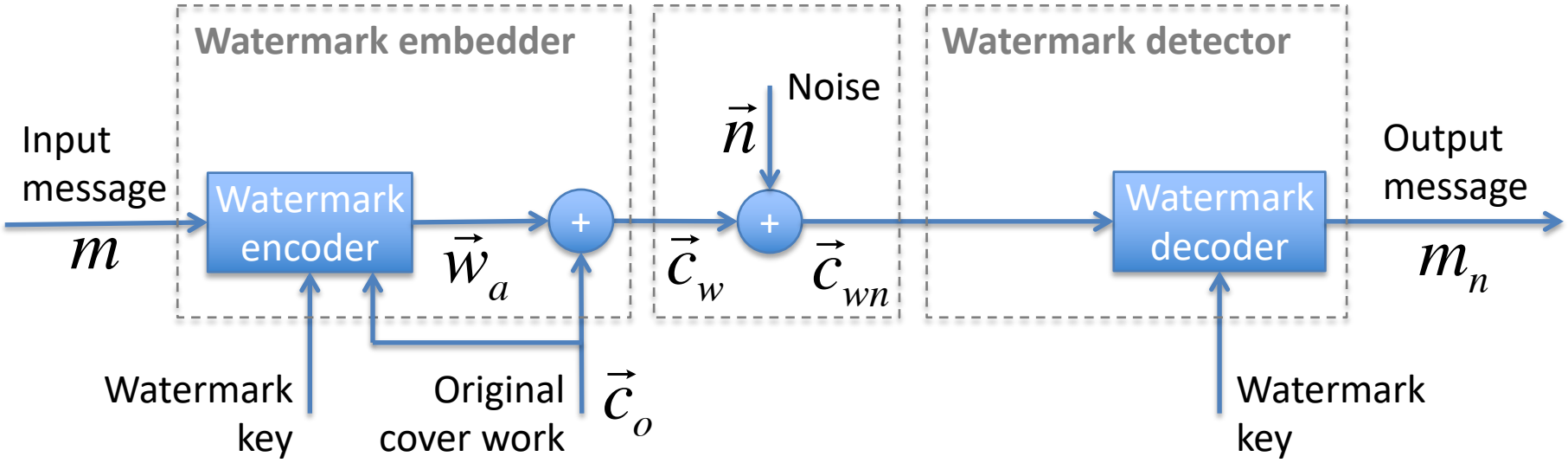
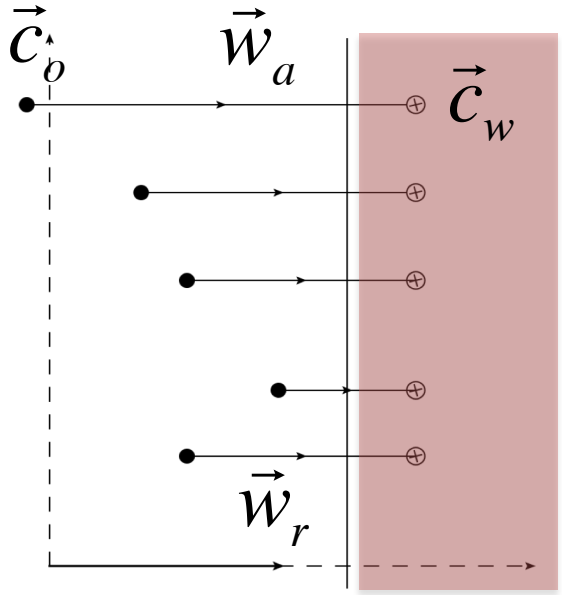
Image Data Compression

Error-correcting watermarking codes

Reminder: a basic watermarking system



Fixed-linear-correlation ("Fixed-LC") embedder:



- Message is mapped to an *added pattern* w_a
- w_a is added to *cover work* w_o to produce a *watermarked work* c_w (i.e. blind embedder: ignoring properties of cover work)
- Further processing adds *noise* n (compression, attacks etc.)
- *Media space*: multi-dimensional space of all works
- *Detection region*: a region in media space, containing works in which watermarks are detected
- *Detection value*: parameter used to identify WM'ed works
- *Marking space*: representation of media space, convenient for WM embedding/detection (transformed via an *extractor*)

Embedding multi-bit messages: direct message coding

Our first LC-based watermarking scheme encodes only 1-bit messages (and WM presence).
Can we extend it to encode many bits at once?

In what follows, we assume blind embedding / decoding

A simple way to encode longer messages:

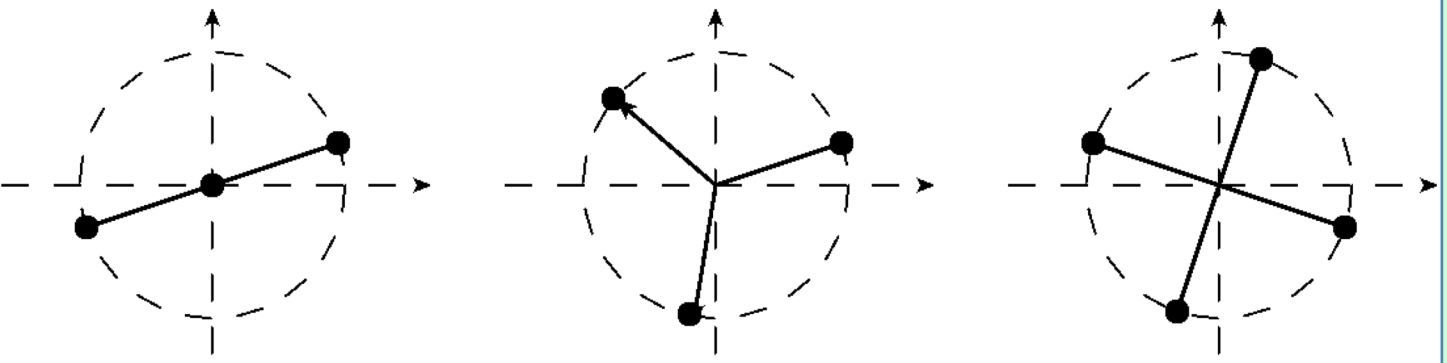
- Define a set M of messages, with the number of messages $N = |M|$.
- Let each message be associated with a separate message mark $W[m_i]$, where $m_i \in M$.

To detect a message:

- Compute the detection value for each of *N* message marks: $z[W[m_i]]$, …, $z[W[m_N]]$.
- Output the message with the highest detection value (**max likelihood detection**):

$$m^* = \arg\max_{m_i} \left(z[W[m_i]] \right)$$

How do we select message marks? Fidelity, FPR, robustness, … + low confusion probability, or *high code separation*: marks should be far apart in the marking space:

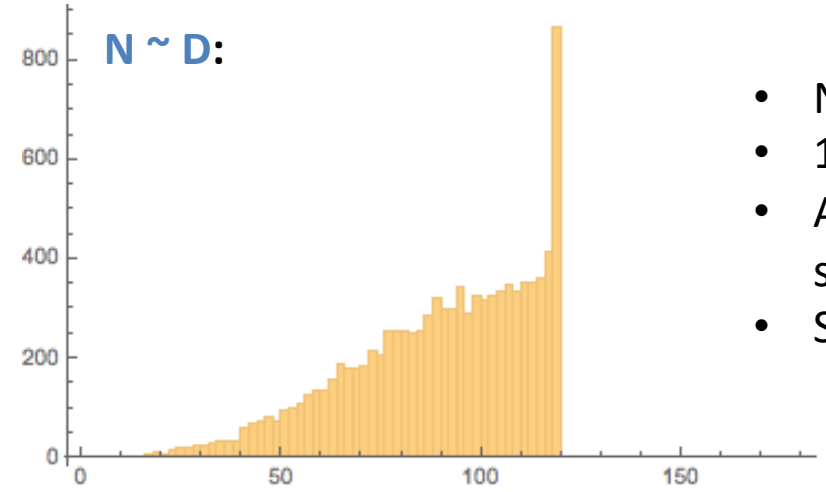


LC detection:
assuming fixed embedding power (vector length), need to place N points on the surface of a D-dimensional sphere so that to maximize the minimal distance between the ***codewords*** (i.e. respective points in space).

Automatic separation of randomly generated codes

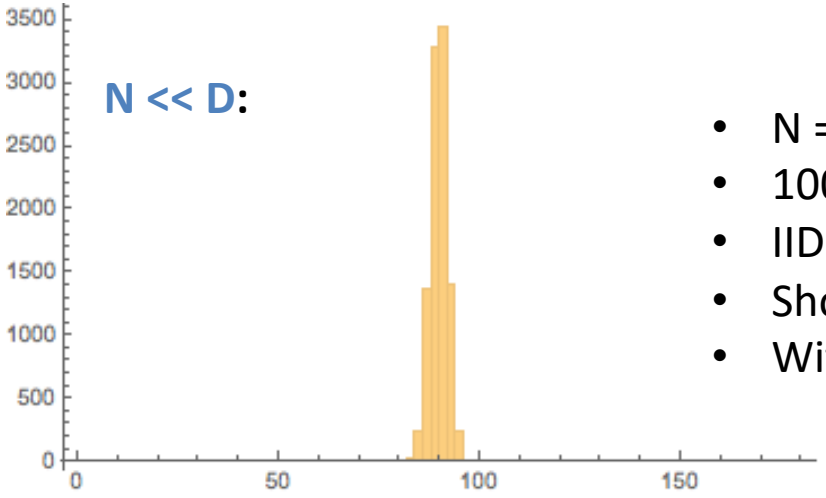
Number of messages is N, dimensionality of vector space is D:

N ~ D:



- N = 3, D = 3:
- 10000 randomly generated triplets of 3D vectors
- All vector components sampled from an IID symmetric Gaussian distribution in 3D space
- Shown: average angle between vectors in each triplet

N << D:



- N = 3, D = 256:
- 10000 randomly generated three-message vectors
- IID symmetric Gaussian distribution in 256D space
- Shown: average angle between vectors in each triplet
- With higher D, peak at 90° grows higher and narrower

Since codes are orthogonal, may embed more than one message in a single work!

Multi-symbol message coding

Direct encoding does not scale well: 16-bit messages require 65536 reference marks (i.e. the detection requires that one computes 65536 detection values!).

Alternative: an alphabet A of size |A| may represent $|A|^L$ distinct messages (each of length L). E.g.: |A| = 4, L = 8, => may encode 65536 messages and detect with 4 * 8 = 32 comparisons.

Goal of embedding: turn a sequence of symbols into a single added mark (i.e. modulation)

Time (space)-division multiplexing, TDM:

- Divide work into disjoint regions, watermark separately
- E.g.: break an image into tiles, divide audio into short samples

Equivalent to CDM: pad each partial mark with zeros to cover the entire work

Frequency division multiplexing, FDM:

- Divide work into disjoint bands in the frequency domain
- May be implemented with a transform done by an extractor

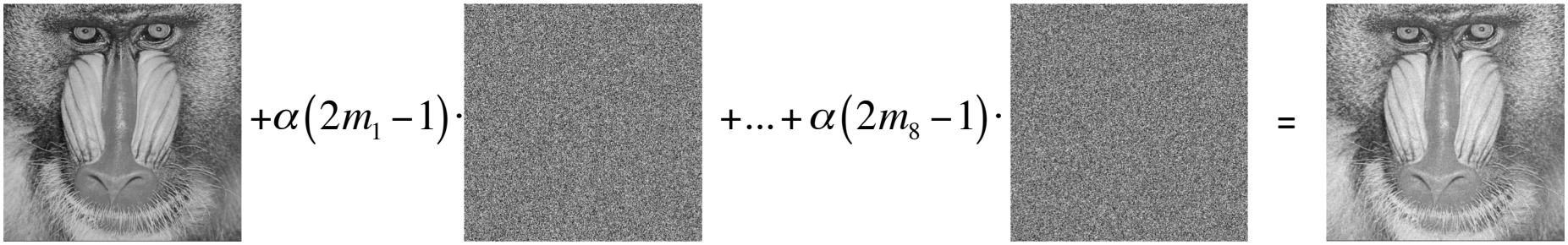
Equivalent to CDM: if a frequency transform is orthogonal, patterns of non-overlapping frequencies have zero correlation both in spatial and in temporal domains

Code division multiplexing, CDM:

- Message: L symbols drawn from the alphabet A
- Define L *|A| reference marks: $W_{AL}[i, s]$ encodes symbol s at position i
- E.g.: |A| = 4, message "3, 1, 4, 4, 2", mark: $w = W_{AL}[1,3] + W_{AL}[2,1] + W_{AL}[3,4] + W_{AL}[4,4] + W_{AL}[5,2]$
- Marks added at different positions should be nearly orthogonal: $i \neq j \Rightarrow W_{AL}[i,a] \cdot W_{AL}[j,b] \approx 0$
- Marks for different symbols at the same position should be well-distinguishable, i.e. ideally have negative correlation: $W_{AL}[i,a] \cdot W_{AL}[i,b] < \lambda < 0$

Example: binary alphabet (|A| = 2), 8-bit messages

Original work: 8 random reference patterns: $w_1, w_2, \ldots, w_8$ WM'ed work:



To ensure better **code separation**, use: $W_{AL}[i, 1] = w_i$, $W_{AL}[i, 0] = -w_i$.

Embedding strength α: chosen so that the embedding power of the final added pattern is 1.0.

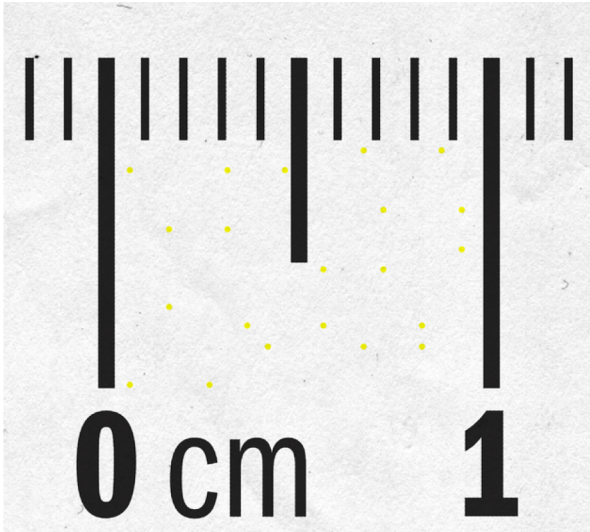
Detector computes correlations with each of the 8 reference marks, and decodes bits based on the sign of the correlation. No distinction is made between marked and non-marked works!

Experimental results [Cox et al '08]:

- Added marks normalized to 2 (strength selection),
- 12000 images watermarked with different 8-bit messages
- 26 of those were incorrectly detected: the scheme is not efficient enough for practical communication!

Any symbol sequence represents a valid message, and the detector cannot distinguish between the correct and incorrect interpretations.

Solution: error-correcting codes



Blind embedding: yellow dots printed by color printers encode their serial numbers

Error-correcting codes (ECC)

Problem with our simple multi-symbol coding:

Let $|A|=4$, $L = 3$. Consider two messages that differ by one symbol:

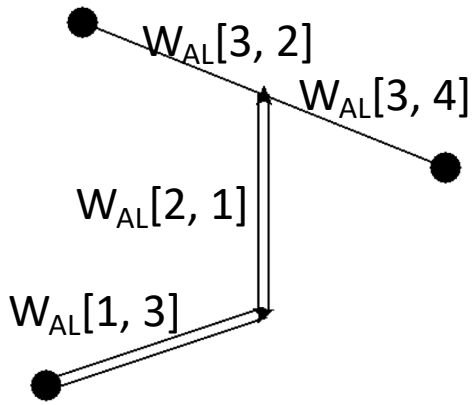
- $W_{312} = W_{AL}[1, 3] + W_{AL}[2, 1] + W_{AL}[3, 2]$
- $W_{314} = W_{AL}[1, 3] + W_{AL}[2, 1] + W_{AL}[3, 4]$

We assume that the marks for different positions are orthogonal:

- $W_{312}W_{314} = W_{AL}[1, 3]^2 + W_{AL}[2, 1]^2 + W_{AL}[3, 2]\ W_{AL}[3, 4]$

For patterns of unit variance, the product is bounded:

- $W_{AL}[3, 2]\ W_{AL}[3, 4] \geq -N$,
- $W_{AL}[i, s]^2 = N$, and therefore $W_{312}W_{314} \geq N$



In general: if two messages differ by **h** symbols, their inner product is bounded from below by **N (L – 2 * h)**

Idea of ECC: the encoder takes a message of **4** bits, and produces **7** bits. If one or two bits are flipped, the code is still decoded correctly (two codewords must differ by at least **3** bits). Minimal inner product without error correction: **N (4 – 2 * 1) = 2N**. With EC: **N (7 – 2 * 3) = N**

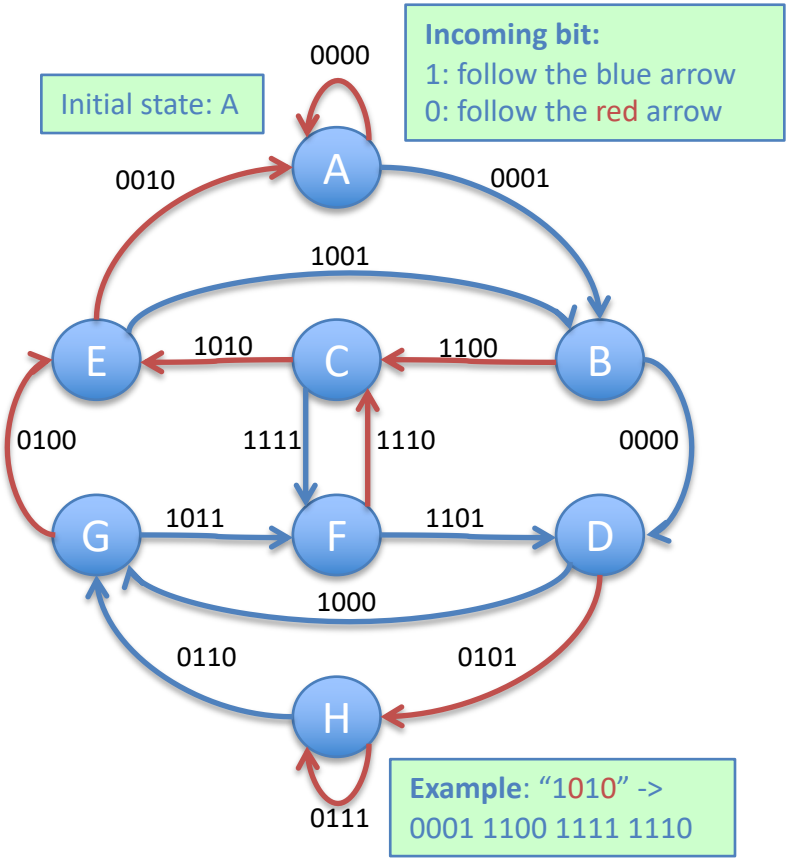
Some well-known error correcting codes:

- Hamming codes: codewords differ by at least 3 bits, robust against random 1-bit errors
- BCH codes, trellis codes: flexible, may correct arbitrary number of (random/burst) errors
- Turbo codes [Berrou et al, '93]: the most effective, but complicated (used in deep space communication); may approach the theoretical boundary (yet another Shannon limit ☺)

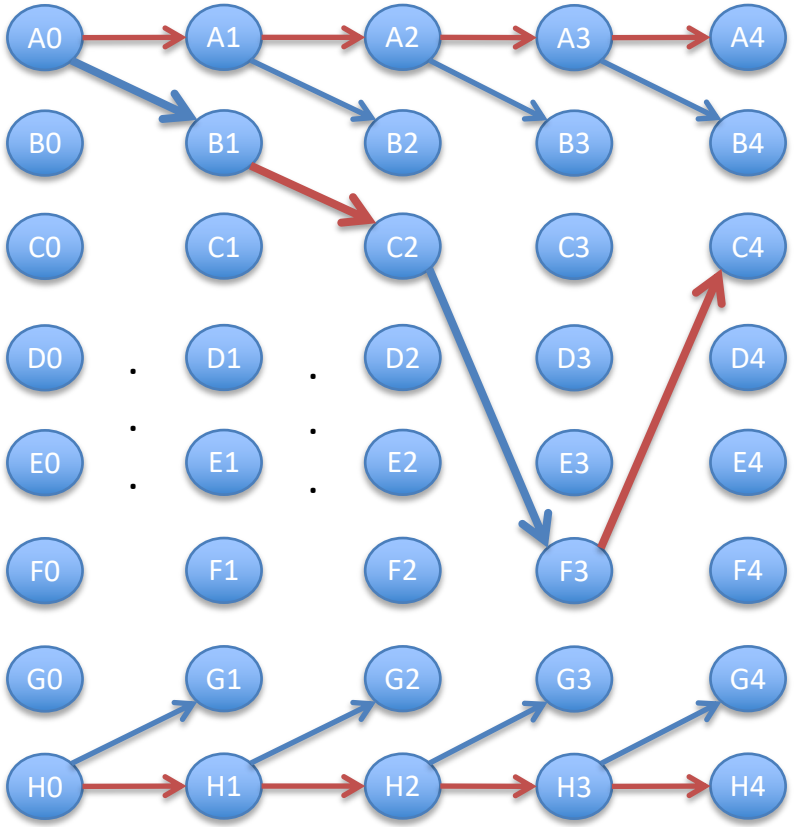
Turbo codes: loopy belief propagation in Bayesian networks…

Trellis (convolutional) codes: a “good-enough” ECC

Encoder: finite state machine
Outputs 4 bits per each input bit



Alternative representation: a *trellis* diagram
(trellis-coded modulation)



Each bit affects the encoding of the subsequent bits (redundant information)

Goal of the decoding: given some coded message, find the most likely path through the trellis (i.e. the one that maximizes the inner product with the message).

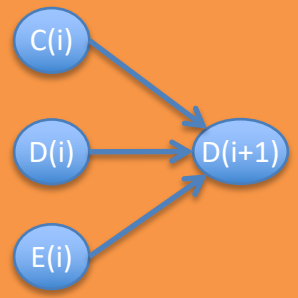
Decoding a trellis-coded message: Viterbi algorithm

The basic postulate: the most likely path passing *through* any node x must include the most likely path *to* x (i.e. greedy decoding)

Hack to ensure that all the decoded paths start from the state **A**:
Initialize **z[A] = 0, z[B] = … = z[H] = -∞**

Algorithm (greedy path search in a graph):

1. Let **v** be the received message vector.
2. Initialize best paths to each state **p[A], p[B], …, p[H]** as zero-length paths.
3. Initialize current partial inner products (likelihoods) **z[A], z[B], …, z[H]** to zero.
4. Start with the trellis column **i = 0**.
5. Compute inner products between **v** and the 16 reference marks, corresponding to all arcs from the column **i** to the column **i + 1**.
6. For each state **x** in the column **i + 1**, compute the current partial inner product leading to **x**, and select the most likely incoming arc.
7. Update **z[…]** and **p[…]**, increment **i**.
8. Repeat from the step 5 until **i** reaches the message length **L**.



$m = \max(z[C] + v.w_{CD}, z[D] + v.w_{DD}, z[E] + v.w_{ED})$

Let the winner be the arc **ED**, then update:

- **p[D] := p[E] + arc(ED)**,
- **z[D] := m**.

Experimental results [Cox et al '08]:

- Added marks normalized to **2**,
- **12000** images watermarked with different **8**-bit messages
- **1** message was incorrectly detected

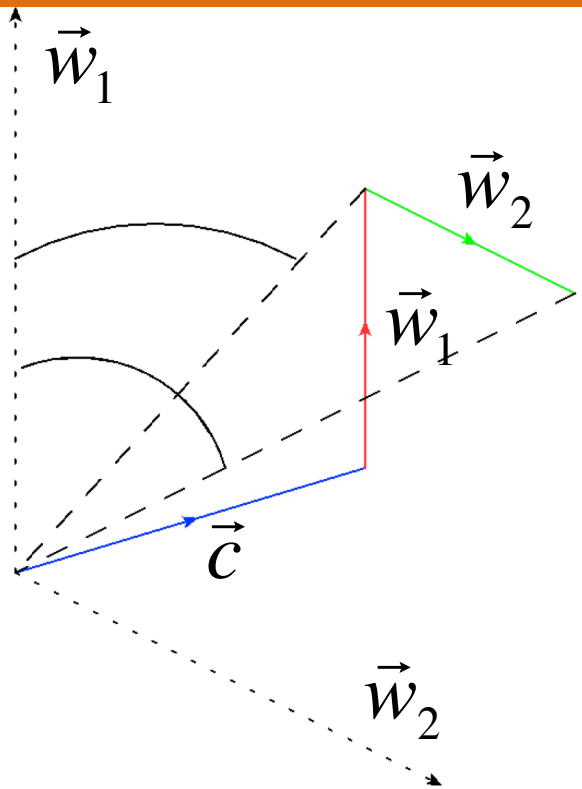
Detecting multi-symbol watermarks

Direct message embedding:

- Simple: check if the highest detection value for a message (out of **|M|** alternatives) is above some pre-defined threshold.

Multi-symbol encoding:

- Declare that only some messages are valid (e.g. append a checksum). False positive probability: **16**-bit message, **9**-bit checksum -> **FPR = 1/512** (only one out of 2^9 messages is valid).
- Detection of individual symbols: **FPR ≈ (|A| P_{fp0})^L**
- However, this applies only when the embedded marks are independent (e.g., with the LC-based detection)
- When using *normalized correlation a*s detection value, new pattern always decreases detection value for other patterns.
- Thus, need some form of *vector quantization* to decide if a vector does contain a watermark.
- **One possible solution:**
  - Decode the most likely message m
  - Re-encode m, obtain the entire expected embedded mark
  - Apply a separate test for its presence (e.g. LC or NC-based)

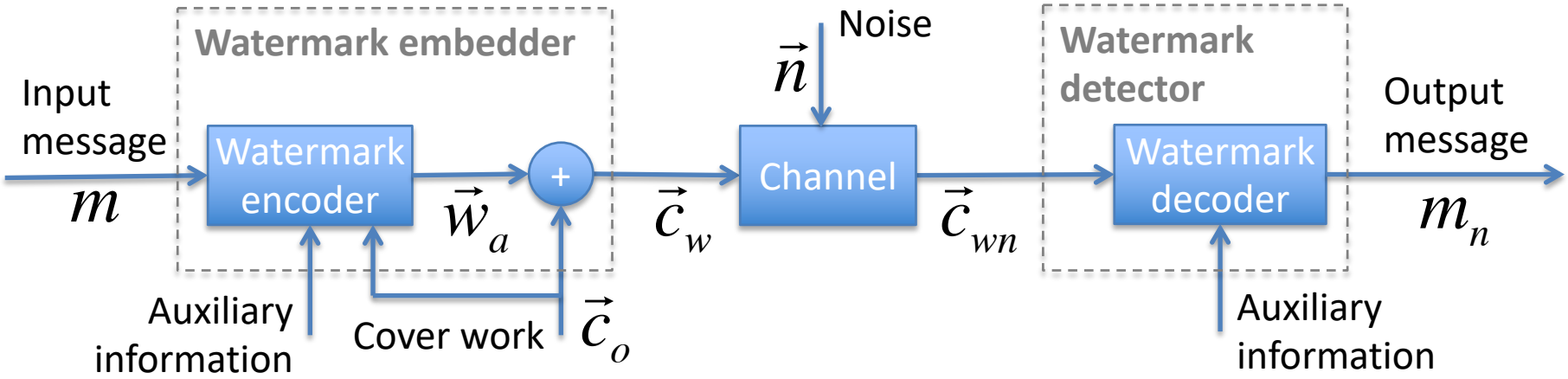


$$\vec{w}_1 \perp \vec{w}_2 \perp \vec{c},$$

$$\alpha(\vec{c} + \vec{w}_1, \vec{w}_1) < \alpha(\vec{c} + \vec{w}_1 + \vec{w}_2, \vec{w}_1)$$

So far, everything was related to blind embedding and detection…
What effects does the cover work lead to?

Watermarking with side information



Quantitative parameters of a watermarking system:

- Constraint on the embedding fidelity: $d(\vec{c}_0, \vec{c}_w) < P_e$
- Constraint on the channel distortion: $d(\vec{c}_w, \vec{c}_{wn}) < P_a$
- Error rate (probability of misdetection): $p(m_n \neq m)$
- Channel rate (# of bits / symbol of cover work): $R = \frac{1}{n}\log_2(|M|)$ [for definiteness, always defined in media space!]
- Assume that the channel behavior is fixed and known to the embedder and the detector

Achievable parameters:

Triple (R, P_e, P_a) is called achievable if $\forall \varepsilon > 0 \exists$ embedding scheme with parameters (R', P_e', P_a') such that

$$R' > R + \varepsilon,$$
$$P_e' < P_e,$$
$$P_a' < P_a,$$
$$p'(m \neq m_n) < \varepsilon.$$

In general, there is no way to know which parameters are achievable. However, for the case of additive white Gaussian noise (AWGN), there exists a known theoretical boundary!

Gaussian watermarking model: notations

- Cover work model: $\vec{s} = (s_1, \ldots, s_n),\ s_i \sim N(0, Q)$

Depends on marking space: e.g. DCT AC coefficients in natural images are roughly Gaussian

- Given message m, embedder produces WM'ed work $\vec{x}$ such that $d(\vec{x}, \vec{s}) < P_e$.
- Channel adds noise: $\vec{n} = (n_1, \ldots, n_n),\ n_i \sim N(0, P_a)$

E.g. quantization error in DCT coefficients is roughly Gaussian

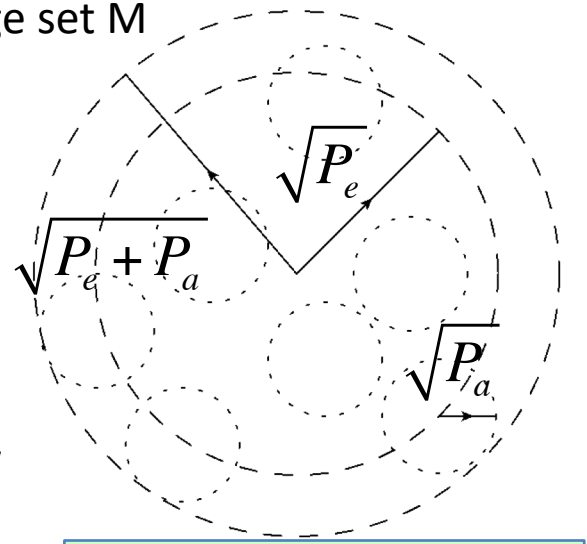
Here we use the MSE distance:

$$d(\vec{x}, \vec{s}) = \frac{1}{n} \sum (x_i - s_i)^2$$

- Degraded signal: $\vec{y} = \vec{x} + \vec{n}$
- Both embedder and detector know Q, P_e, P_a, and the message set M

Optimal watermarking for a single (known) cover work:

- Assume that **Q = 0**, i.e. that the cover work is a zero vector
- **Problem: send a vector message through an AWGN channel**
  - A random Gaussian vector with variance **r^2** is *always* near the surface of the ball of radius **r**, i.e. **$E(x^2) \to r^2$**
  - Volume of an **n**-dimensional ball of radius **r**: **$V(r) = K(n)\, r^n$**
  - Two independent random vectors are always nearly orthogonal, **$E(x.y) \to 0$**
  - Each coding vector (drawn from inside the ball of radius **$P_e^{1/2}$**) is degraded by the channel so that it ends up on a surface of a ball of radius **$P_a^{1/2}$**
  - If the decoding is unambiguous, those smaller balls cannot overlap:



Based on funny geometry in higher-dimensional spaces: n -> ∞

$$|M| \le \frac{V\left(\sqrt{P_e + P_a}\right)}{V\left(\sqrt{P_a}\right)} = \left(\frac{\sqrt{P_e + P_a}}{\sqrt{P_a}}\right)^n, \quad \text{and recall } R = \frac{1}{n} \log_2 (|M|)$$

- **Classical result** [Shannon, '49]: for achievable transmission,

$$R \le \frac{1}{2} \log\left(1 + \frac{P_e}{P_a}\right)$$

Shannon's theorem for correlated Gaussian signals

Re-formulate: for Gaussian variables **X** and **Y**, such that **Y = X + N** (**X** independent of **N**):

$$R \le \frac{1}{2}\log\left(\frac{E[Y^2]}{E[N^2]}\right)$$

Generalize: let **X, Y** be jointly Gaussian variables, such that **ρ = E[XY] ≠ 0**. Then:

- Re-write **Y = Z + λX**, where **Z** is independent of **X** (⇔ uncorrelated with **X**, i.e. **E[ZX] = 0**)
- Therefore, **E[XY] = E[X(Z + λX)] = λE[X²]**, and **λ = E[XY] / E[X²]** (optimal signal scaling).
- The "signal" now is **λX**, and "noise" is **Z**: **E[Z²] = E[Y²] – λ²E[X²] = (E[X²]E[Y²] – E[XY]²) / E[X²]**.

- By Shannon's theorem:

$$R \le \frac{1}{2}\log\frac{E[Y^2]}{E[Z^2]} = \frac{1}{2}\log\left(\frac{E[X^2]E[Y^2]}{E[X^2]E[Y^2]-E[XY]^2}\right) = I(X;Y)$$

Mutual information between **X** and **Y**

Optimal codebook now is constructed over the re-scaled distribution **λX**

Notes:

- Mutual information is a central concept in the information theory!
- Geometrical interpretation: **I(X; Y) = - log(|sin α(X, Y)|)**, **α** being the "angle" between **X** and **Y**
- By construction, Shannon limit is robust against unknown gains in the transmission channel

Remaining questions:

1. How does the addition of a cover work change those bounds?
2. How does one optimally watermark Gaussian signals? (dirty-paper codes + distortion compensation…)