
“Higher Order Statistics”	Lyu & Farid, 2002
“Chi-Square”	Westfeld & Pfitzmann, 1999
“Raw Quick Pairs”	Fridrich et al., 2000
“RS”	Fridrich et al., 2001
“Difference Histogram”	Zhang & Ping, 2003
“Pairs” (for palette images)	Fridrich et al., 2003
“Triples”	Ker, 2005
“Couples/ML”	Ker, 2007
“2Couples” (for embedding in 2 LSBs)	Ker, 2007
“WS”	Fridrich & Goljan, 2004; Ker & Böhme, 2008; Böhme, 2008

Detector performance



F5 steganography

… uses the LSBs of the nonzero quantized coefficients.



-376	-260	105	-30.5
-16.6	21.9	24.8	3.3
-13.6	-3.4	3.8	11.7
-24.1	-1.3	2.6	-2.0

-23	-23	11	-2
-1	2	2	0
-1	0	0	0
-2	0	0	0

$$d^{(i,j,k)}[\,\mathbf{I}\,] = \text{ Quantized coefficient at mode}\,(i,j) \text{ in } 8{\times}8 \text{ block } k$$

A. Westfeld. *F5—A Steganographic Algorithm.* In Proc. 4th Information Hiding Workshop, Springer LNCS, 2001.

F5 detector

- a simplification of the "Extended DCT Feature" classifier due to Pevný & Fridrich.

Rather than examine the F5 embedding operation in detail, this detector uses machine learning (supervised learning for classification) techniques.

T. Pevný & J. Fridrich. *Merging Markov and DCT Features for Multi-Class JPEG Steganalysis*. In Proc. Electronic Imaging 2007, SPIE.

Steganalysis

Aim: to detect whether an object contains a covert payload or not.

Steganalysis can be…

- **targeted** at a particular embedding method **(most common)**, or
- **blind**, with potential to unmask even unknown embedding methods **(rare, usually weak)**.

The output can be…

- **simple binary**: yes or no to the presence of payload, or
- **quantitative**, estimating the size of the payload.

Most steganalysers use one of two methodologies:

1. Combinatorial analysis of embedding operation (must be targeted).
2. An application of machine learning techniques.

Supervised learning for classification

Suppose a universe of objects which fall into discrete, disjoint, classes.

Key elements:

- Select feature vector

  Each object is projected onto a vector of (hopefully) relevant features

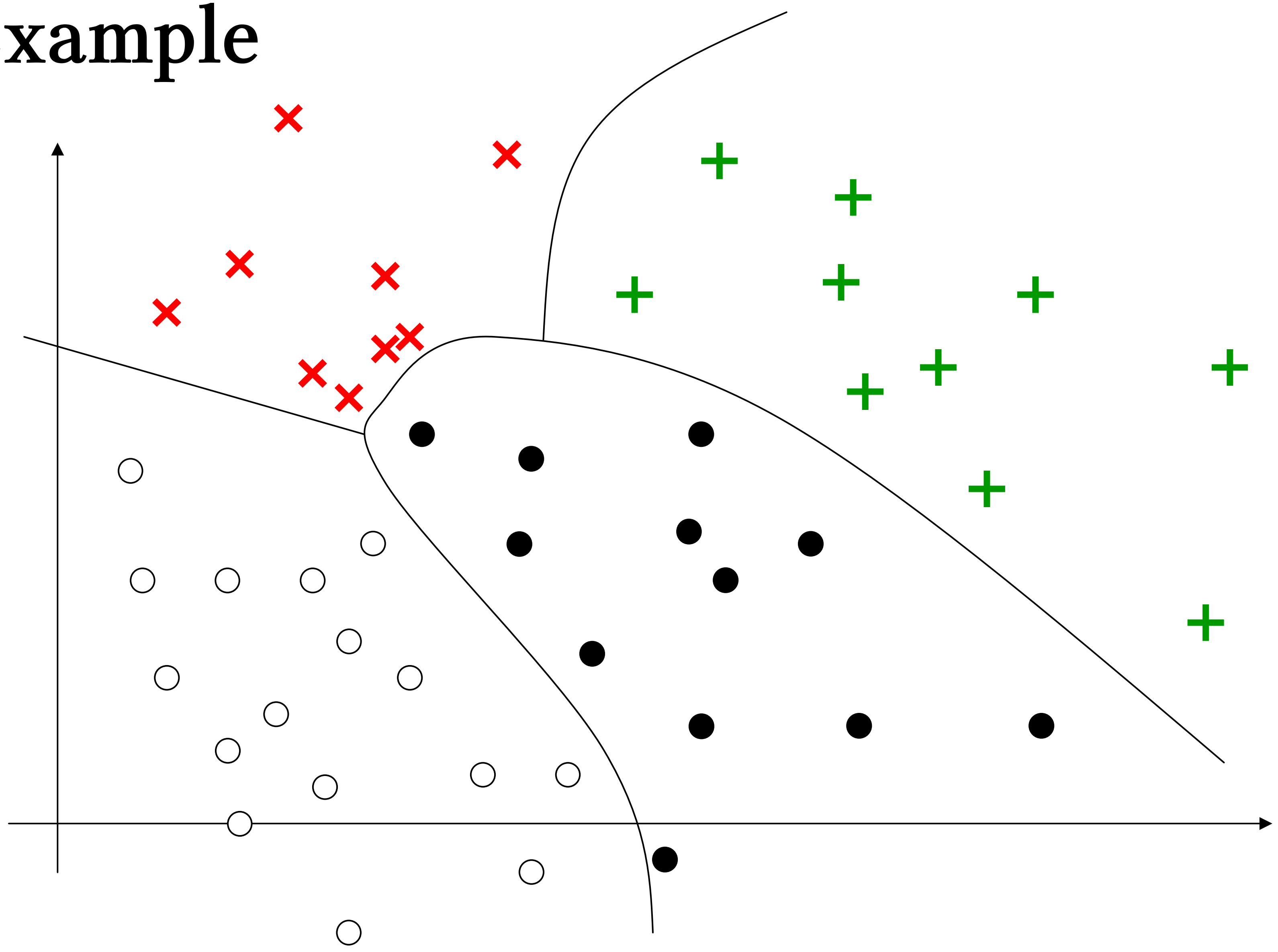
- Training phase

  Separate feature space into class regions based on known objects

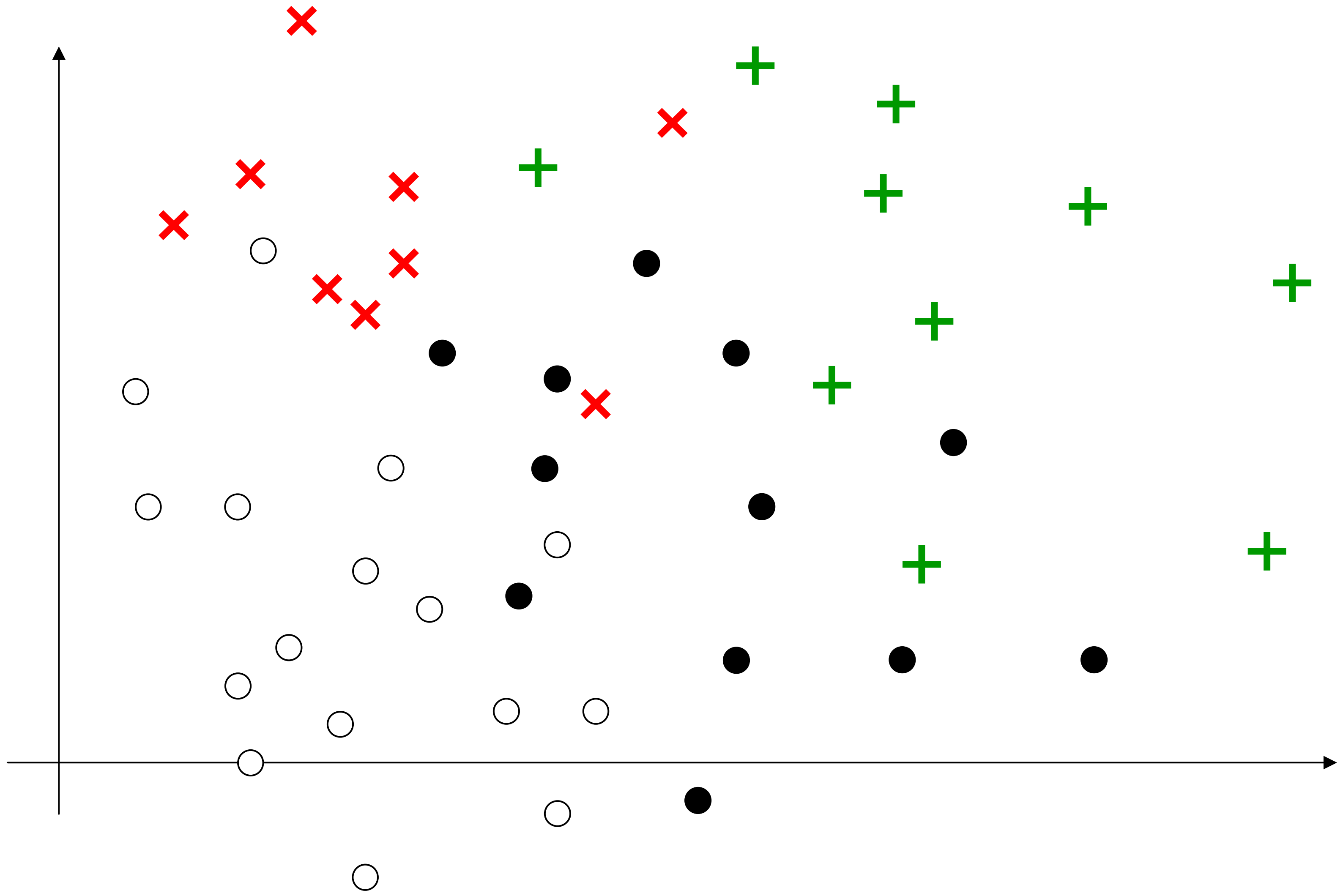
- Application

  Predict class of new objects, based on their features

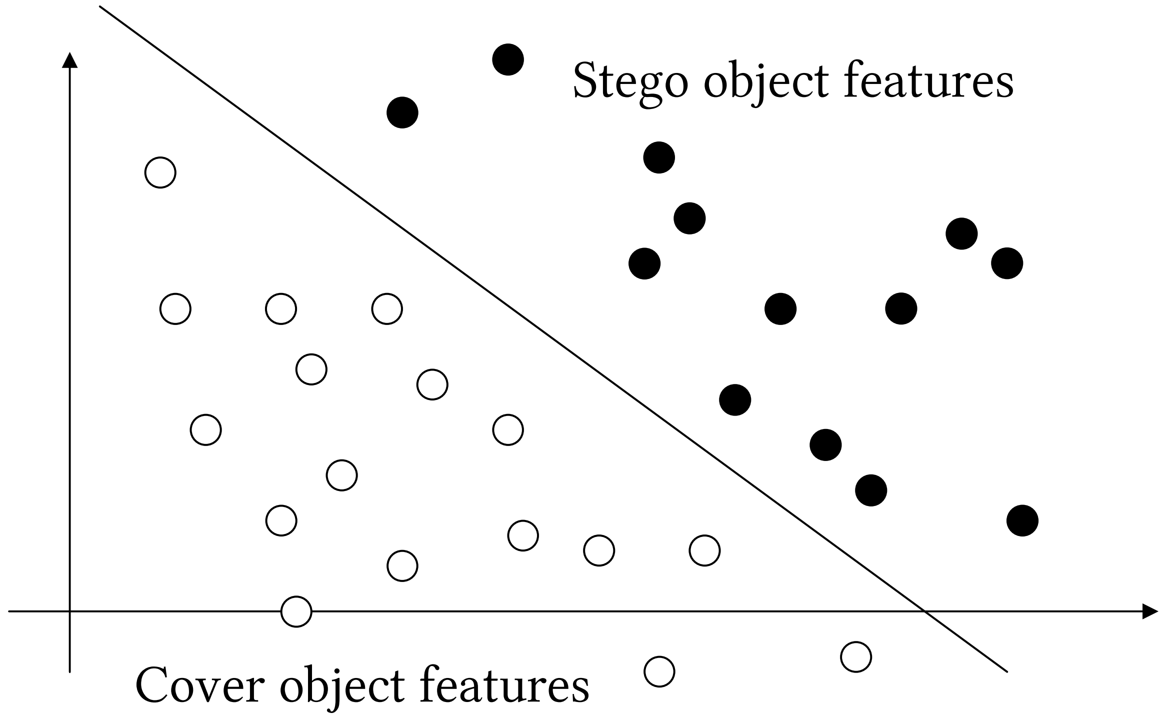
Example



Example



Ideal



DCT features

For features we use the **histogram** of coefficients (for each DCT mode separately):

$$h^{n}_{(i,j)}[\mathbf{I}] = \frac{\#\{d^{(i,j,k)}[\mathbf{I}] = n \mid 1 \le k \le B\}}{B}$$

And also the **"dual histogram"**:

$$g^{n}_{(i,j)}[\mathbf{I}] = \frac{\#\{d^{(i,j,k)}[\mathbf{I}] = n \mid 1 \le k \le B\}}{\#\{d^{(i,j,k)}[\mathbf{I}] = n \mid 1 \le k \le B, 1 \le i', j' \le 8\}}$$

To keep the dimensionality down, we consider only

$$(i, j) \in \{(2,1), (1,2), (2,2), (3,1), (1,3)\}$$
$$-5 \le n \le 5$$

for a total of 110 features.

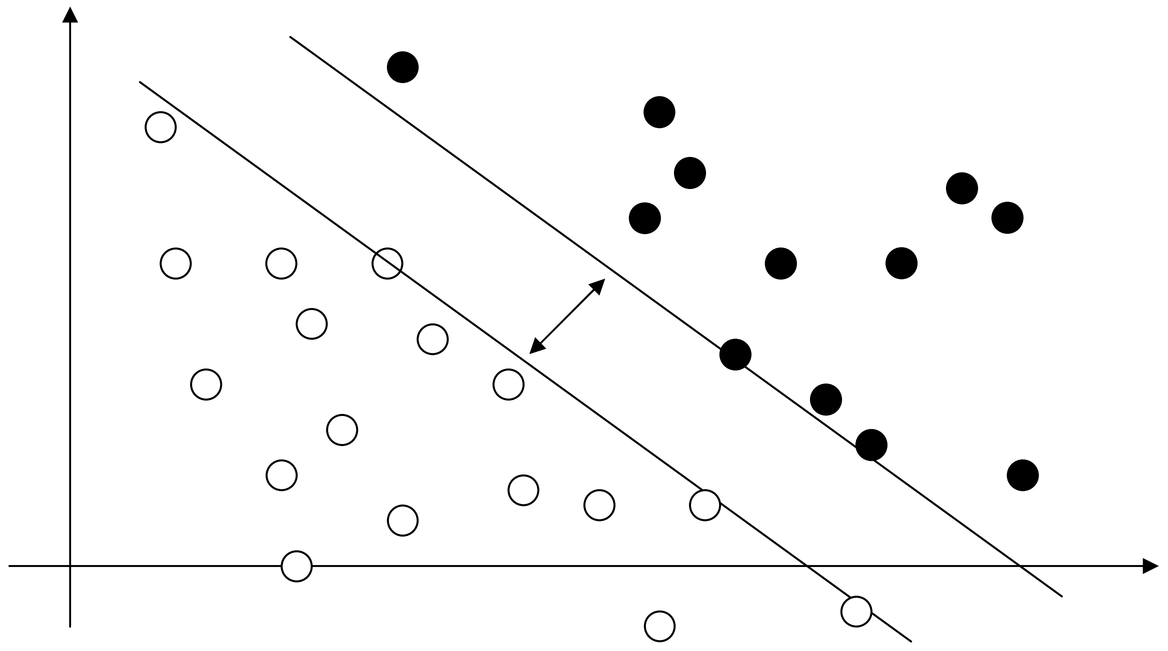
Classification engines

Popular methods to determine the class regions from the training data include:

- Fisher Linear Discriminator
- Multi-layer Perceptron (a.k.a. Neural Network)
- Support Vector Machine
- k-Nearest Neighbours

In most cases the classes are separated by hyperplanes, but the “kernel trick” allows certain types of nonlinear classification at little extra cost.

Support Vector Machine



A SVM finds a separating hyperplane with maximum margin between the classes.

- *When no such hyperplane exists "soft margin" SVMs can be used.*
- *The "kernel trick" allows nonlinear boundaries.*

Performance

To make a steganography detector,

1. Take a set of cover images, and create a set of stego images.
2. Compute the 110 features for every image.
3. Train a SVM on this data (also optimizing the learning parameters).

Test the trained SVM on fresh images. Result... hopeless performance.

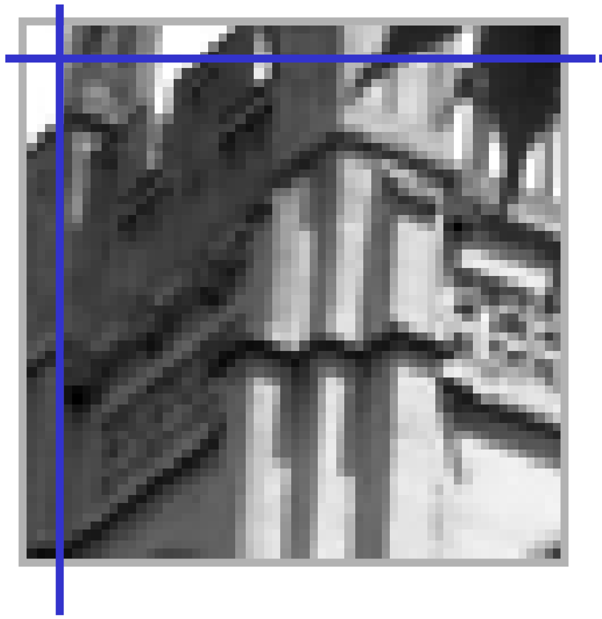
Calibration

We need a rough estimate for feature values of the cover, given the stego object.

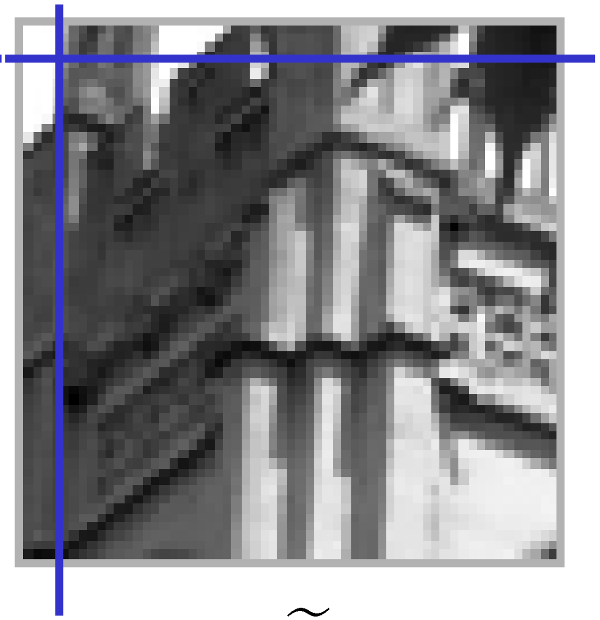
Decompress stego object

$\mathbf{I}$

Crop 4 rows & columns, recompress with same JPEG parameters



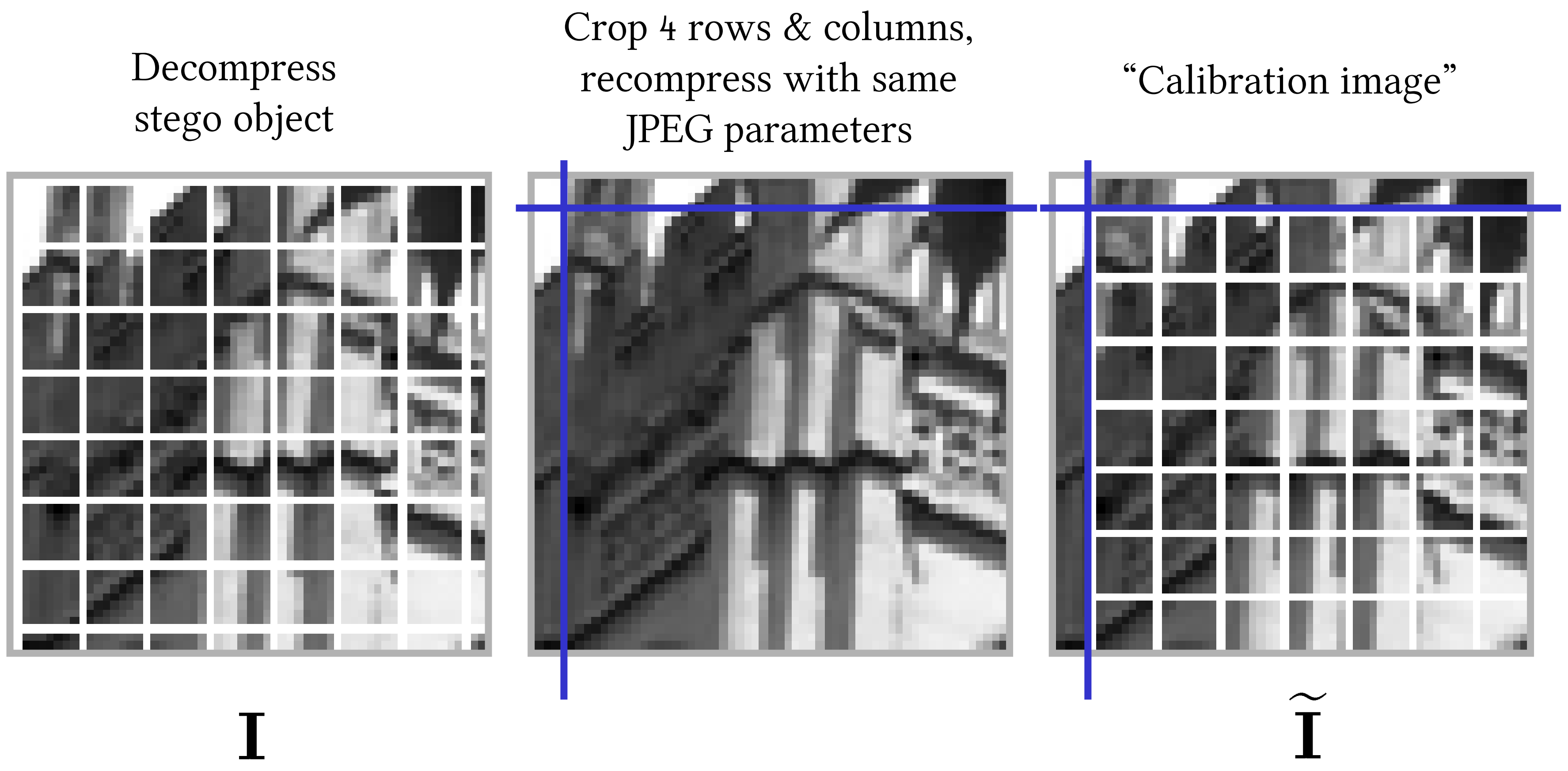
"Calibration image"



$\tilde{\mathbf{I}}$

Calibration

We need a rough estimate for feature values of the cover, given the stego object.



Calibrated features

We use the **calibrated histogram**:

$$h^n_{(i,j)}\left[\,\mathbf{I}\,\right] - h^n_{(i,j)}\left[\,\widetilde{\mathbf{I}}\,\right]$$

And also the **calibrated dual histogram**:

$$g^n_{(i,j)}\left[\,\mathbf{I}\,\right] - g^n_{(i,j)}\left[\,\widetilde{\mathbf{I}}\,\right]$$

For

$$\boldsymbol{(i,j) \in \{(2,1),(1,2),(2,2),(3,1),(1,3)\}}$$
$$\boldsymbol{-5 \le n \le 5}$$

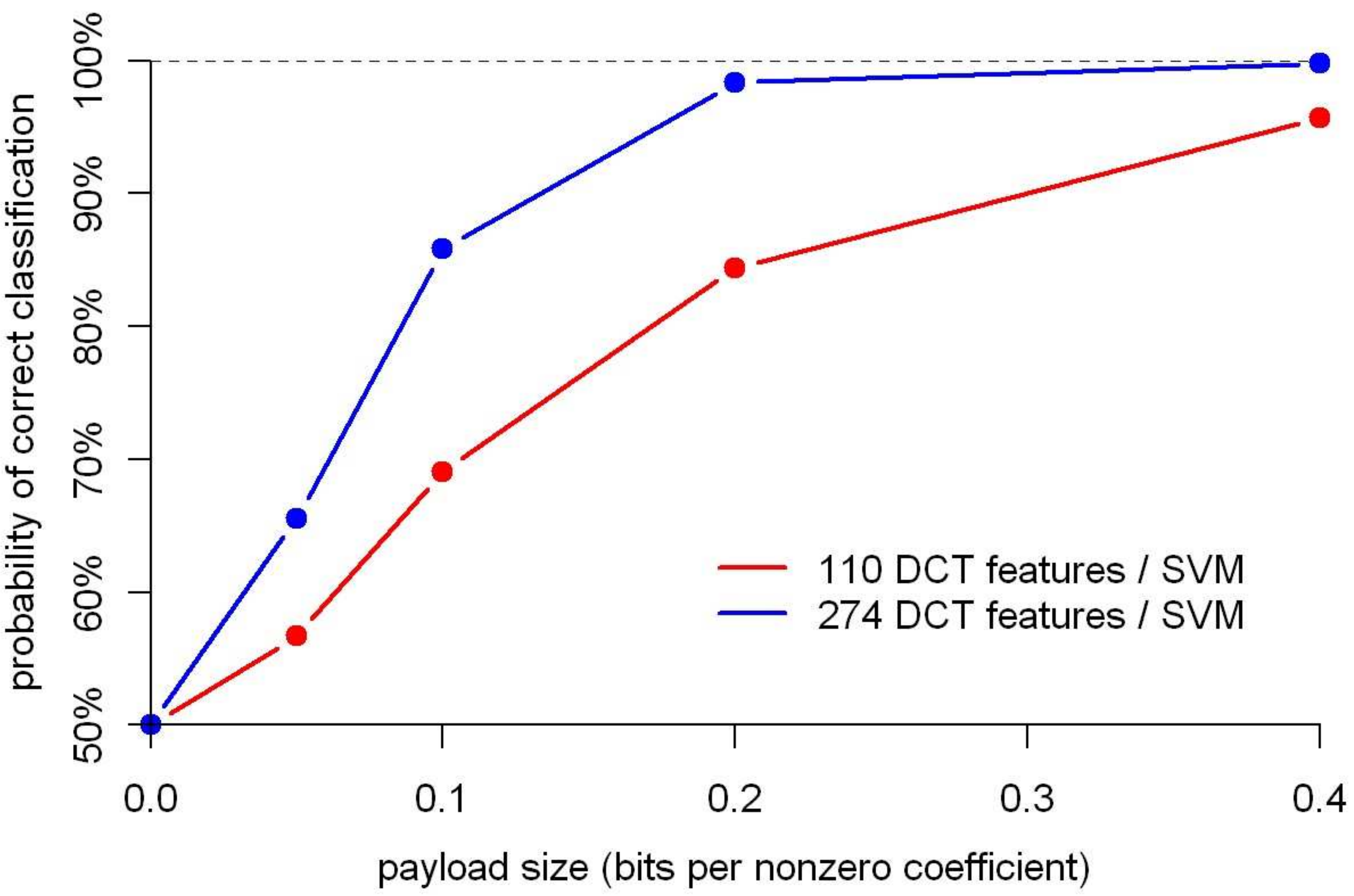
Performance

To make a steganography detector,

1. Take a set of cover images, and create a set of stego images.
2. Compute the 110 calibrated features for every image.
3. Train a SVM on this data (also optimizing the learning parameters).

Test the trained SVM on fresh images. Result...

Performance



Other detectors for F5

(most work for other JPEG embedders too)

Category attack	Lee & Westfeld, 2006 & 2007
"Binary Similarity Measures"	Avcibas et al., 2001
"Higher Order Statistics"	Lyu & Farid, 2002
"KFD"	Harmsen & Pearlman, 2004
23 "DCT features"	Fridrich, 2004
"Markov features"	Shi et al., 2005
"Merged features" (Markov + DCT)	Pevný & Fridrich, 2007

Steganalysis

Most steganalysers use one of two methodologies:

1. Combinatorial analysis of embedding operation.

Advantages	Disadvantages
often highly sensitive	difficult to find cover properties
usually of low computational complexity	can be complex to derive a detector
applicable to many cover types	

2. An application of machine learning techniques.

Advantages	Disadvantages
embedding need not be fully understood	easy to include too many useless features
can utilize standard techniques	often computationally expensive
easy to add new features	different cover types need separate training