Image Data Compression

Class Introduction

About the Instructor (me)

- 2001: Novosibirsk State University (Russia), **BSc** in Physics/Informatics
- Studied processes at electron-positron collider experiments in Russia, Germany.

- 2003: Novosibirsk State University, **MSc** in Experimental Particle Physics
- Collaboration member, analyzed data at BELLE collider experiment in Tsukuba, Japan.

- 2008: University of Alberta (Canada), **PhD** in Theoretical Particle Physics
- Performed large-scale computer-aided analytic calculations in quantum field theory.

- 2008-2011: KIT, Institute for Theoretical Particle Physics (TTP), Postdoctoral Researcher
- Computed higher-order quantum corrections to Higgs boson production at LHC.

- Since 2011: Fraunhofer IOSB (Karlsruhe), Senior Scientist at MRD department
- 2011-2017: KIT, Institute for Anthropomatics, IES Laboratory, Scientific Advisor
- Feel free to explore new HiWi and research positions at www.ies.anthropomatik.kit.edu!

Primary research areas: Optical Metrology, Ill-posed Inverse Problems, 3D Reconstruction, Probabilistic Graphical Models, Inverse Problems, Sensor Data Fusion, Tracking.

- Contact: alexey.pak@iosb.fraunhofer.de, Fraunhoferstrasse 1, Tel. **0721-6091-667**

Useful books to read

- T. Strutz, **Bilddatenkompression**, Vieweg+Teubner, 2005
- (Theoretical and technical details of many formats, such as JPEG, MPEG, H.264)

- J. Beyerer, F. Puente León, C. Frese, **Automatische Sichtprüfung**, Springer 2012
- (Foundations of optics, image acquisition and exploitation)

- I. Cox, M. Miller, J. Bloom, J. Fridrich, T. Kalker, **Digital Watermarking and Steganography**, Morgan Kaufmann, 2008
- (Detailed overview of both subjects and relevant theoretical results)

- T. Cover, J. Thomas, **Elements of Information Theory**, Wiley-Interscience, 2006
- (General information theory and communication theory)

- R. Gonzalez, R. Woods, **Digital Image Processing**, Prentice Hall, New Jersey, 2008
- (Digital image processing, including compression, enhancement, restoration etc.)

Why do we need to compress data?

We now have better CPUs and storage devices, but still need data compression:

- New sensors, faster digital transmission and processing allow better signal quality
- Data sizes grow on par (or faster) with storage or bandwidth capacity
- New applications: archiving, digital TV, video surveillance, telemedicine, search...
- Internally, modern CPU/GPU/supercomputer architectures are bandwidth limited; data compression is used even to send data to the next computing unit!
- In fact, efficient data compression becomes ever more crucial to modern applications!

Specific features of image and video data (as opposed to "common" data):

- 2D (or 3D) multi-channel signals (or higher-dimensional, e.g. with 3D-information)
- Pixel values are spatially and/or temporally correlated
- No simple and clear quantitative measure of "image / video quality"
- "Typical" natural images <u>extremely</u> non-random, very complex distribution

Example: an hour-long 4K movie
(3 bytes / pixel) * (3840 x 2160 pixels / frame) * (24 frames / sec) * (1h = 3600 sec) = 2.14×10^{12} bytes.

Compare to:
1-hr movie, MPEG compression: ~44 Gb = 4.7×10^{10} bytes!

Redundancy: how much information does an image contain?

A “natural” image: short and long-range correlations between pixels, un-even distribution of pixel values, some areas can be “predicted” from its neighborhood (e.g. sky color).

An un-natural image: pixel values are random, no strong correlations at any scale

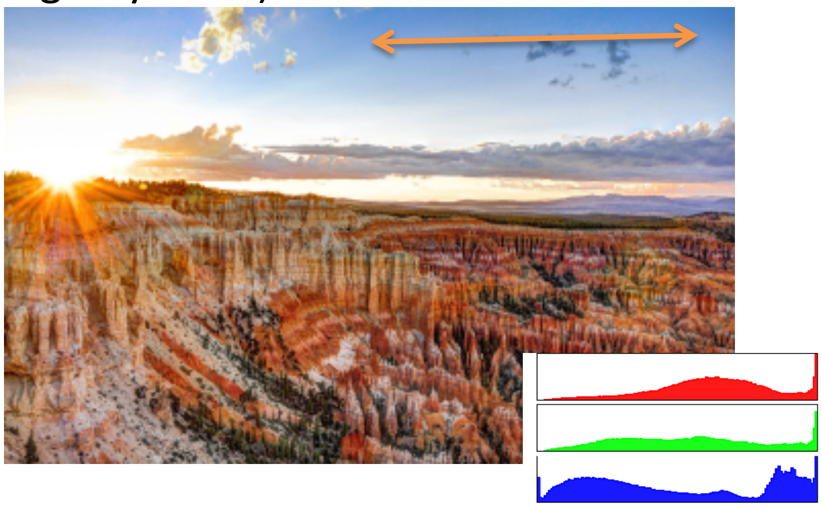


Image histogram

vs

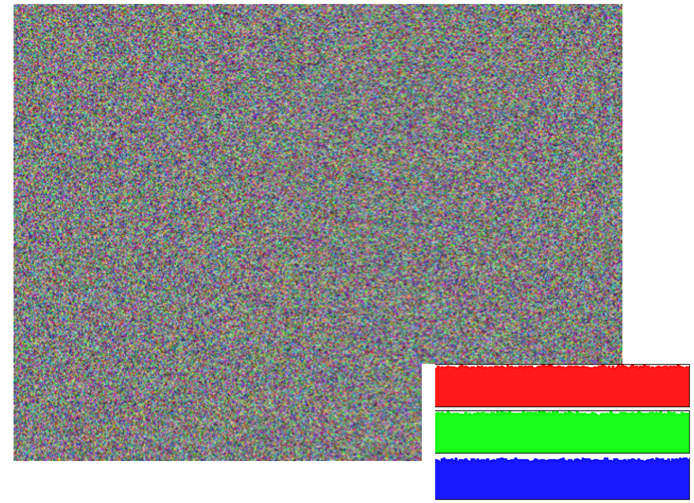


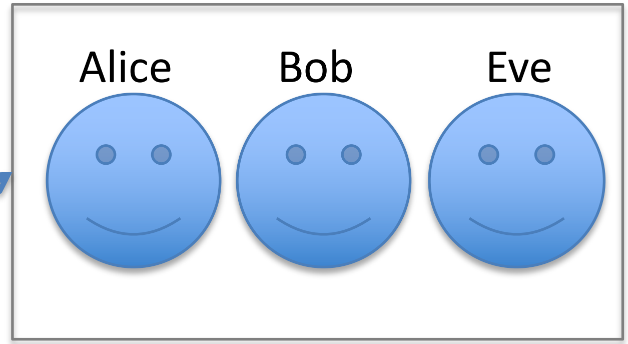
Image histogram

When pixel values are not independent, each value contains redundancy, which can be removed in order to compress data without losses.

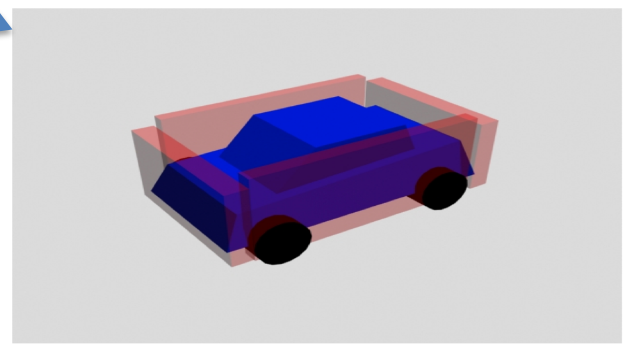
Image above: (2880 x 1800 pixels) * (3 bytes / pixel) = 15.5×10^6 bytes; however, can be stored without losses in a 9.9 Mb PNG file.

Relevance and irrelevance: useful information in an image

Given an image, we can exploit it differently depending on the given task:



Information extracted from image



1) Task (Computer Vision):
- **Identify people**

Useful information:
Number of persons, their names, poses, etc.

2) Task (Computer Vision):
- **Identify vehicles**

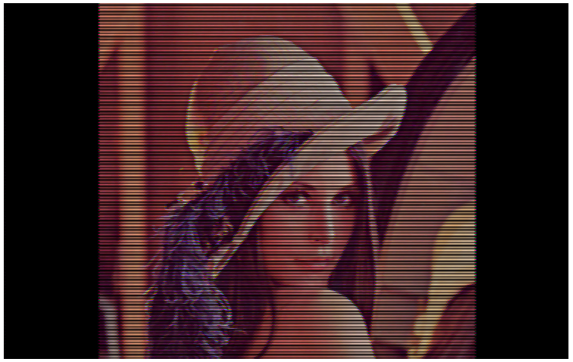
Useful information:
Car size, position, orientation

3) Task (Human Observer):
- **Perceive image aesthetics**

Useful information: ?
Perhaps everything perceivable by humans…

Useful information is task-dependent; the remaining information in the image is irrelevant. Lossy compression removes irrelevance from image preserving its usefulness for given task (i.e. task is tolerant to the introduced changes).

Lossy image compression and irrelevance

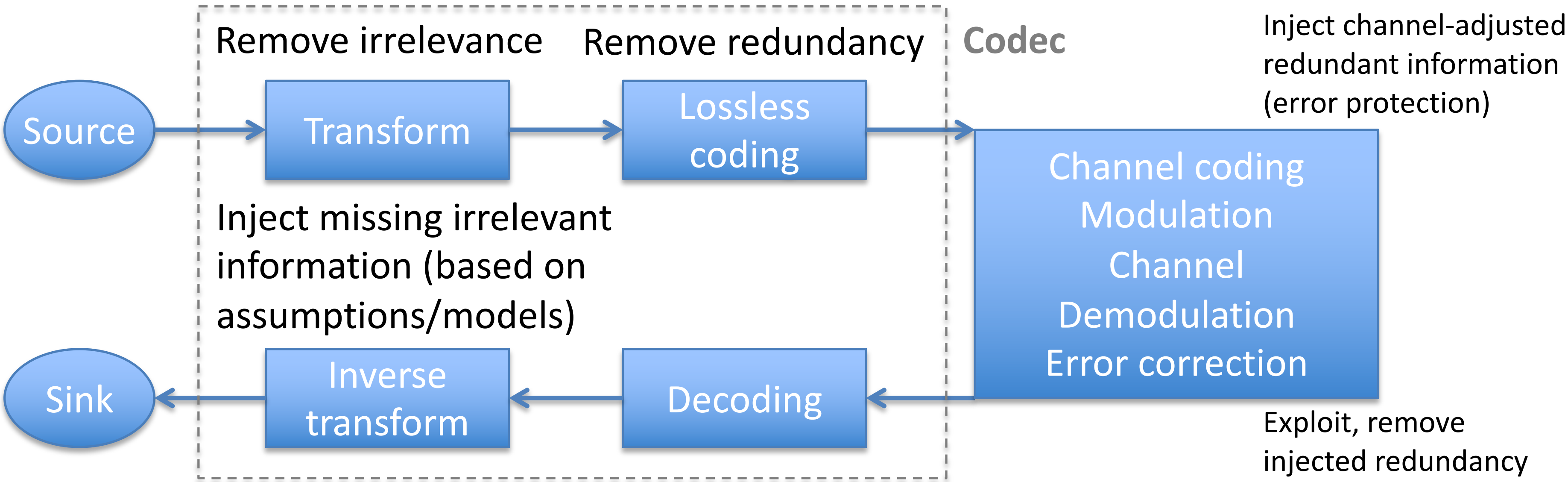


NTSC color TV standard (1953-2009):
- Artifacts should be "minimally noticeable" by humans

Chosen discretization and sampling:
- 525 scan lines, 29.97 frames/sec (interlaced)

Transform:
- Luminance-chrominance color encoding: YCrCb (not RGB!)
- Luminance signal: ~6 MHz band
- Truncate chrominance signal: ~1.5 MHz, less resolution

- **Lossy compression**: irreversible removal of irrelevant information
- Compare to AI: an algorithm has to understand what a human brain can or cannot perceive.



What is this course about?

Topics discussed in this class:

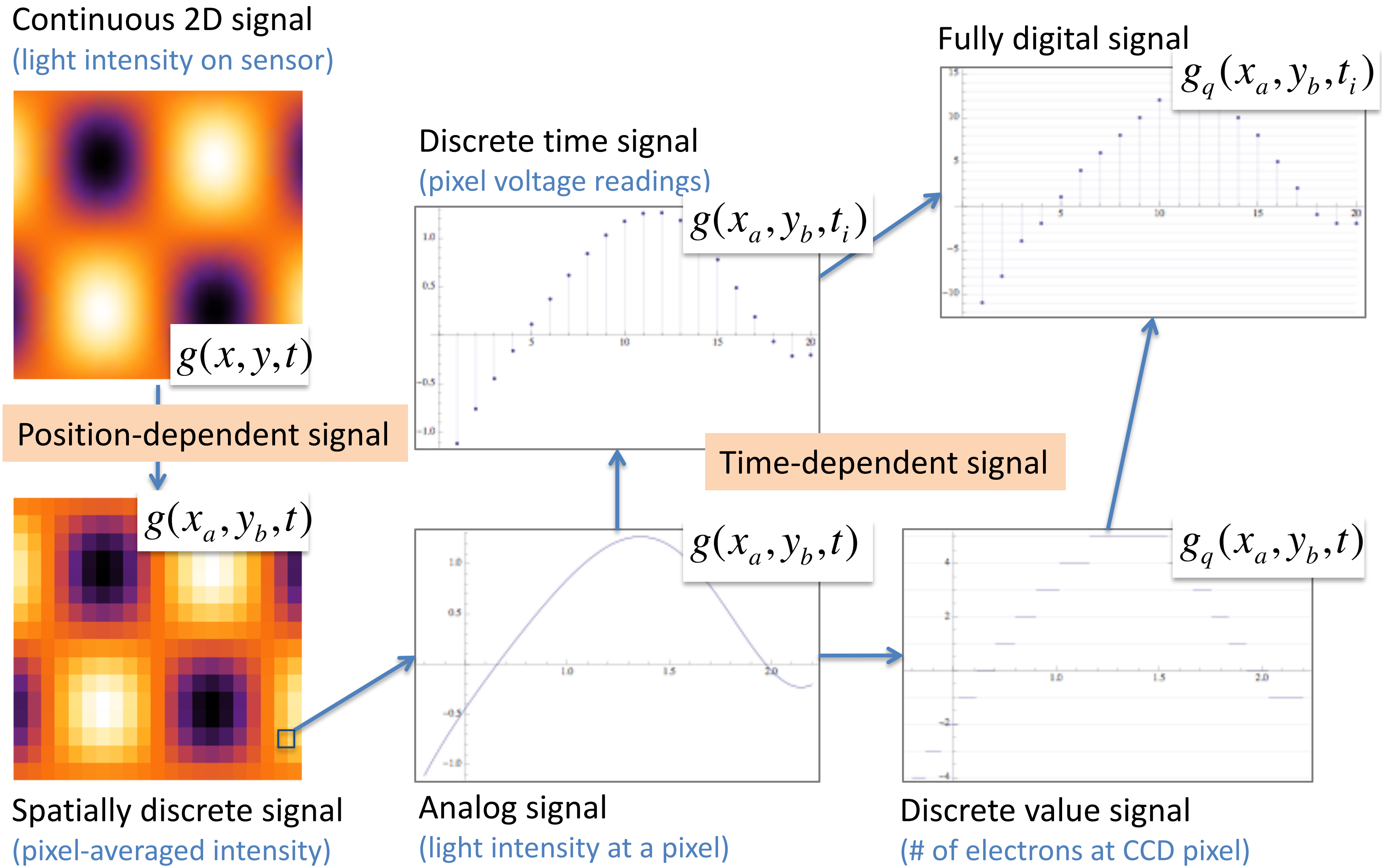
- Origins and nature of information represented by images
- Statistical properties of “natural” images
- Human perception of visual information
- Different forms of image data representation
- Qualitative and quantitative metrics of information in images
- Changes in quality and quantity of information during common (digital) transformations and manipulations with images
- Non-conventional ways to exploit the information contained in images

Any questions so far?

Image Data Compression

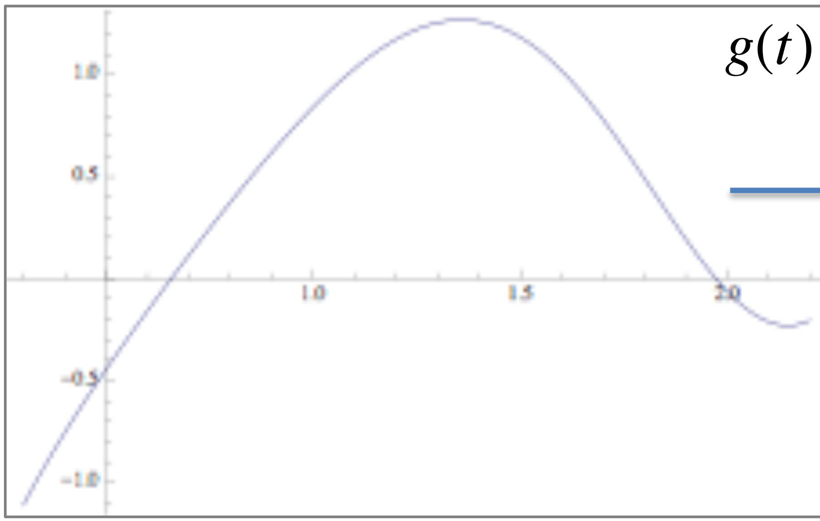
Data Reduction Techniques

Unavoidable data reduction: discretization/digitization

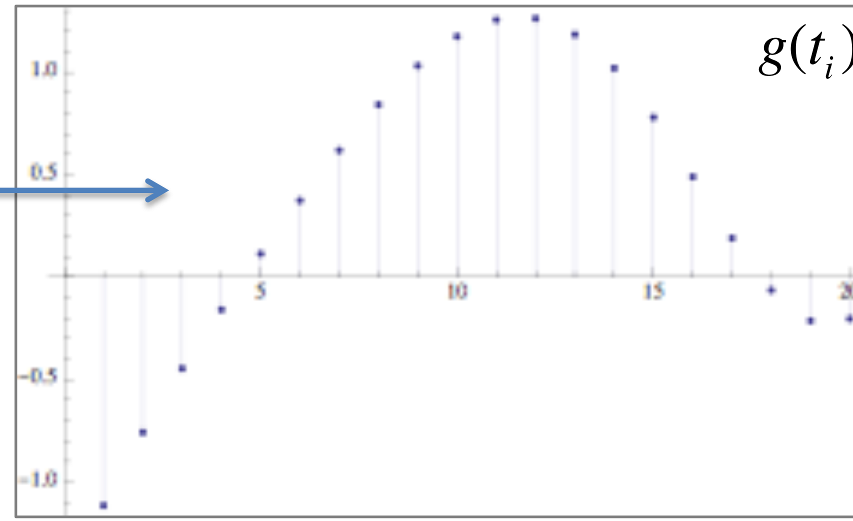


Sampling a continuous-time signal

Continuous-time continuous-value signal
(i.e. sampled at very high frequency)



Discrete-time continuous-value signal
(re-sampled at given rate)

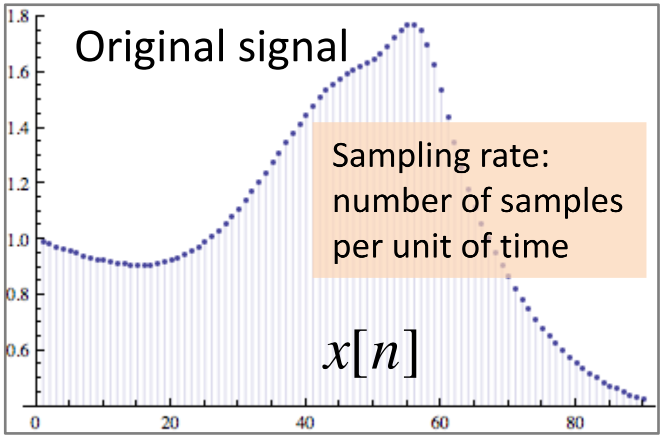


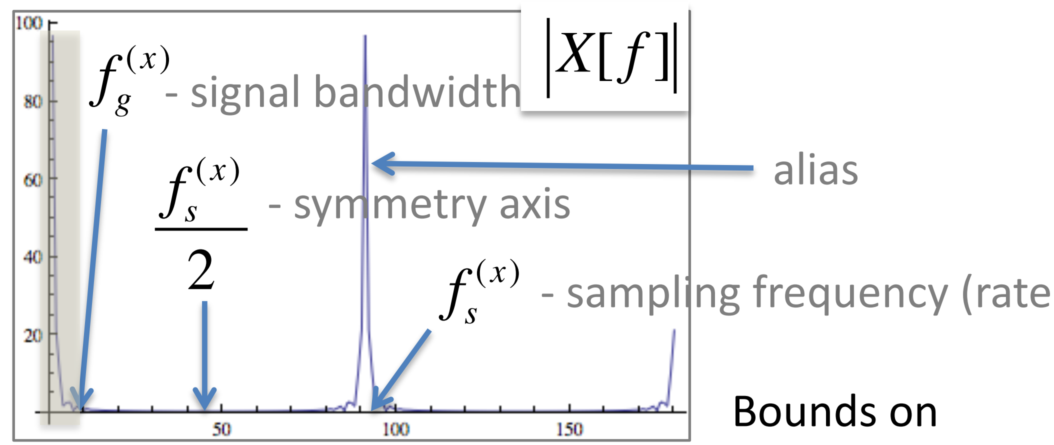
Simple data reduction: sub-sampling

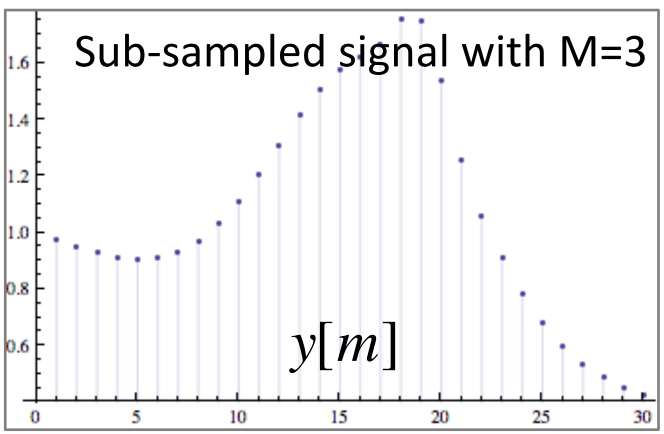
$x[n]$ → $y[m] = x[m \cdot M]$

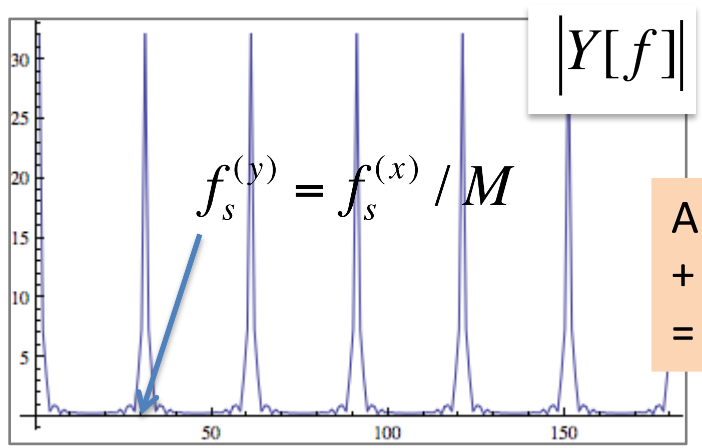
DFT: $$X[f] = \sum_{n=1}^{N} x[n] \cdot e^{-2\pi i (n-1)(f-1)/N}$$

- Discrete Fourier Transform







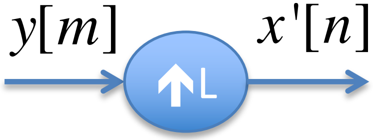


Bounds on sub-sampling:

$$f_g^{(x)} < \frac{f_s^{(x)}}{M}$$

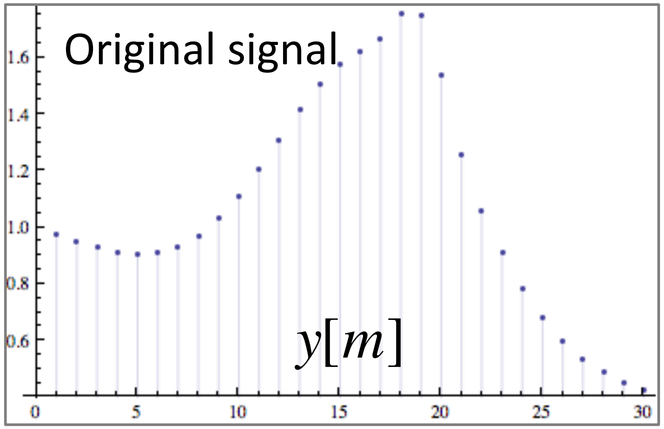
Anti-aliasing filtering
+ Sub-sampling
= **Decimation**

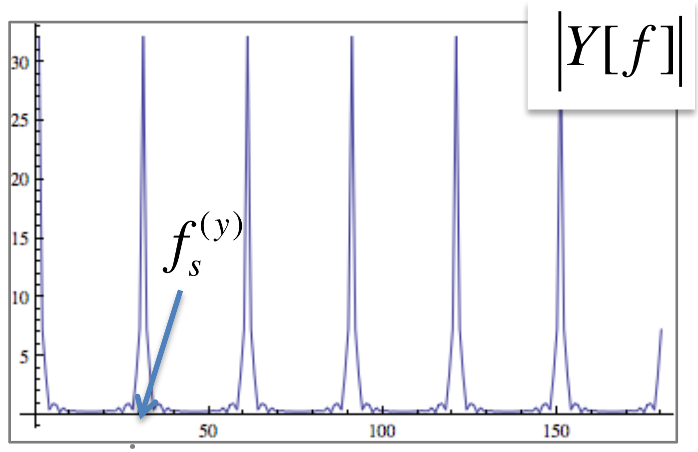
Simple data reduction: up-sampling



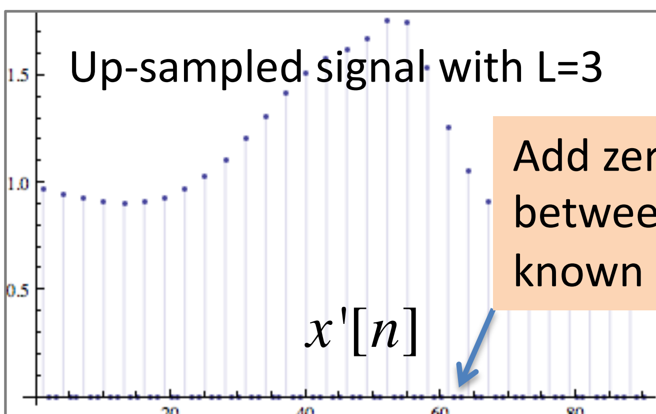
$$x'[n] = \begin{cases} y[n/L], n = m \cdot L \\ 0, otherwise \end{cases}$$

"Given a signal sampled at some low frequency, recover intermediate values (i.e. samples at a higher frequency)."

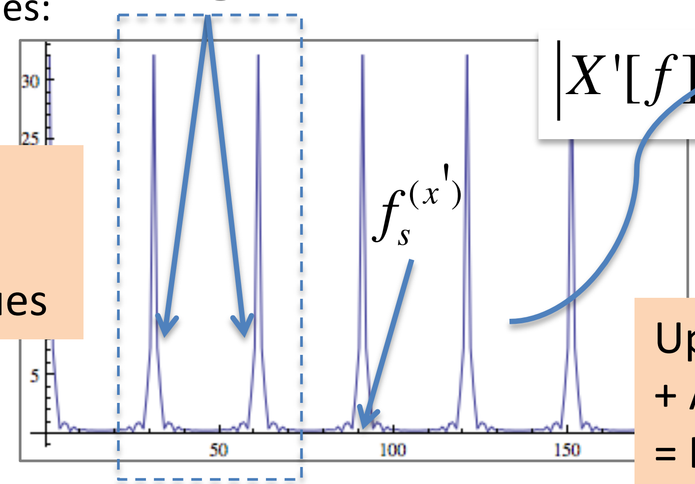


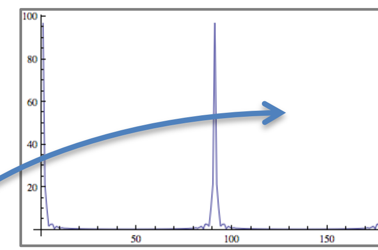


The simplest way to find intermediate values:



Add zeros between known values

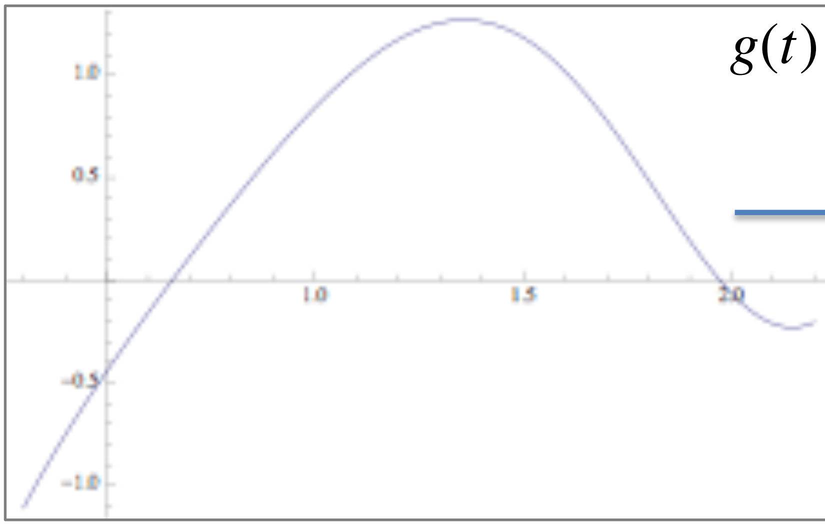




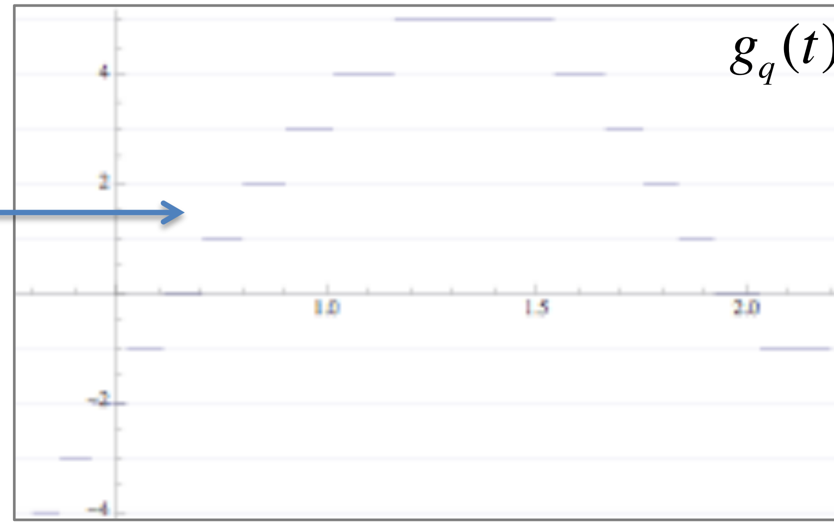
Up-sampling
+ Anti-imaging filtering
= **Interpolation**

Quantizing a continuous-value signal

Continuous-value signal
(i.e. a large number of possible signal values)



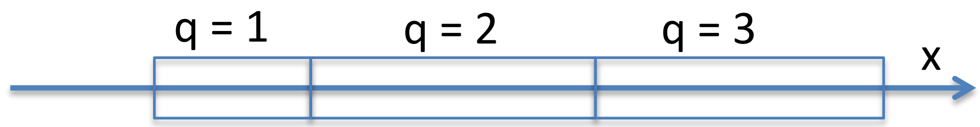
Discrete-value signal
(limited number of chosen signal values)



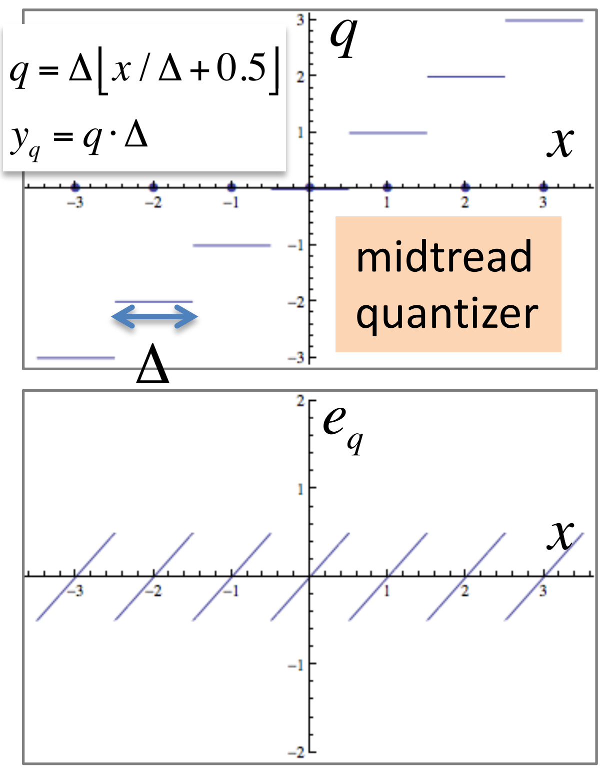
Simple data reduction: scalar quantization

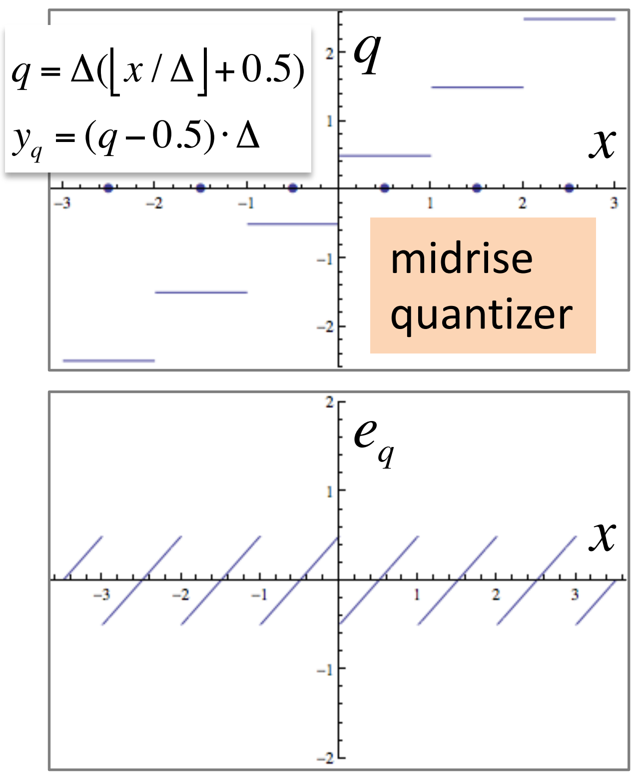
Mapping from continuous 1D real-valued signal value to quantized (discrete) values:

- Quantization: $x \to q$
- Reconstruction: $q \to y_q$
- Quantization error: $e_q = x - y_q$



Uniform quantization with interval size Δ – two important types:





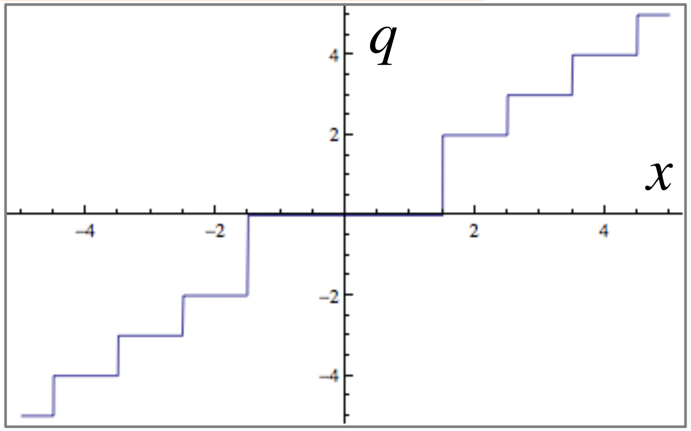
Assuming uniformly distributed signal, power of error in both cases is:

$$p(e_q) = \begin{cases} 1/\Delta, |e_q| \le \Delta/2 \\ 0, otherwise \end{cases}$$

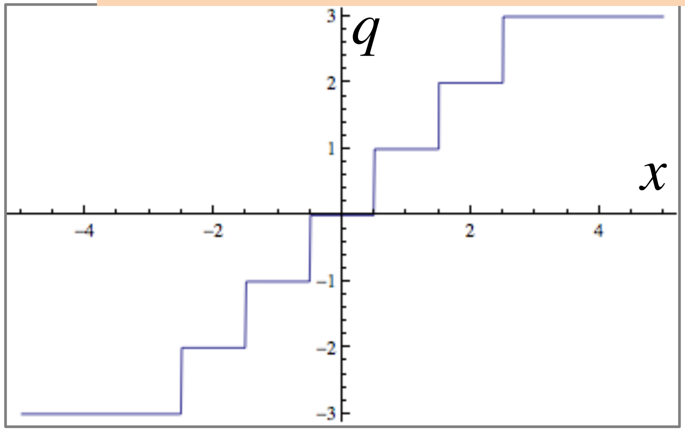
$$\langle e_q \rangle = 0$$

$$\langle e_q^2 \rangle = \int e_q^2 \cdot p(e_q) \cdot de_q = \Delta^2/12$$

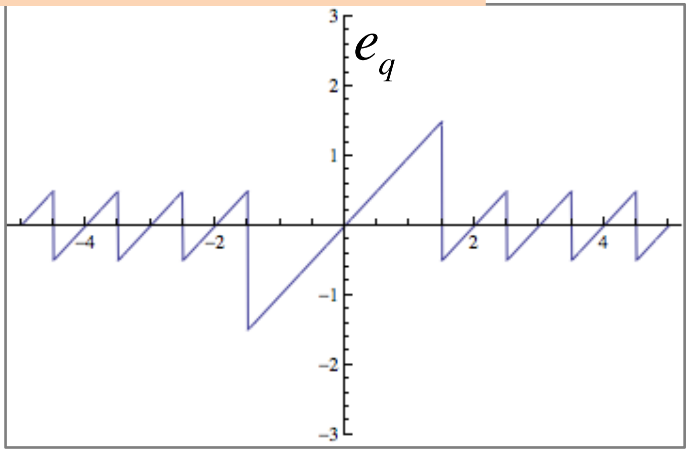
quantizer with a dead zone:

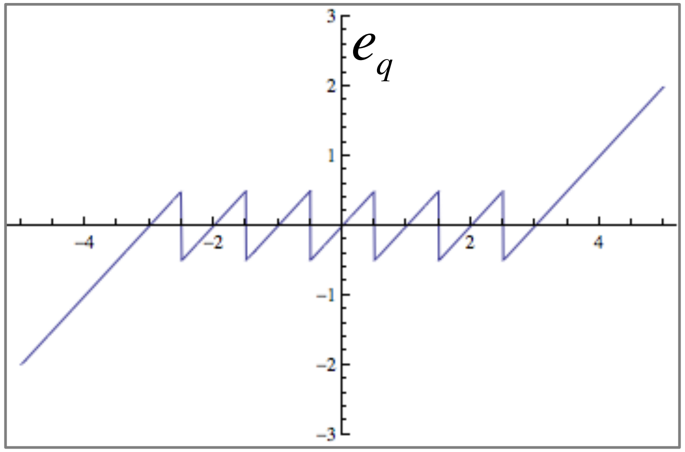


quantizer with limited amplitude:



quantization error:

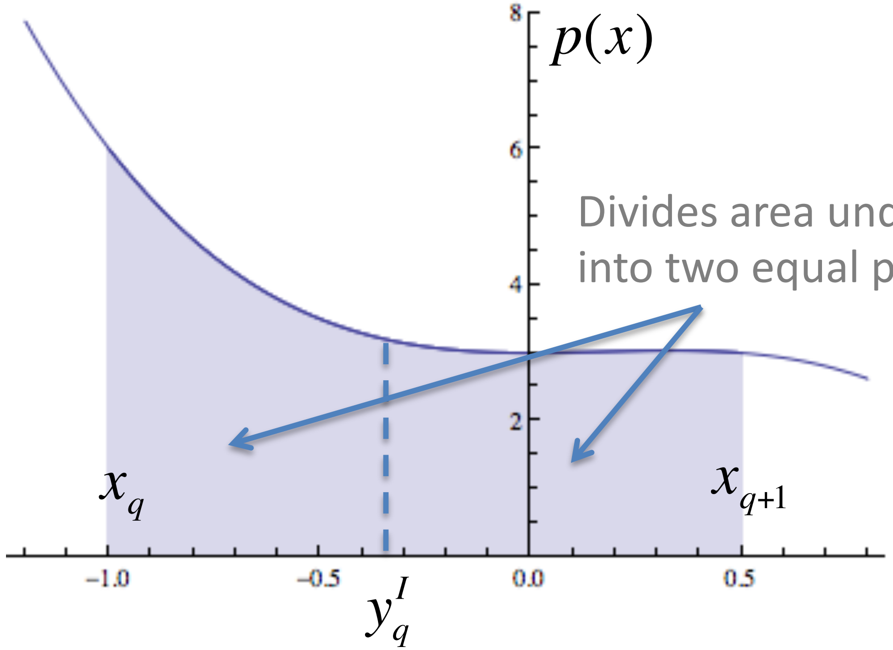




Non-uniform quantization: PDF-optimized

Choice of reconstruction values:

- Consider interval: $I = [x_q, x_{q+1})$
- Signal probability distribution function (PDF): $p(x)$
- To find: reconstruction value y_q
- Objective: minimize average error, $\langle e_q \rangle_I \to 0$



Result: all intervals become equally likely

- Solution:

$$y_q^I = \langle x \rangle_I = \frac{\int_{x_q}^{x_{q+1}} x \cdot p(x) \cdot dx}{\int_{x_q}^{x_{q+1}} p(x) \cdot dx}$$

Choice of interval boundaries:

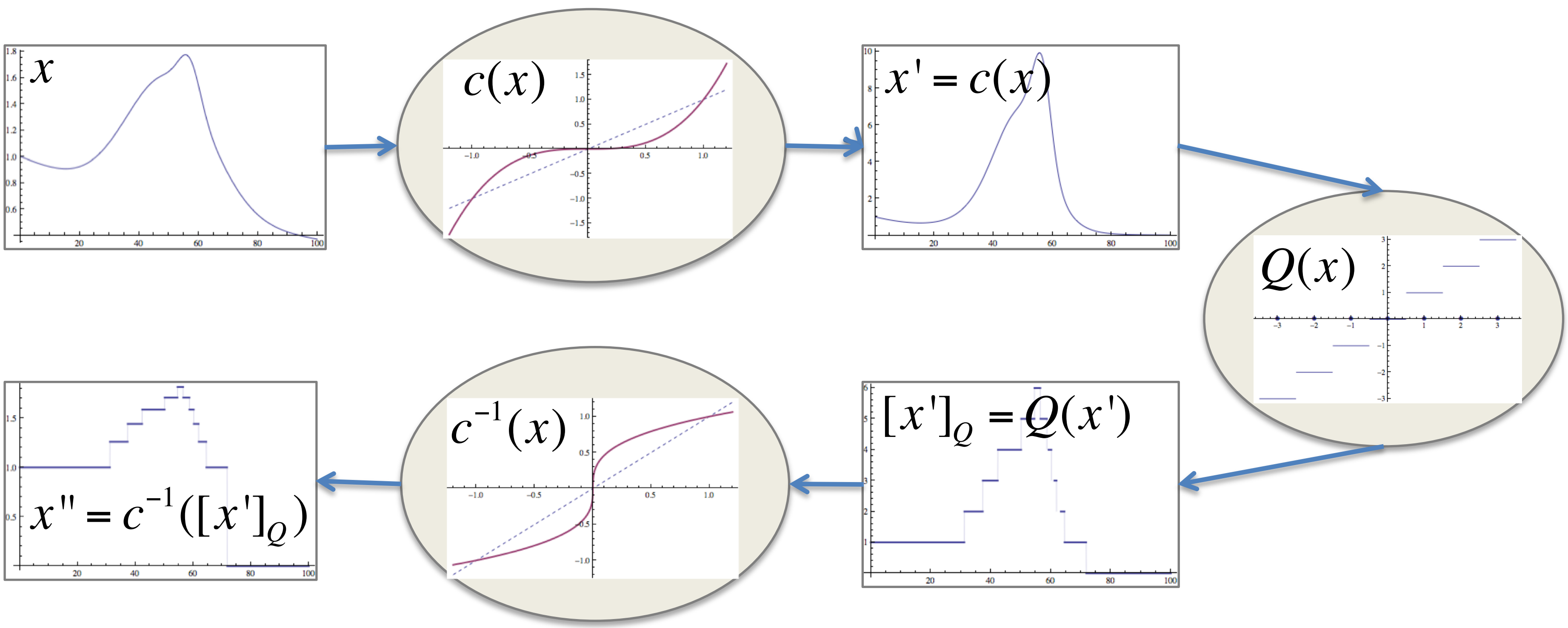
Algorithm [Max (1960), Lloyd (1982)]

Given: $p(x)$, initial boundaries $\{x_q\}$
repeat
$\quad y_q \leftarrow y_q^I(\{x_q\})$
$\quad x_q \leftarrow \frac{y_q + y_{q+1}}{2}$
endrepeat

Perception-optimized

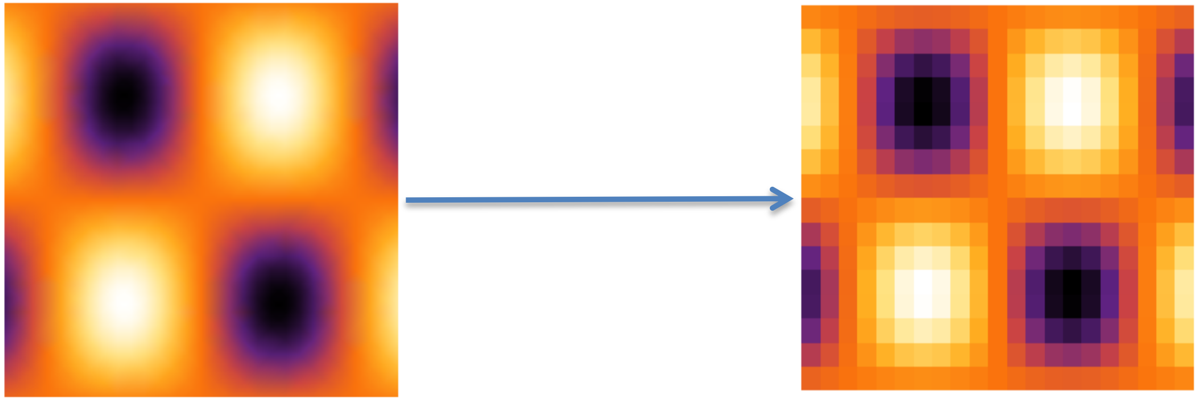
- Variance of quantization error does not exactly correspond to how distortions are perceived; usually strong dependence on amplitude (cf. sound)
- Minimize power of average <u>perceived</u> error (computed with models, human tests, ...)
- Hard to optimize for all possible cases and applications!

SNR (Signal-to-noise ratio) – optimized quantization:
introduce some compander (Compressor+Expander) function ***c(x)***, pre-process signal

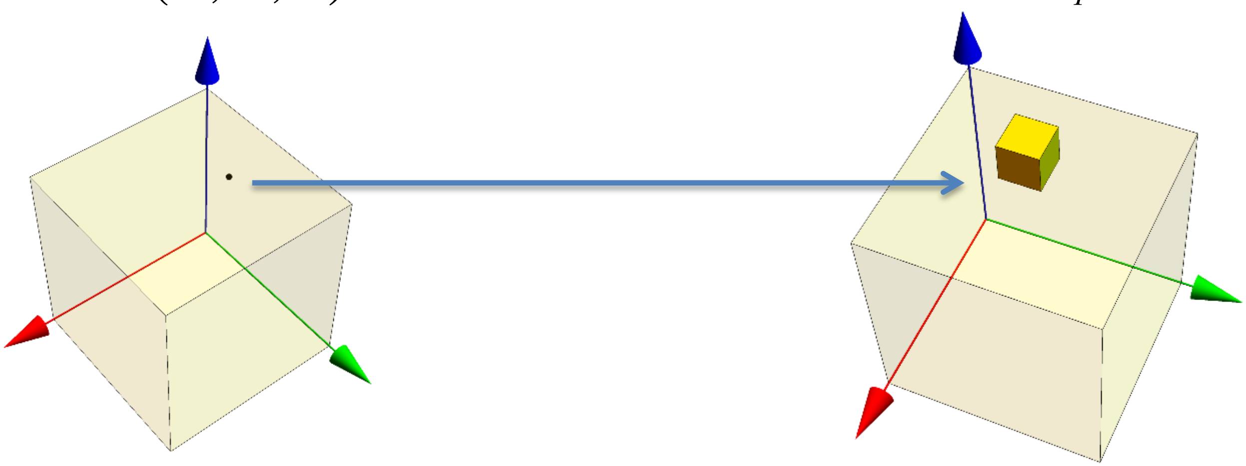


Vector quantization: multi-dimensional signals

$g(x, y)$ $\quad\quad$ $g(x_a, y_b)$



$\vec{v} = (R, G, B)$ $\quad\quad$ $(R, G, B) \rightarrow v_q$



Vector quantization as a classification problem

- Quantization: $\vec{x} \rightarrow q$
- Reconstruction: $q \rightarrow \vec{y}_q$

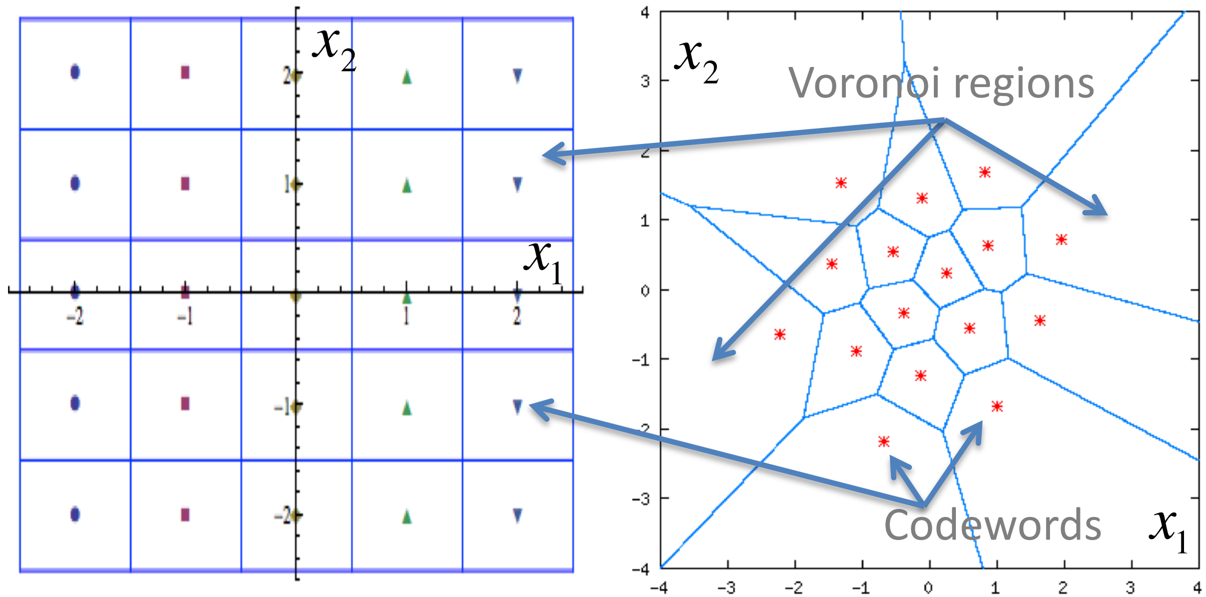
- Codebook:

$$Y = \{\vec{y}_i : i = 1,2,...,N\}$$

- Distance metric:

$$d(\vec{x},\vec{y}) \in R^+$$

- Voronoi region:



$$V_i = \{\vec{x} \in R^k : d(\vec{x},\vec{y}_i) \le d(\vec{x},\vec{y}_j), \forall j \ne i\}$$

- Non-overlapping space partitioning:

$$\bigcup_{i=1}^{N} V_i = R^k$$

$$V_i \bigcap V_j = \varnothing \forall i \ne j$$

- Example: Euclidean metric

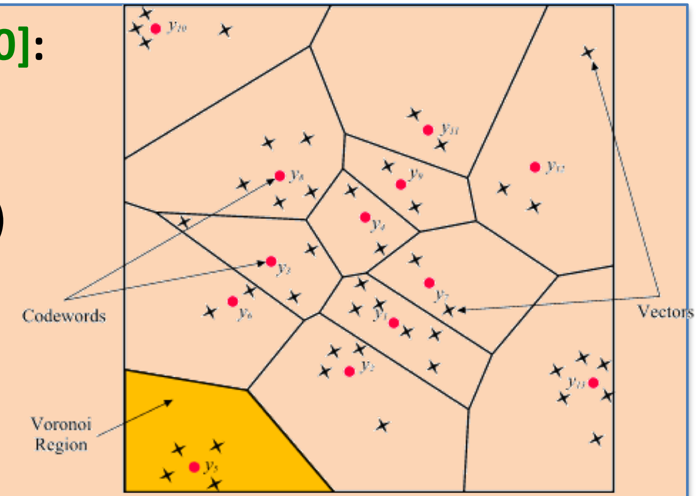
$$d_e(\vec{x},\vec{y}) = \sqrt{\sum (x_i - y_i)^2}$$

Vector quantization: codebook design

- Optimal design of a codebook for given input vectors is NP-hard (classification problem)

LGB: simple (sub-optimal) algorithm [Linde, Buzo, Gray ‘80]:

1. Determine size N of the codebook
2. Select N random codewords as initial codebook Y
3. Classify set of M training vectors (i.e. find V_i for each x_k)
4. Compute new codewords as means in each cluster:



$$\vec{y}_i \leftarrow \frac{1}{N_i} \sum_{\vec{x}_k \in V_i} \vec{x}_k$$

5. Repeat steps 2 and 3 until convergence (i.e. changes in codewords become small)
6. Alternatively: threshold on relative quantization error change, $\Delta D / D < \varepsilon$:

$$D = \frac{1}{M} \sum_i D_i, \quad D_i = \frac{1}{N_i} \sum_{\vec{x}_k \in V_i} d(\vec{x}_k, \vec{y}_i)$$

(cf. k-means algorithm)

- Alternative VQ methods: Pairwise Nearest Neighbour (PNN), Simulated Annealing, Maximum Descent (MD), Frequency-Sensitive Competitive Learning (FSCL), etc.