Image Data Compression

Steganography and steganalysis

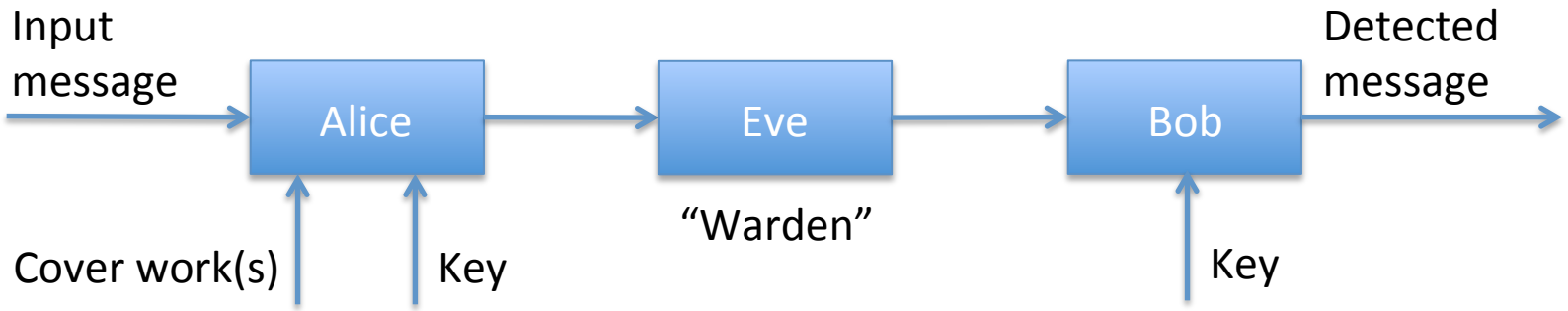
Stenography vs watermarking

Watermarking: *imperceptibly* altering a Work to embed a message about that Work
Steganography: *undetectably* altering a Work to embed a secret message
Steganalysis: detection whether secret steganographic communication is taking place

- Undetectability ≈ no algorithm to determine whether a work contains a hidden message
- No requirements on imperceptibility – changes can be large (if undetectable)
- Cover work has no intrinsic value – e.g. can be selected for convenience from source
- Adversary has no access to original cover work
- Typically longer embedded messages: 1000s of bits vs 10-100 bits in watermarks
- Typically no physical channel noise, but (possible) interference from adversary

Types of steganalysis:

- Forensic – in addition to detection, tries to determine attributes of message or algorithm
- Targeted – assumes specific embedding algorithm (or family of algorithms)
- Blind – generic detection of changes typical to multiple or all stego algorithms



Choice of cover works

- Work chosen from a source of pre-existing works
  - E.g.: a database of innocuous works is used for communication
- Work synthesis: work is created specifically to disguise the message
  - Mimic functions: e-mail message is generated to resemble spam text
  - Text generated from a codebook of random phrases
  - Data masking: synthesized work reproduces properties of innocuous works
- Pre-existing work is selected and modified
  - AKA cover modification: most common and most advanced method

Perfect [image] compression [if it existed] would be a decent steganographic algorithm:

- Every natural image is compressed to some very random-looking string
- In compressed form, no redundancy is left, each bit encodes only relevant information
- When decoded, this string turns into some natural-looking image
- For steganography, use de-compressor: any random string (e.g., a message from Alice) is decompressed into some natural-looking image!

Possible attack:

- Let the message be binary ("yes" or "no"). Among multiple intercepted images, two will be repeated often, which is unlikely for images drawn from their natural distribution.

Moral: for *statistical undetectability*, must reproduce *distribution* of natural images!

Steganographic security: definition [Cachin '98]

Assume that the warden permits Alice to send any work x to Bob, if x is drawn from the probability distribution P_c of unmodified cover works (i.e. probability of x is $P_c(x)$).

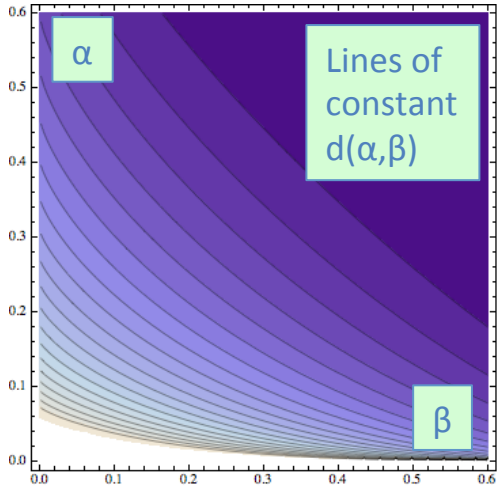
Based on observations, the warden estimates the distribution of sent works P_s. The distribution of cover works P_c is considered known. Then the two distributions can be compared with **Kullback-Leibler divergence** (KLD):

$$D_{KL}\left(P_c \,\|\, P_s\right) = \sum_{x \in C} P_c(x) \cdot \log \frac{P_c(x)}{P_s(x)}$$

If the KLD is zero, the system is declared **perfectly secure**.
Otherwise, if

$$D_{KL}\left(P_c \,\|\, P_s\right) \le \varepsilon,$$

the system is considered **ε-secure**.



Eve's detector outputs binary signal (presence of stego message). Let α be probability of FP error, and β – probability of FN error. KLD between two (binary) distributions is:

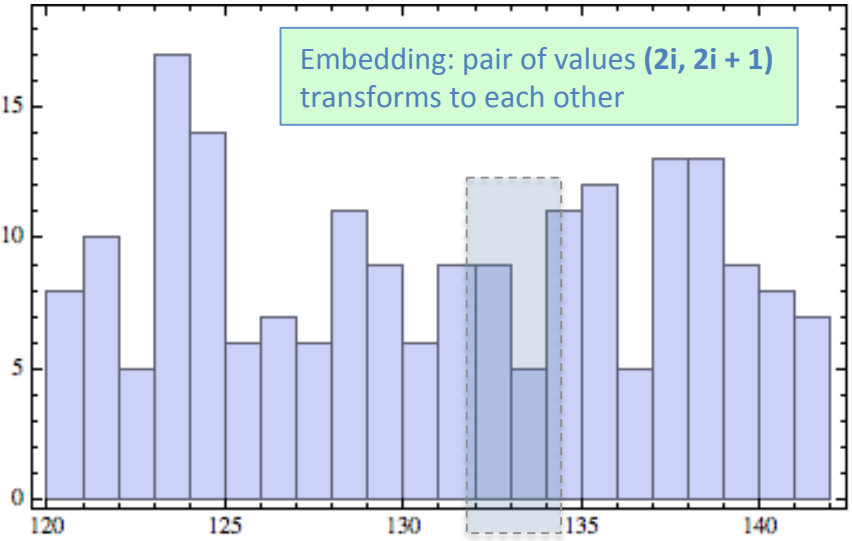
$$d(\alpha,\beta) = (1-\alpha)\log\frac{1-\alpha}{\beta} + \alpha\log\frac{\alpha}{1-\beta}$$

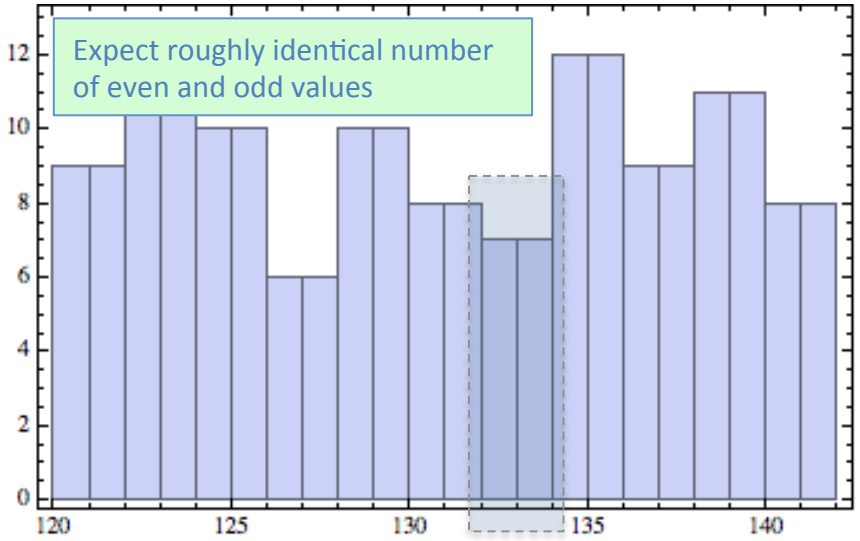
Detection is processing; *KLD may not decrease due to processing*: $d(\alpha,\beta) \le D_{KL}(P_c \,\|\, P_s) \le \varepsilon$
Thus, if e.g. $\alpha = 0$, then $\beta \ge 2^{-\varepsilon}$, etc.

Simplest steganography: LSB embedding

Method: set the least significant bit (LSB) of each pixel value to corresponding message bit

Histogram attack: message embedding introduces artifacts in the histogram of pixel values





How does one detect the presence of a message?

Histogram attack [Westfeld, Pfitzmann 2000]

Let $T_c[i]$ be histogram values of cover work, and $T_s[i]$ - that of stego work ($i = 0,\ldots,255$)
(i.e. $T_c[i]$ = number of pixels that have value i, out of n pixels in the entire work)

- Pair of values $(2i, 2i + 1)$ is closed with respect to embedding transformation:
  - Embedding 1: $2i \to 2i + 1$, $2i + 1 \to 2i + 1$
  - Embedding 0: $2i \to 2i$, $2i + 1 \to 2i$
- Secret message is a random bit stream, i.e. contains 50% of 1s, and 50% of 0s
- Let us embed a message of length qn, with q being the fraction of embedded pixels
  - Expected number of pixels modified from $2i$ to $2i+1$: $0.5\, q\, T_c[2i]$
  - Expected number of pixels modified from $2i+1$ to $2i$: $0.5\, q\, T_c[2i + 1]$

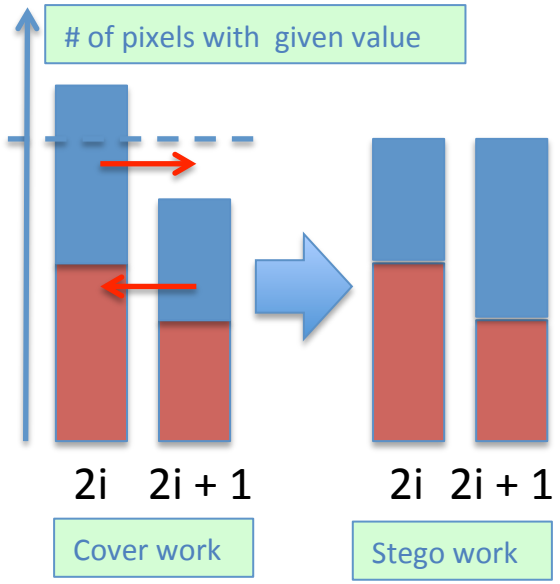
Now, we may find expected values of the resulting histogram:

- $E[\, T_s[2i]\,] = (1 - 0.5\, q)\, T_c[2i] + 0.5\, q\, T_c[2i + 1]$
- $E[\, T_s[2i + 1]\,] = (1 - 0.5\, q)\, T_c[2i + 1] + 0.5\, q\, T_c[2i]$

Full embedding: $q = 1$, and $E[\, T_s[2i]\,] = E[\, T_s[2i + 1]\,]$

Invariant wrt embedding: $S[2i] = T_c[2i] + T_c[2i] = T_s[2i] + T_s[2i + 1]$

- Expected value of histogram bin can be found from stego work itself: $E[\, T_s[2i]\,] = 0.5\, S[2i]$



How do we then detect "un-naturalness" of given histogram?

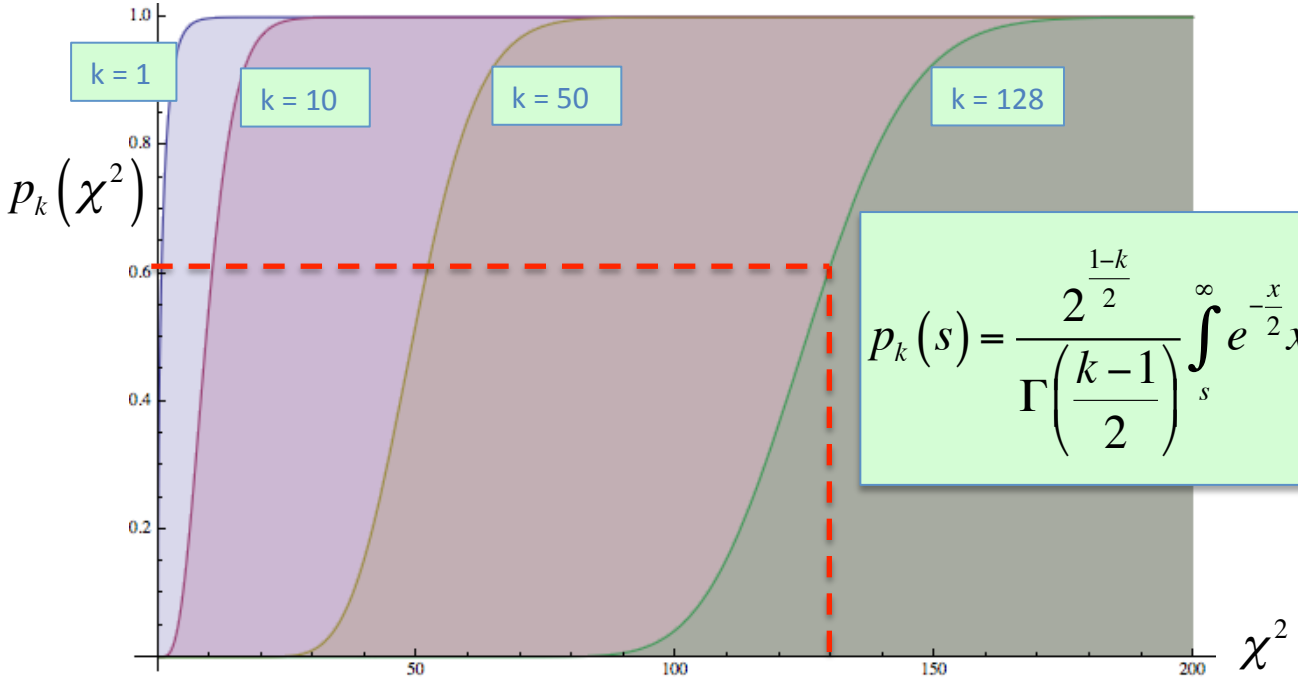
Histogram attack [Westfeld, Pfitzmann 2000]

Pearson's chi-squared test

- Build test statistic:

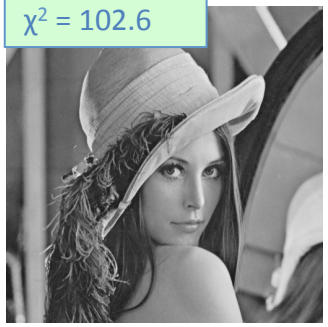
$$\chi^2 = \sum_{i=0}^{k} \frac{\left(T_s[2i] - \bar{T}_s[2i]\right)^2}{\bar{T}_s[2i]}, \qquad \bar{T}_s[2i] = E\left[T_s[2i]\right] = \frac{1}{2}\left(T_s[2i] + T_s[2i+1]\right)$$

- For k – 1 = 127 degrees of freedom, compute corresponding **p-value**:

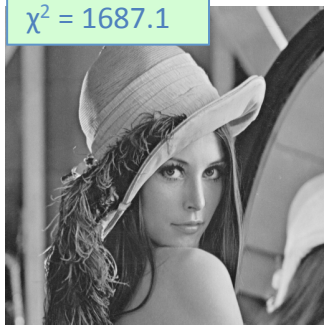


$$p_k(s) = \frac{2^{\frac{1-k}{2}}}{\Gamma\left(\frac{k-1}{2}\right)} \int_s^{\infty} e^{-\frac{x}{2}} x^{\frac{k-3}{2}}\, dx$$

χ^2 = 102.6



χ^2 = 1687.1



Example:

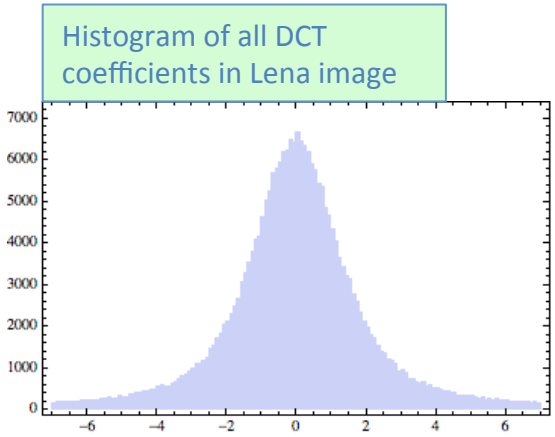
- If chi-squared is 130, then the probability that the work contains a fully embedded message (q = 1) is ≈ 60%.

Assumptions: independent histogram bin counts, chi-square distribution of bin values, …

Statistics-preserving steganography: OutGuess [Provos '01]

Main idea:

- Embed message in LSB of DCT coefficients (8x8 blocks)
- Coefficients along a pseudo-random path (seed is the stego key)
- Skip coefficients that are zero or one, to avoid large distortions
- Three passes through coefficients:
  1. Determine # of suitable coefficients, max message length
  2. Embed message in chosen coefficients
  3. Modify remaining coefficients to restore original histogram



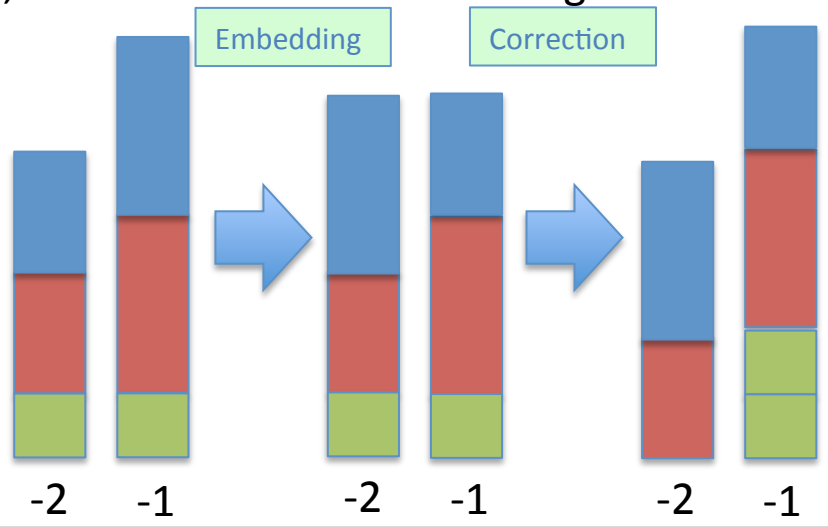
Assume some quantization and binning of n suitable coefficients, embed message of qn bits.

- Pair (0, 1) is not used; most abundant and unbalanced suitable pair: (-2, 1).
- Decrease in $T_c[-1]$ due to embedding: $E[\, T_c[-1] - T_s[-1] \,] = 0.5\, q\, (T_c[-1] - T_c[-2])$.
- # of unused coefficients with value -2: $(1 - q)\, T_c[-2]$, switch to -1 to correct histogram:

$$(1 - q)\, T_c[-2] \geq 0.5\, q\, (T_c[-1] - T_c[-2]),$$

$$q \leq 2\, T_c[-2] / (T_c[-1] + T_c[-2]).$$

- Other pairs can be corrected with the same step!
- Need some care with almost-empty bins (statistically insignificant variations may remain)



Sample pair analysis [Dumitrescu, Wu 2005]

Main idea: detect changes in pairwise correlation of pixel values (statistics beyond histogram!)

- Let P = {(i, j)} be set of pairs of all adjacent pixels in image, v[i] be i-th pixel value, and

$$X = \{(i,j) \in P \mid [(v[j] = 2m)\, AND\, (v[i] < v[j])]\, OR\, [(v[j] = 2m+1)\, AND\, (v[i] > v[j])]\}$$

$$Y = \{(i,j) \in P \mid [(v[j] = 2m)\, AND\, (v[i] > v[j])]\, OR\, [(v[j] = 2m+1)\, AND\, (v[i] < v[j])]\}$$

$$Z = \{(i,j) \in P \mid v[i] = v[j]\}$$

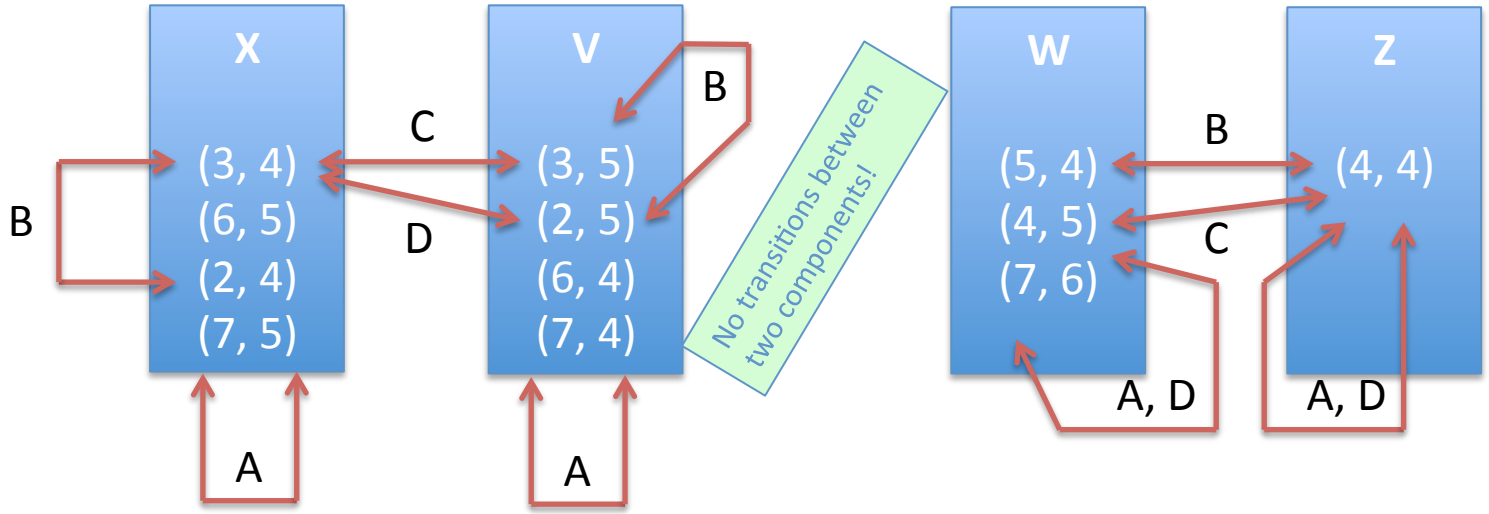
$$W = \{(i,j) \in Y \mid |v[i] - v[j]| = 1\}$$

$$V = \{(i,j) \in Y \mid |v[i] - v[j]| \neq 1\}$$

Complete partitioning:
P = X ∪ Z ∪ Y
= X ∪ Z ∪ W ∪ V

Effect of LSB embedding on pair (i, j):
A) Both i and j remain un-modified
B) Only i is modified
C) Only j is modified
D) Both i and j are modified

- A modification moves pair from one set to the other:



Sample pair analysis [Dumitrescu, Wu 2005]

Let ρ(T, S) be fraction of pixel pairs in set S = P, X, Y, … affected by transition T = A, B, C, or D.

- If relative length of embedded message is q (i.e. message length is qn bits), then

$$\rho(A,P)=\left(1-\frac{q}{2}\right)^2,\rho(B,P)=\rho(C,P)=\frac{q}{2}\left(1-\frac{q}{2}\right),\rho(D,P)=\left(\frac{q}{2}\right)^2$$

- Before [non-adaptive] embedding, all pairs are equally likely to receive modifications:

$$\rho(T,X)=\rho(T,Y)=\rho(T,W)=\rho(T,V)=\rho(T,Z)=\rho(T,P)$$

- After modification:

$$|X'|=|X|\big(\rho(A,X)+\rho(B,X)\big)+|V|\big(\rho(C,V)+\rho(D,V)\big)=|X|\left(1-\frac{q}{2}\right)+|V|\frac{q}{2},$$

$$|V'|=|V|\big(\rho(A,V)+\rho(B,V)\big)+|X|\big(\rho(C,X)+\rho(D,X)\big)=|V|\left(1-\frac{q}{2}\right)+|X|\frac{q}{2},$$

$$|W'|=|W|\big(\rho(A,W)+\rho(D,W)\big)+|Z|\big(\rho(B,Z)+\rho(C,Z)\big)=|W|\left(1-q+\frac{q^2}{2}\right)+|Z|q\left(1-\frac{q}{2}\right).$$

- Additional constraints:

$$|X|=|Y|=|V|+|W|,$$
$$|P|=|X'|+|V'|+|W'|+|Z'|=|X|+|V|+|W|+|Z|$$

- And finally:

$$\left(\frac{|W'|+|Z'|}{2}\right)q^2+\big(2|X'|-|P|\big)q+\big(|Y'|-|X'|\big)=0$$

Simple quadratic equation!

Model-preserving steganography

Instead of specific statistic, let us attempt to keep some stochastic data model intact

- Cover work is modeled as pair of random variables, c_{inv} and c_{emb}
- c_{inv} does not change during embedding, used by detector to determine model for c_{emb}
- Model for cover works: $p(c_{emb} | c_{inv})$

Consider a subset $L(C_{inv})$ of work elements (e.g. pixels) with specific value $c_{inv} = C_{inv}$

- Random message (with 50% 1s and 50% 0s) is run through entropy decoder (e.g. AC), so that probabilities of 1s and 0s in output stream match $p(1 | C_{inv})$ and $p(0 | C_{inv})$
- After embedding, obtain stego work with similar statistical properties!

Fraction of bits that can be embedded in $L(C_{inv})$ equals then conditional entropy of the model:

$$H\left[p\left(c_{emb} \mid c_{inv} = C_{inv}\right)\right] = -\sum_{C_{emb}} p(C_{emb} \mid C_{inv}) \cdot \log_2\left[p(C_{emb} \mid C_{inv})\right]$$

And the total embedding capacity is:

$$R = \sum_{C_{inv}} \left|L\left(C_{inv}\right)\right| \cdot H\left[p\left(c_{emb} \mid c_{inv} = C_{inv}\right)\right]$$

Real-life example algorithm:

- Let c_{inv} be the 7 MSBs in 8-bit DCT coefficient (2, 2), c_{emb} be its LSB, and its histogram $T_c[i]$
- Use invariant $S[2i] = Tc[2i] + Tc[2i + 1]$ to fit some model $f[i]$ via known points $(2i, S[2i])$
- Read out expected values $T_e[i] = 0.5\ f[i]$ for all integer values i
- **Model:** for each value of $MSB(2i)$, set $p(c_{emb} = 0 | c_{inv} = MSB(2i)) = T_e[2i]/(T_e[2i] + T_e[2i + 1])$
- Use AC to decompress part of message to length $S[2i]$, embed into all corresponding pixels

Blind steganalysis using calibration

Generic approach to blind detection:

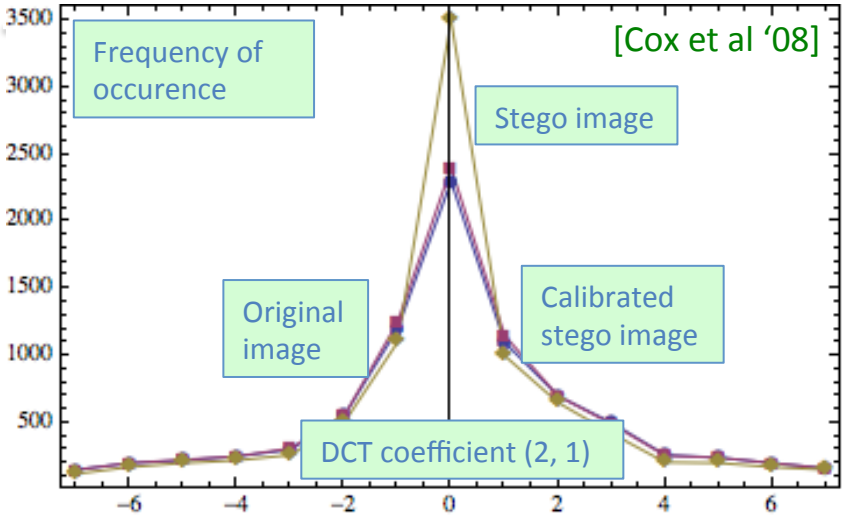
- Compute a number of features (histograms, pairwise correlations, “blockiness”, …)
- Calibrate features: attempt to estimate same features of original work from stego work
- Compute difference vector between the stego work and calibrated features
- Employ some pre-trained classifier on difference vector (SVM, NN, FD, …)

Almost all steganographic methods designed for JPEG embed in DCT coefficients

- **Calibration in DCT domain:** crop image by a few lines, slightly rotate, resize, warp etc.

Stego image

Cut four lines, four rows



- **Calibration in spatial domain:** filter away high-frequency noise (effect of embedding)
- Performance estimated based on test sample of stego and natural images

Machine learning at its best, need many features and lots of data!

The End ☺

So long, and thanks for all the fish (i.e. for your attention)!