Lossy Compression of Data

Lossy Compression

- In order to compress data without loss, we cannot go below the Shannon threshold (the entropy).
- What happens if we try to use a encoding that does not respect this threshold?
- Let's try with the message JE PARS A PARIS and the following encoding:

letter	A	P	R	S	J	E	I
code word	1	0	11	10	01	00	111

- With such a code, we use only a total of 18 bits instead of 33. The binary representation of the message is 01 00 ..
- However, the recipient cannot know if the message starts with JE, JPP, PAE, PAPP or PSP,…

Lossy Compression

- However, sometimes we have to accept losing information, e.g.,
  - when we represent a **real number with a fixed number** of bits (the information included in a real number can be "infinite”),
  - when we **sample** a signal whose greatest **frequency is "infinite”,**
  - when you **download** from a website all our vacation **photos** (a few gigabytes for a hundred photos with the highest resolution).

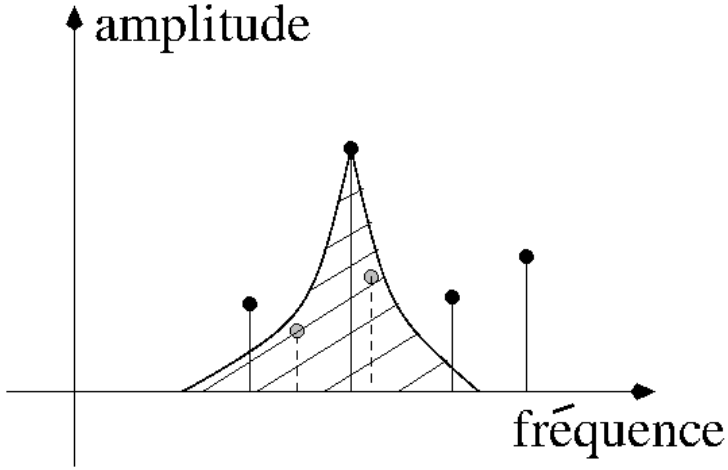
- How can we proceed in these cases?
- First filter the high frequencies!

Lossy Compression: Sound

- Recall, the sound recorded on a CD is sampled with 44.1 kHz (a bit more than 2x20kHz, the max. frequency a human ear can perceive)
- Each sample is encoded with 2 × 16 bits (2 channels for stereo).
- So, for one second, you need 44.1k × 16b × 2 ≈ 1'411kb (about **176kB**). A song (with 3.5min) needs about 3.5 × 60 × 176kB = 36'960kB (about 37MB).
- The MP3 format encodes 1s in only 128 kbits (or **16kB**). So, we need about 3.3MB for 3.5min, which corresponds to a 90% reduction in the size of a file! (Without noticeable audible distortion of sound)
- What is the "trick"?

Lossy Compression: Sound

- Key: **perceptual coding** aka **psychoacoustic modeling**
- When a sinusoid with a certain frequency is present with great amplitude in a sound, it masks the other sinusoids of near frequency and of smaller amplitude.



- Perceptual coding works by assigning fewer bits of data to the masked elements of a recording than to the 'relevant' ones.
- The same is true in the time domain (a tone with a high amplitude will mask another one close in time but with much lower amplitude)

What is lost with MP3?

- Developers of MP3 decided with information to drop. It took them years to find the right setup so that one cannot hear a difference.
- Example:
  - https://www.youtube.com/watch?v=oYaTZPicwmA&start_radio=1&list=RDoYaTZPicwmA
  - Minutes 9:23-10:30 playing original signal and inverted signal and both at the same time (no sound – they cancel each other) + explanation why there is no sound above
  - Minutes 11:42-11:55 playing inverted and MP3 one after the other
  - Minutes 12:29-12:50 playing inverted and MP3 at the same time (we hear the difference between the two files)

Lossy Compression: Images

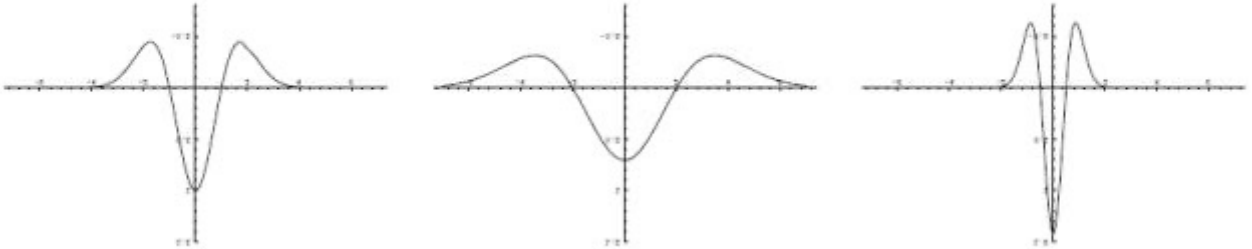
- In the same way that the human ear cannot hear sounds beyond ~20 kHz, the human eye has a resolving power of about one minute of arc = 1/60 ~ 0.017 degrees which means that it cannot distinguish:
  - craters on the moon with a diameter less than 100km
  - objects that smaller than 1mm and 3m away
  - pixels smaller than 0.2 × 0.2mm on a computer screen (from a distance of 50cm).

- How can we filter the high (spatial) frequencies in an image?
- One simple way: average the colors in the surrounding areas
- This is a moving average filter!

Lossy Compression: Images

Lossy Compression: Images

- We again need to make a compromise:
  - The larger the pixels are, the less memory we will need to store the image but the more deformed the original signal will be.
  - This effect is called **distortion**.
- There are, of course, much more sophisticated algorithms to compress an image with loss.
  - JPEG (Joint Photographic Experts Group) format:
    - analyzes the spatial frequencies present in the image,
    - keeps only the lowest frequencies,
    - uses a lossless compression algorithm on the result.
  - format JPEG 2000: the same idea but with wavelets (ondelette)



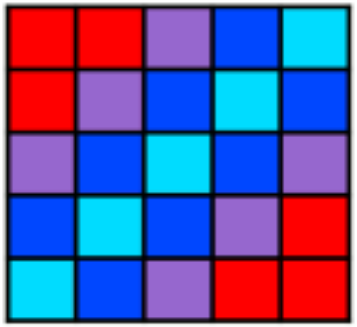
(used like frequencies, Wavelets domain)

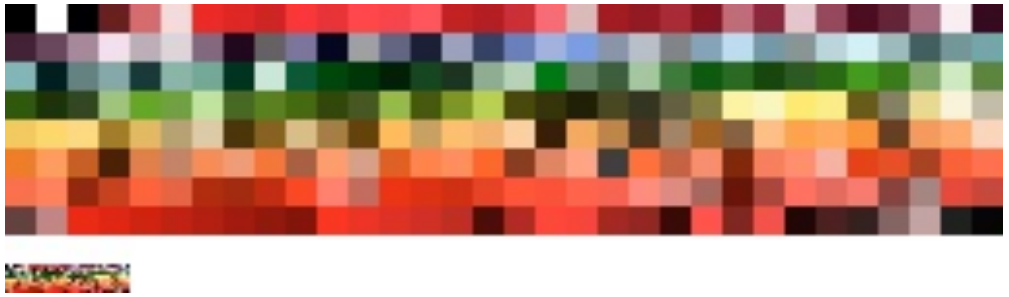
Indexed Colors

- Another coding an image: indexed colors GIF
- Build a palette of 4, 16 or 256 colors
  - each pixel only stores the index to the palette.
  - each color of the palette is defined with 4 bytes

- For images with:
  - a small number of well-defined colors, e.g., technical drawing, comics, manga, comics (lossless compression)
  - few nuances

0	0	1	2	3
0	1	2	3	2
1	2	3	2	1
2	3	2	1	0
3	2	1	0	0

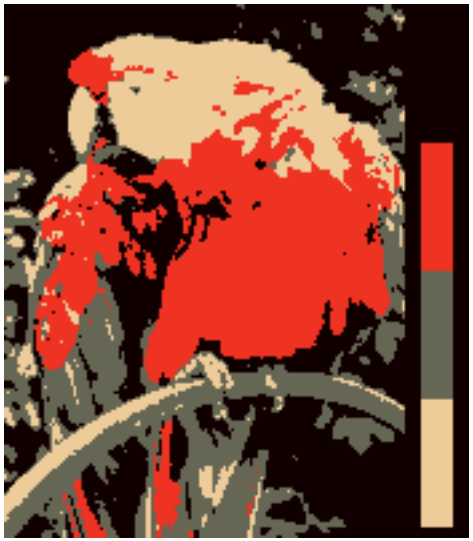




[wikipedia indexed colors]

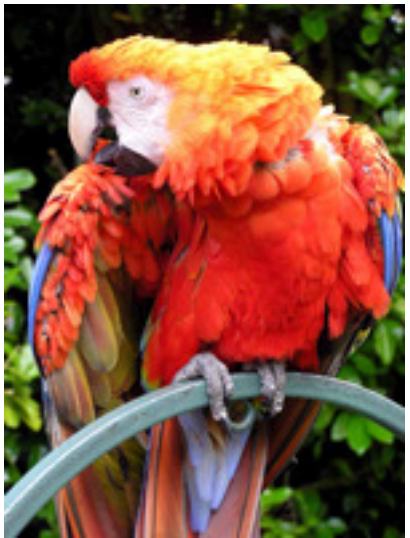
Example

4-color



16-color

256-color



Conclusion

- We discussed how to compress data with and without losing information
- We learned that entropy is a lower bound for lossless compression.
- We use the Shannon-Fano and the Huffman algorithm to create prefix-free code to compress a message
- Finally, we saw techniques to compress data with losing information. Examples are mp3 for sound, jpg for images, and indexed images (gif)

Conclusion

- In this module, we saw:
  - An overview of the theory of **signal sampling**.
  - An overview of the theory of **data compression**.
- These two theories are two facets of one and the same subject with two central results of the same nature!
- Key points of this module:
  - It is always possible to reduce the number of bits to store a signal or data.
  - However, there are thresholds one must respect in order to not lose information.
  - The measures of these thresholds are the **bandwidth** and the **entropy,** respectively.

Questions?

Next Topics

- After the exam, we start with algorithms and complexity
  - Iterative algorithms (for loops)
  - The complexity of iterative algorithms
  - Recursive algorithms (functions that call themselves)
  - The complexity of recursive algorithms
  - Computability, what can and what cannot be solved with algorithms
- Then we will talk about hardware:
  - CPU (central processing unit)
  - How to convert c code into code that a CPU can understand
  - Memory and Storage