Today’s lecture

- Repetition and refinement of some basic concepts:
  - What is an image?
  - Convolution
  - The Fourier Domain
  - Sampling
  - Aliasing
  - Interpolation
  - Point operations
  - Thresholding

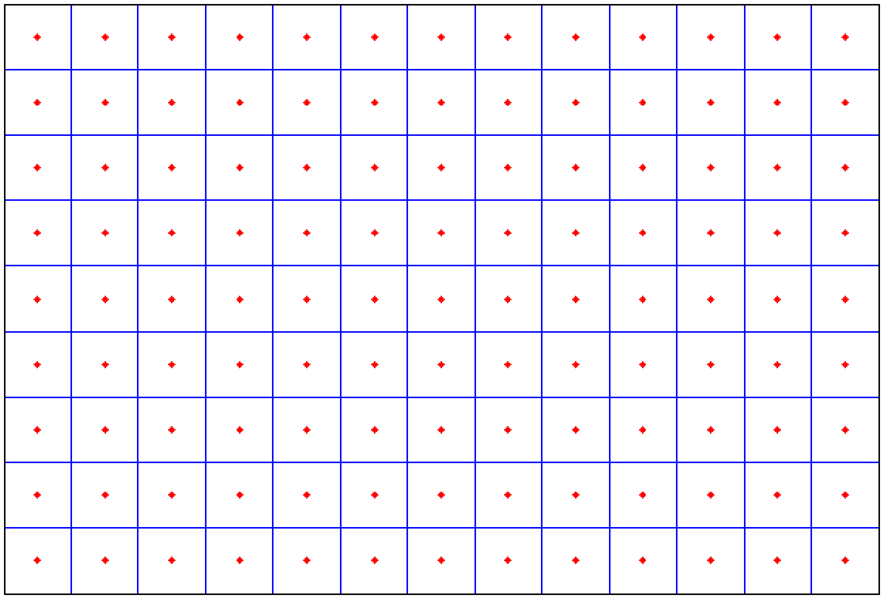
Image definition

- Continuous image $f: \boldsymbol{D}^n \to \mathbb{R} \quad , \quad \boldsymbol{D} \subset \mathbb{R}$
  (**D** is a compact subdomain of $\mathbb{R}$)
  - Finite in size
  - Each point has a scalar value
- Multi-valued image $f: \boldsymbol{D}^n \to \mathbb{R}^k \quad , \quad \boldsymbol{D} \subset \mathbb{R}$
  - Each point has multiple values
- Discrete image $f: \boldsymbol{D}^n \to \mathbb{R} \quad , \quad \boldsymbol{D} \subset \mathbb{Z}$
  - Sampled version of continuous image
  - Sample values still real-valued!
- Digital image $f: \boldsymbol{D}^n \to \mathbb{Z} \quad , \quad \boldsymbol{D} \subset \mathbb{Z}$
  - Sample values also discretized

Samples and pixels

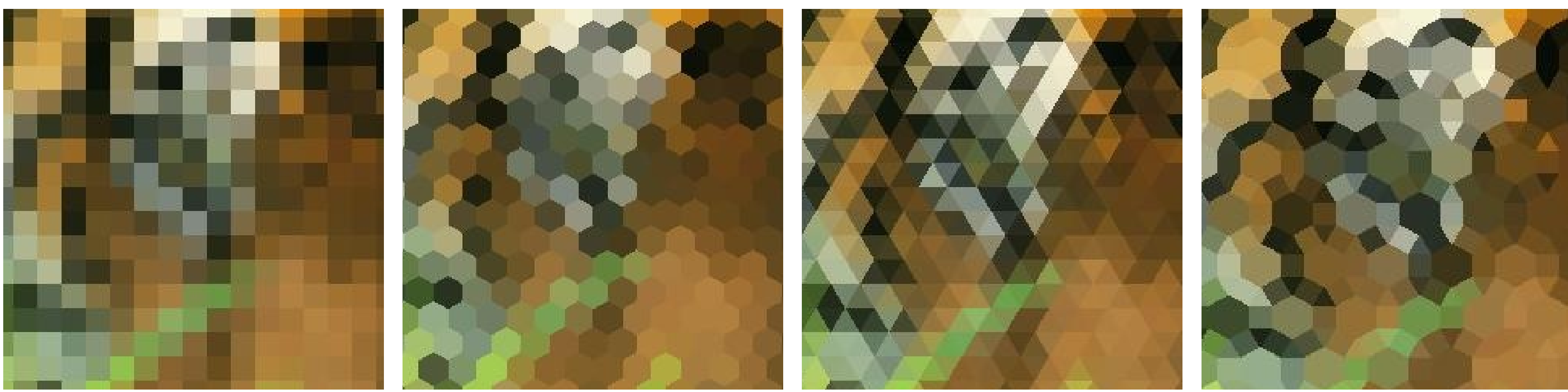
- Sample: the value $f(\boldsymbol{x})$ of a function at a point $\boldsymbol{x}$
  - 1-D signals: people talk of “samples”
  - 2-D images: people talk of “pixels”
  - 3-D images: people talk of “voxels”
  - These are all the same thing!   (suggested: imels)

- Pixels represented as little rectangles on the screen
  - This is the Voronoi tesselation of the samples in a rectangular grid



Alternative sampling grids

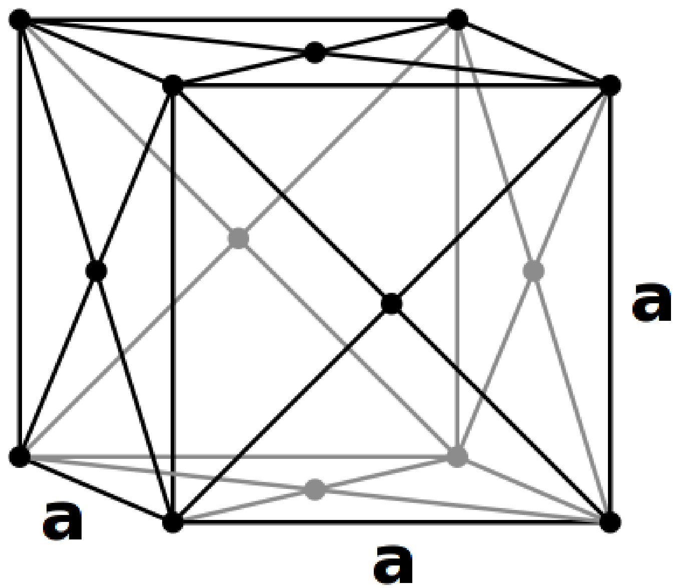
- Do we always need to use rectangular grids?
- Alternative in 2-D: hexagonal grid
  - Advantages:
    - All direct neighbours at same distance
    - All direct neighbours share an edge (no "connectivity paradox")
    - 3 directions of equal distance, vs 2 in square grid

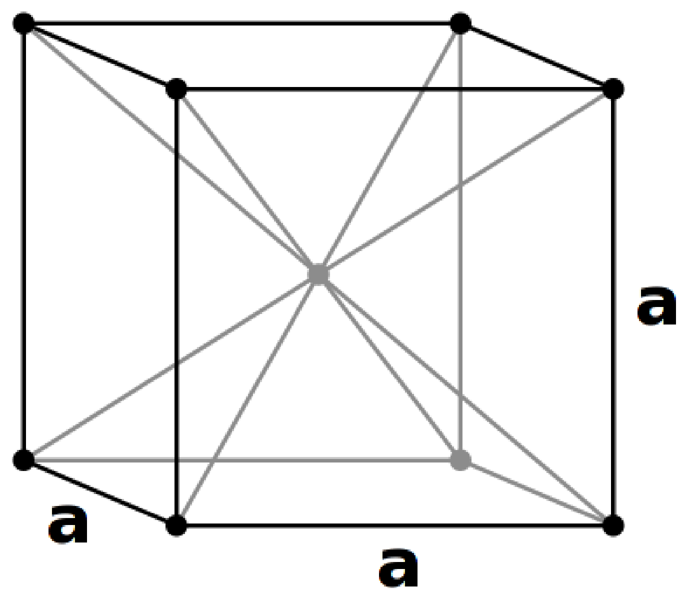


Alternative sampling grids

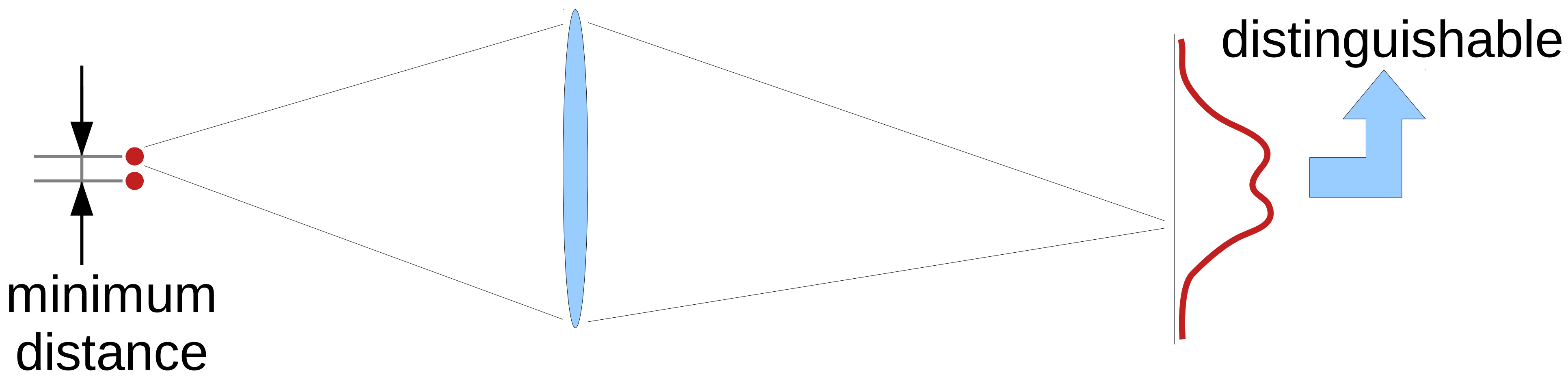
Alternative sampling grids

- Alternatives in 3-D (generalizations of hexagonal grid):
  - Face centered cubic
    - Voronoi tessellation is rhombic dodecahedron
    - Tightest possible packing density (Kepler)
  - Body centered cubic
    - Voronoi tessellation is truncated octahedrons
    - Like hexagonal grid, all direct neighbors share faces





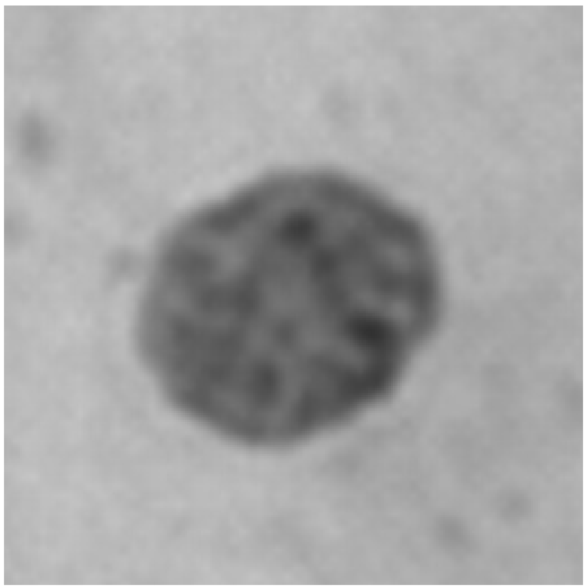
What is resolution?



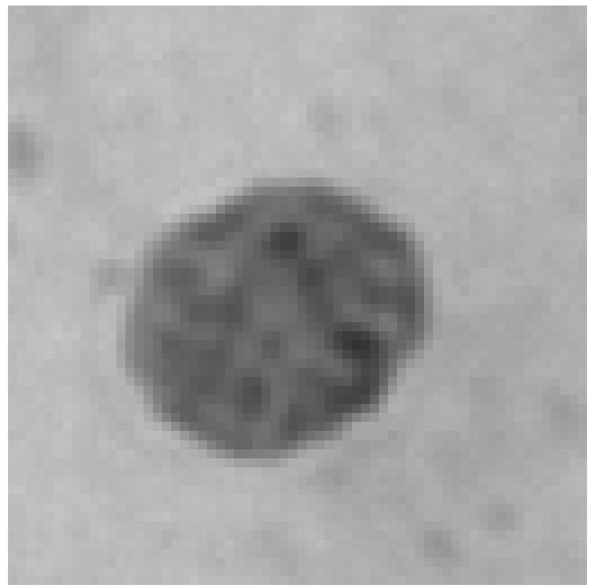
How is this related to sampling density/pixel size?
And how is this related to number of pixels?

Image Acquisition: Sampling

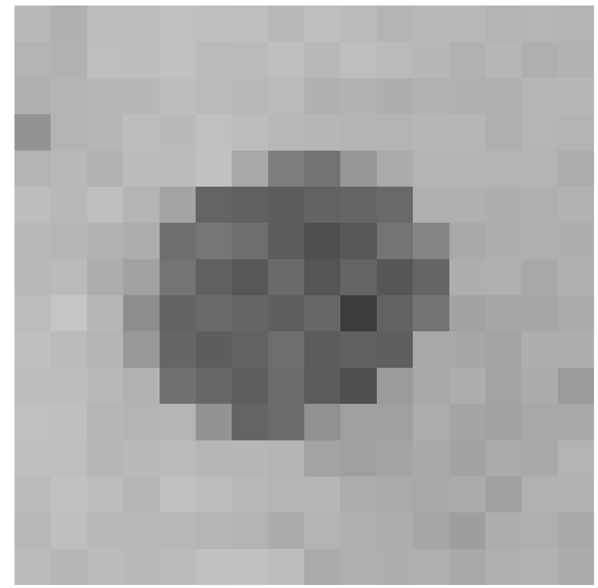
Oversampled



Correctly sampled



Undersampled

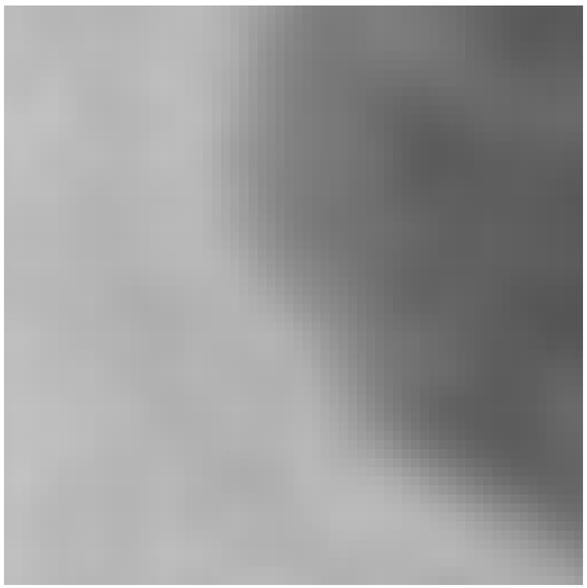


Wastes computer memory

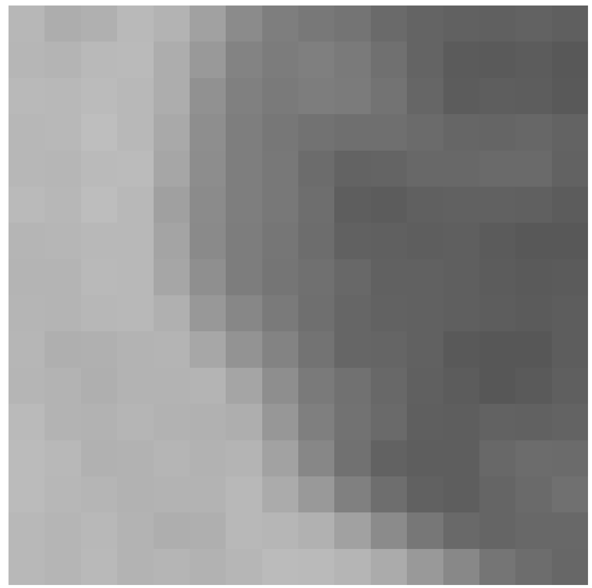
Looses information

Image Acquisition: Sampling

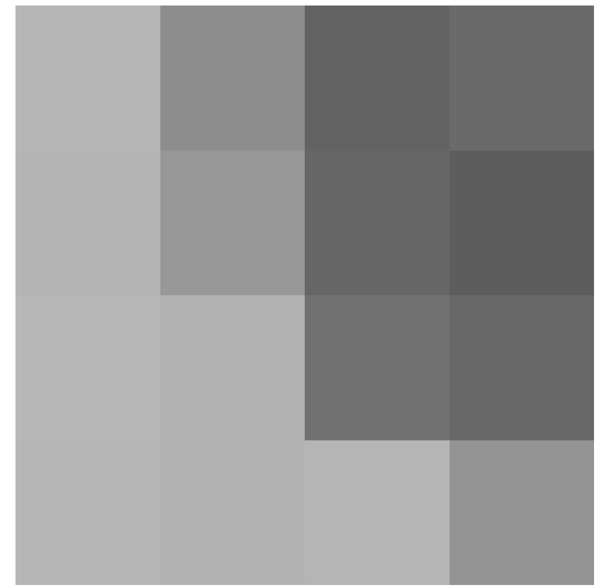
Oversampled



Correctly sampled



Undersampled

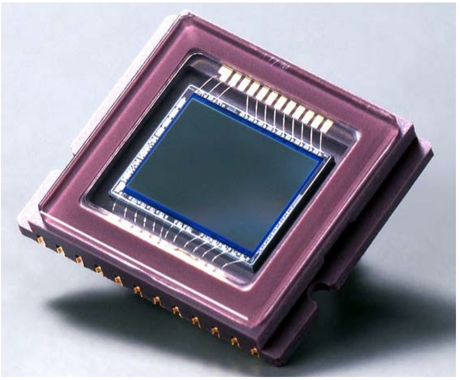


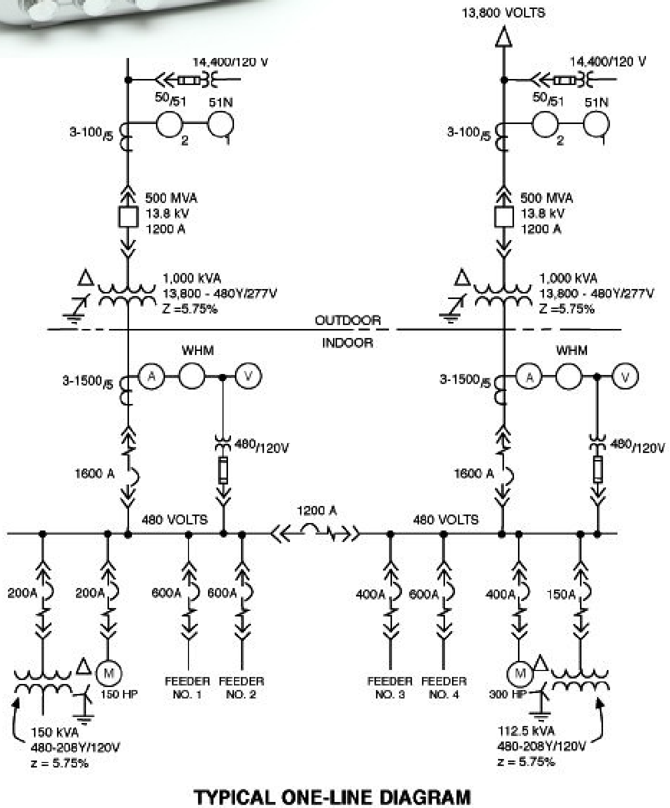
Wastes computer memory

Looses information

Convolution

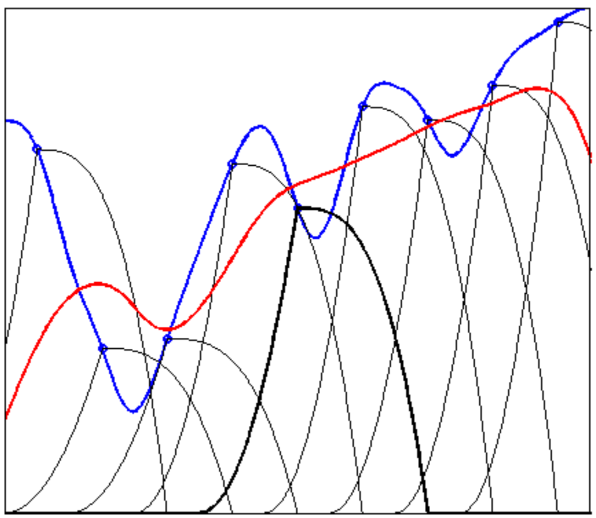
- Convolution mimics Linear Time Invariant (LTI) systems:
  - Sampling
  - Reconstruction
  - Microscope lenses
  - CCD sensor
  - Electric circuits
  - Radio
  - …



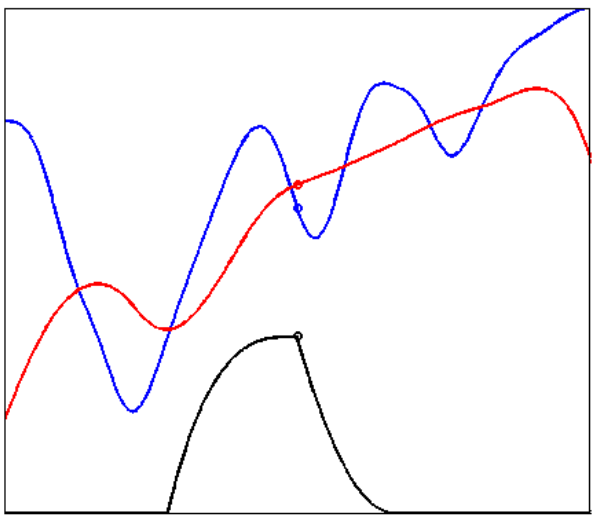


Convolution

$$g(x) = \int_{-\infty}^{\infty} f(\xi)\, h(x-\xi)\, d\xi$$



Sum of infinite number of kernels, weighted with input function



For each output point, integral of multiplication of mirrored kernel with input function

Convolution

$$g(x) = \int_{-\infty}^{\infty} f(\xi)\, h(x-\xi)\, d\xi$$

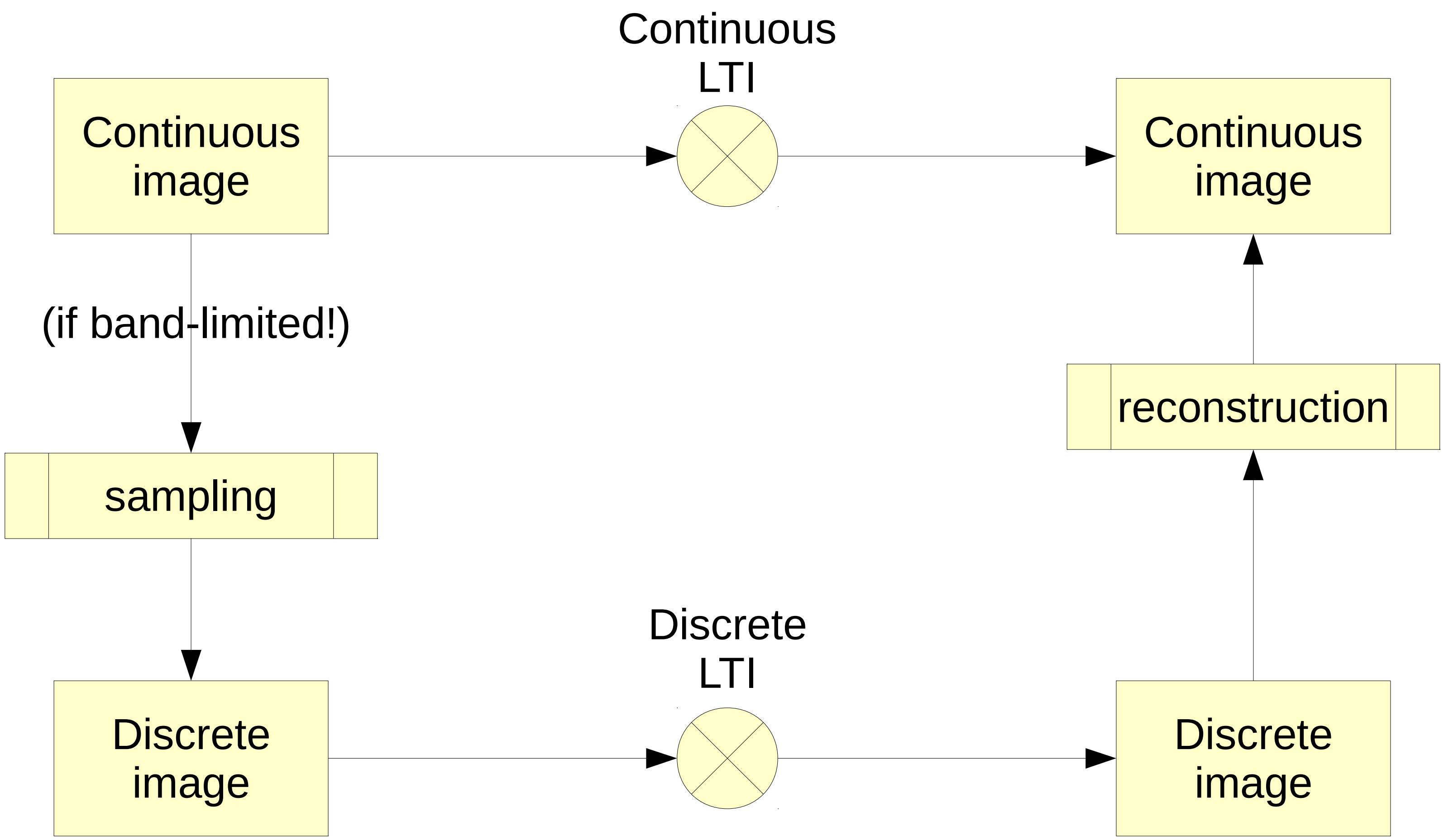
- For discrete image and discrete kernel: finite sum

$$g[n] = \sum_{k=0}^{N-1} f[k]\, h[n-k]$$

Convolution properties

- <u>Linear</u>:
  - Scaling invariant: $(a\,A)\otimes B=a(A\otimes B)$
  - Distributive: $A\otimes(B+C)=A\otimes B+A\otimes C$
- <u>Time Invariant</u>: $shift(A)\otimes B=shift(A\otimes B)$
- Commutative: $A\otimes B=B\otimes A$
- Associative: $A\otimes(B\otimes C)=(A\otimes B)\otimes C$

The sampling property



Convolution at the image edge

Convolution at the image edge

Mean padding

Zero order hold

Convolution at the image edge

Periodic boundary condition

Symmetric boundary condition

Fourier transform in 1D

$$F(\omega) = \int_{-\infty}^{\infty} f(x)\, e^{-i\omega x}\, dx$$

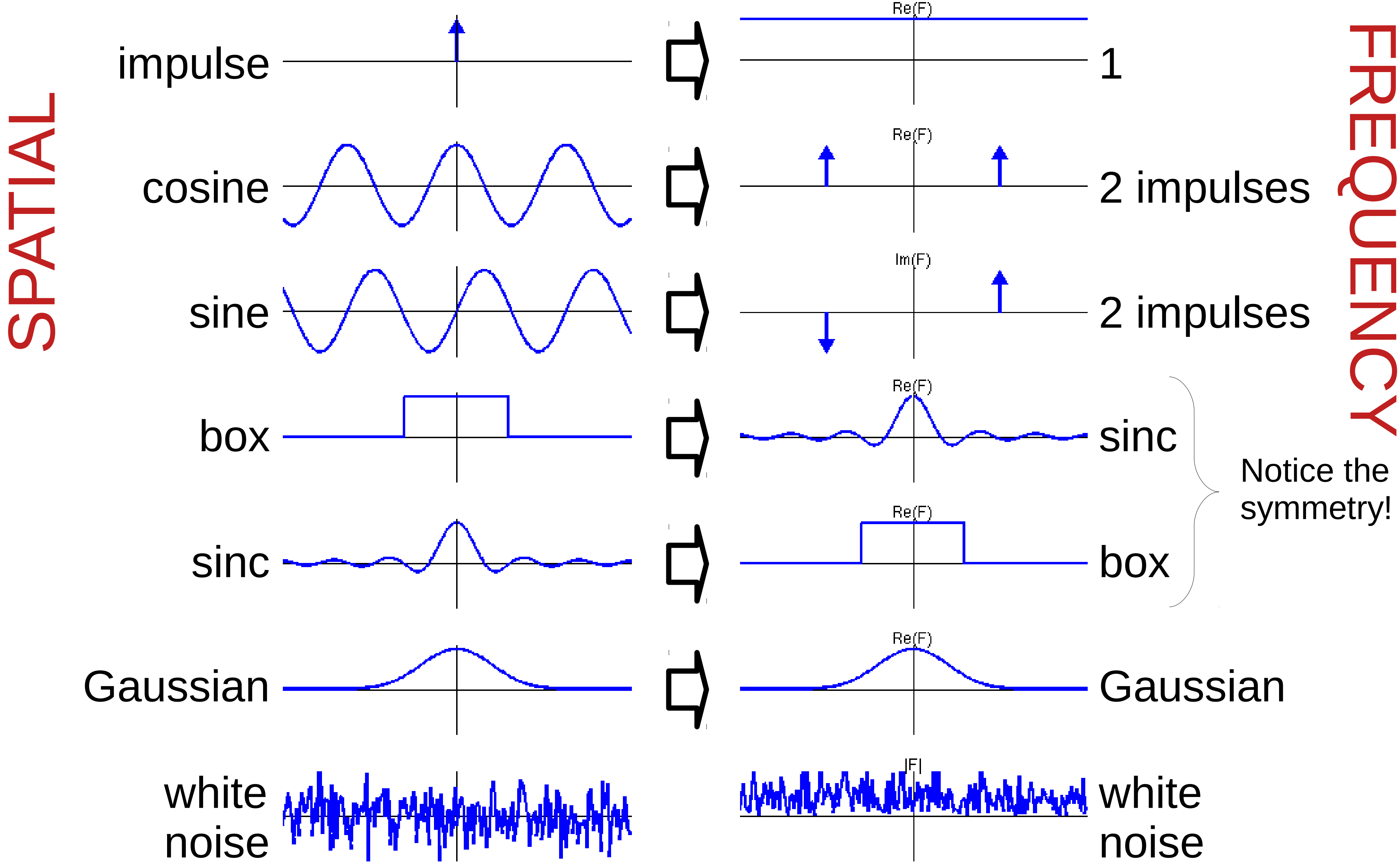
$$f(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)\, e^{i\omega x}\, d\omega$$

$$\omega = 2\pi f = \frac{2\pi}{T}$$

$$F[k] = \sum_{n=0}^{N-1} f[n]\, e^{-i\frac{2\pi}{N}kn}$$

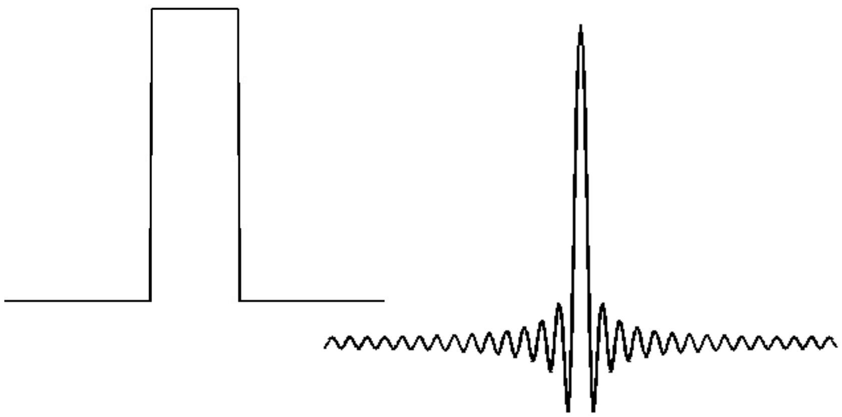
$$f[n] = \frac{1}{N}\sum_{k=0}^{N-1} F[k]\, e^{i\frac{2\pi}{N}kn}$$

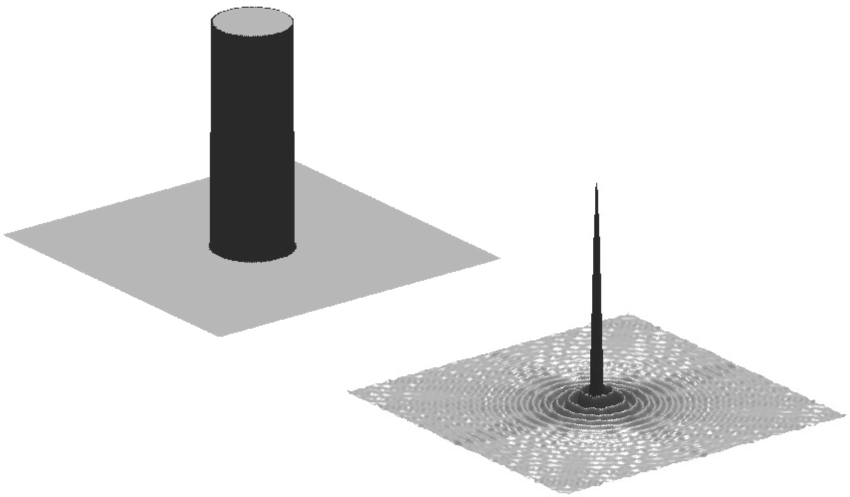
Fourier (frequency) domain



Fourier transform in 2D, 3D, etc.

- Simplest thing there is! — the FT is separable:
  - Perform transform along x-axis,
  - Perform transform along y-axis of result,
  - Perform transform along z-axis of result, (etc.)
- All the same properties apply as for 1-D Fourier Transform
- *Note error in the book (pg 59): 2D Fourier transform needs half the plane, not only the first quadrant!*





Properties of the Fourier transform

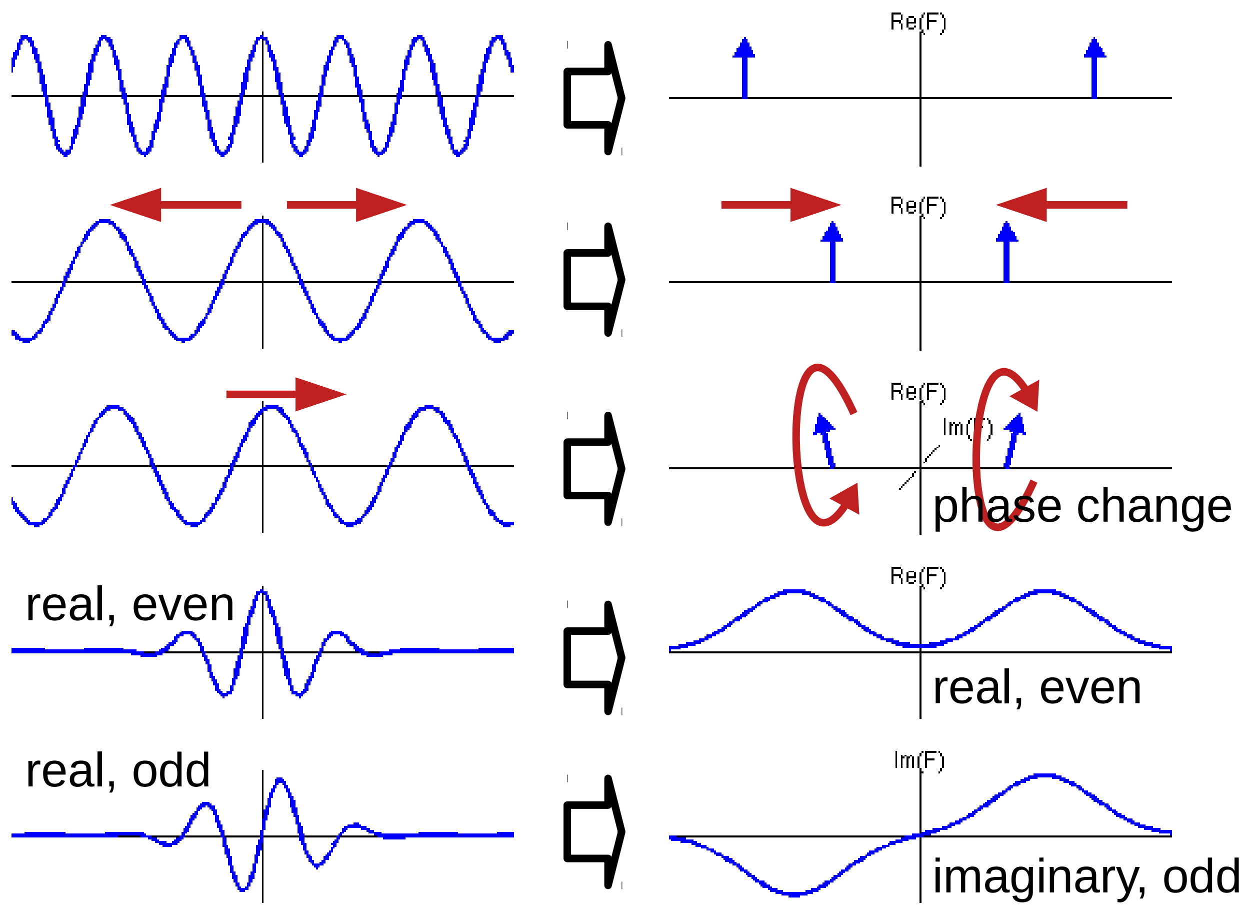
- Linearity

$$\mathscr{F}\{c\,A + d\,B\} = c\,\mathscr{F}\{A\} + d\,\mathscr{F}\{B\}$$

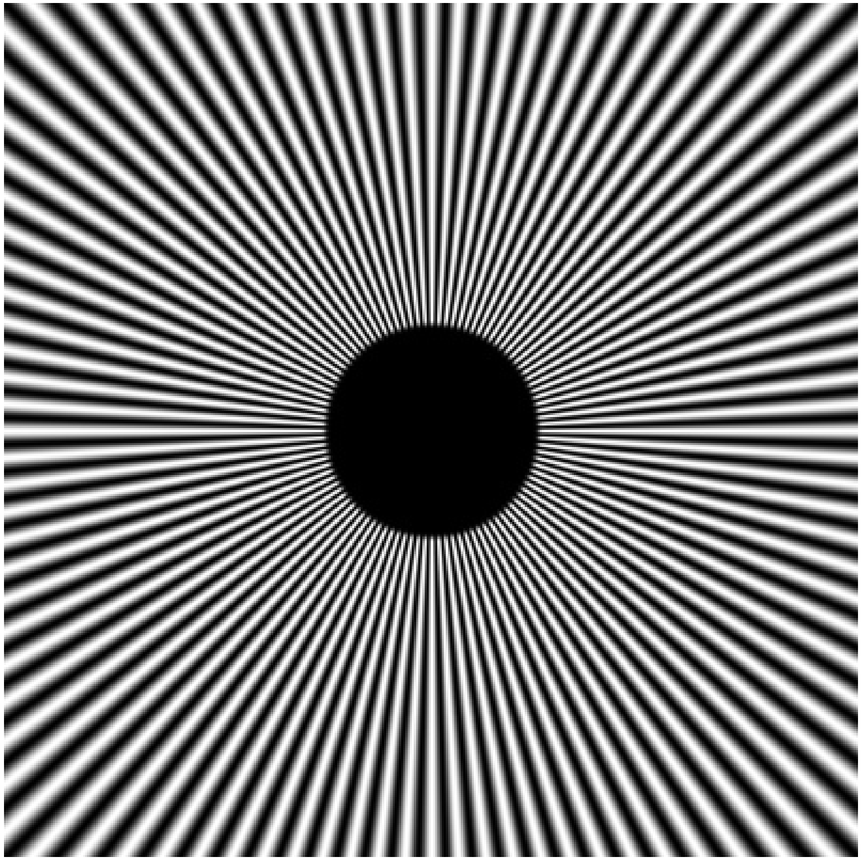
- Convolution

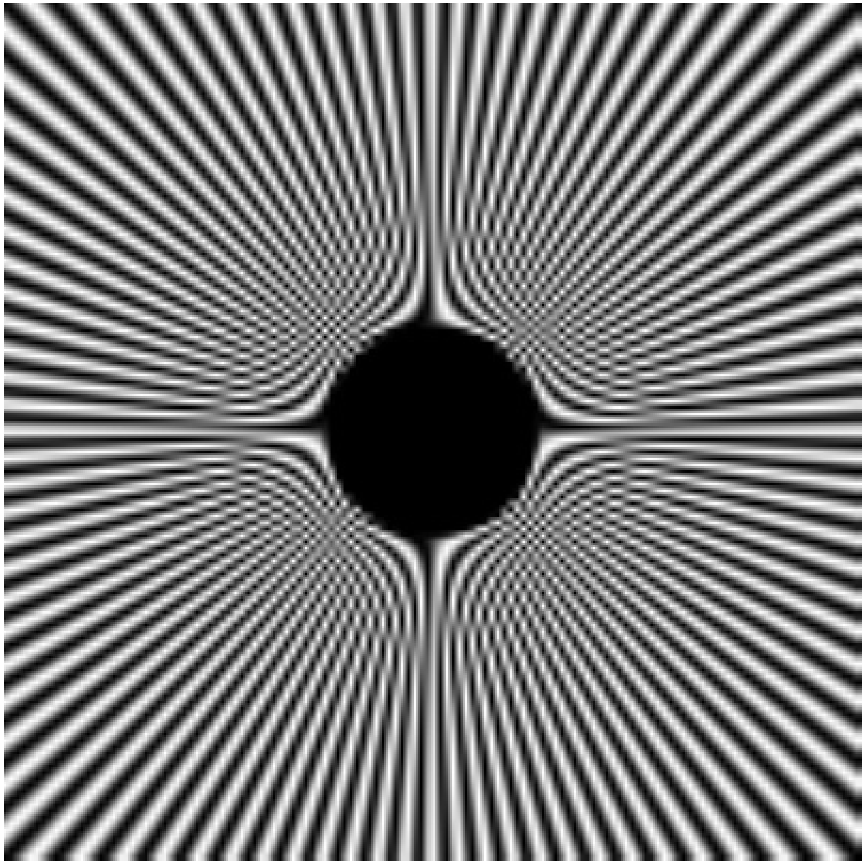
$$\mathscr{F}\{A \otimes B\} = \mathscr{F}\{A\} \cdot \mathscr{F}\{B\}$$

- Spatial scaling
- Spatial shift
- Symmetry



Aliasing

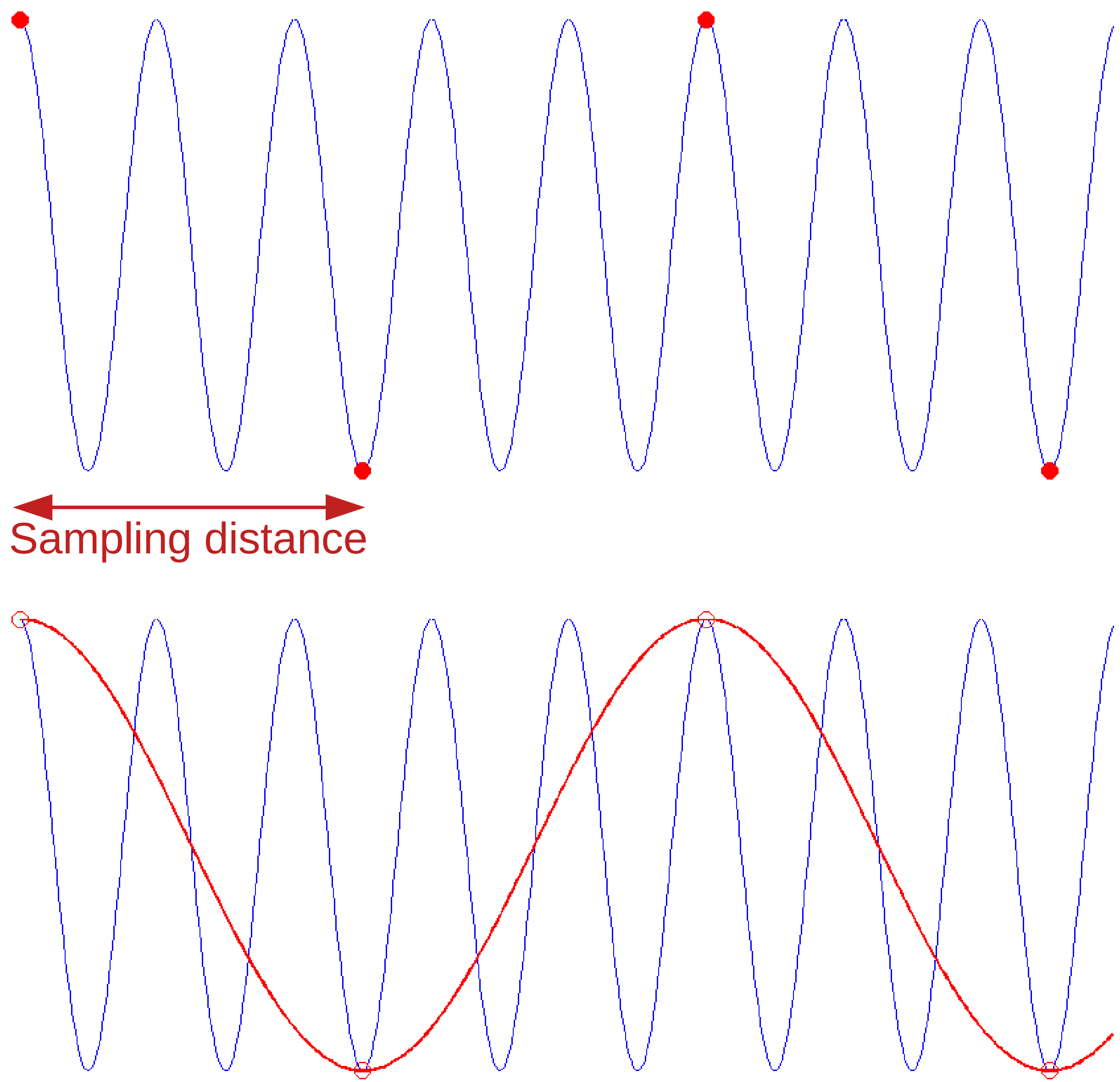




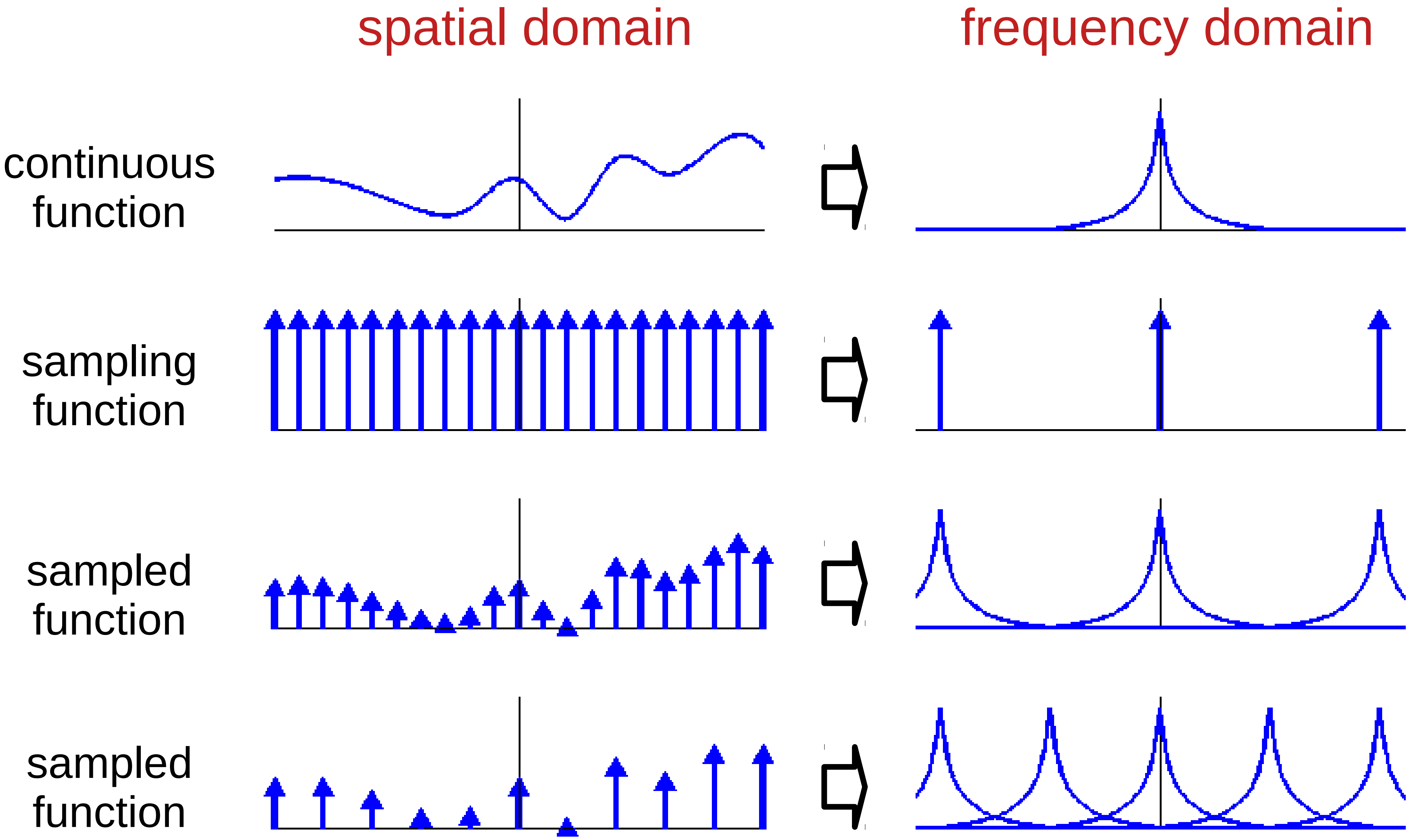
Discarded 8 of every 9 pixels

Aliasing

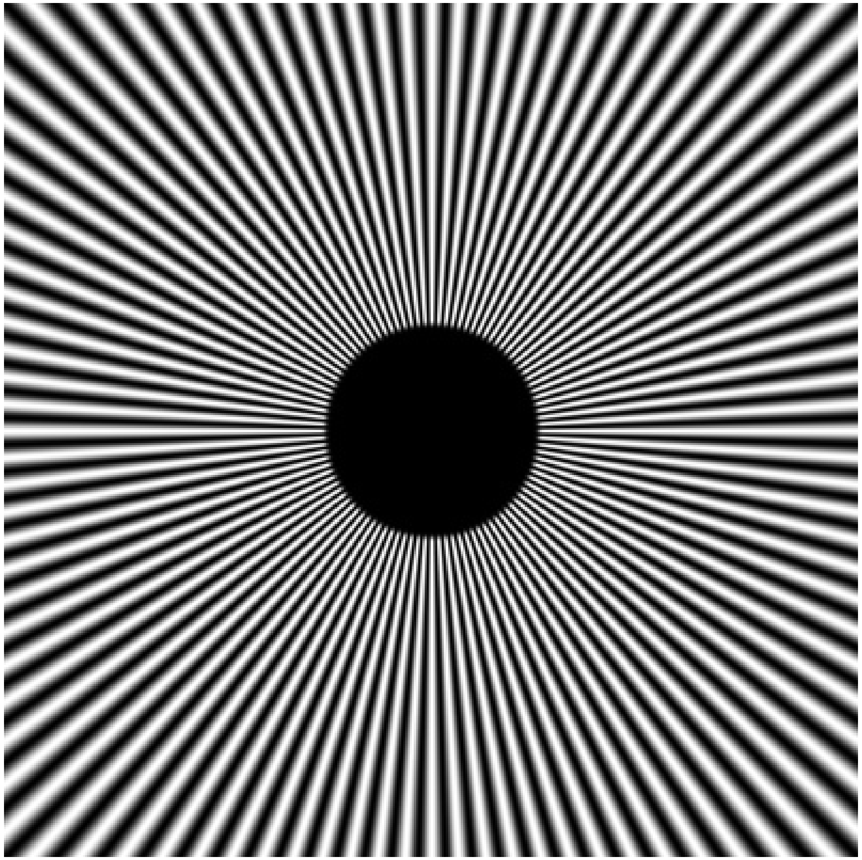
Aliasing

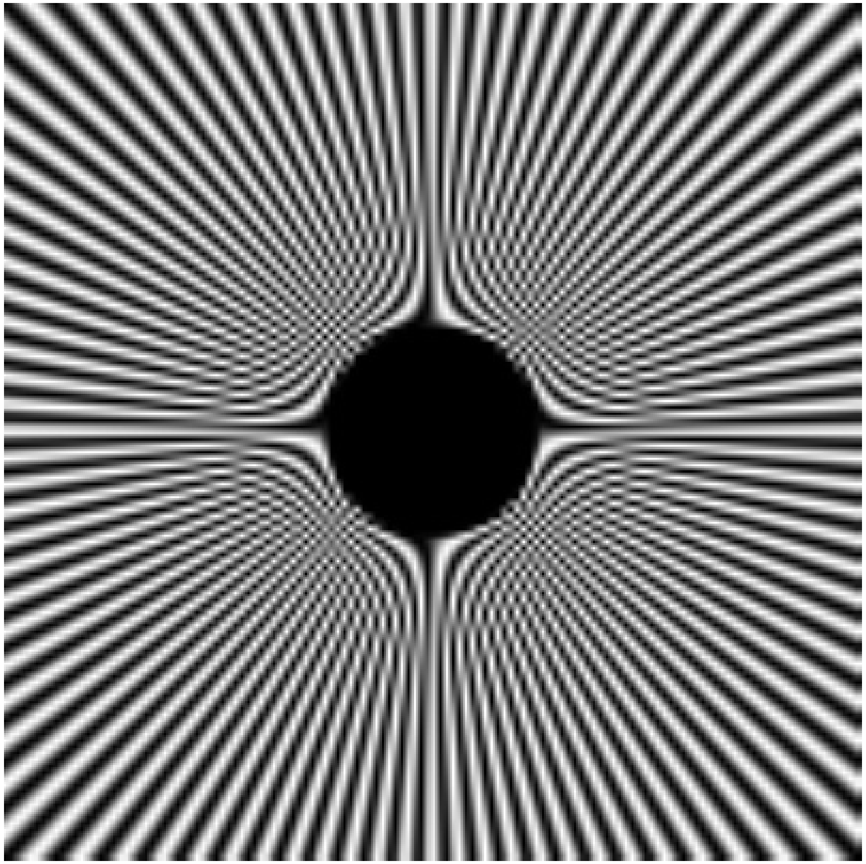


Revisiting sampling



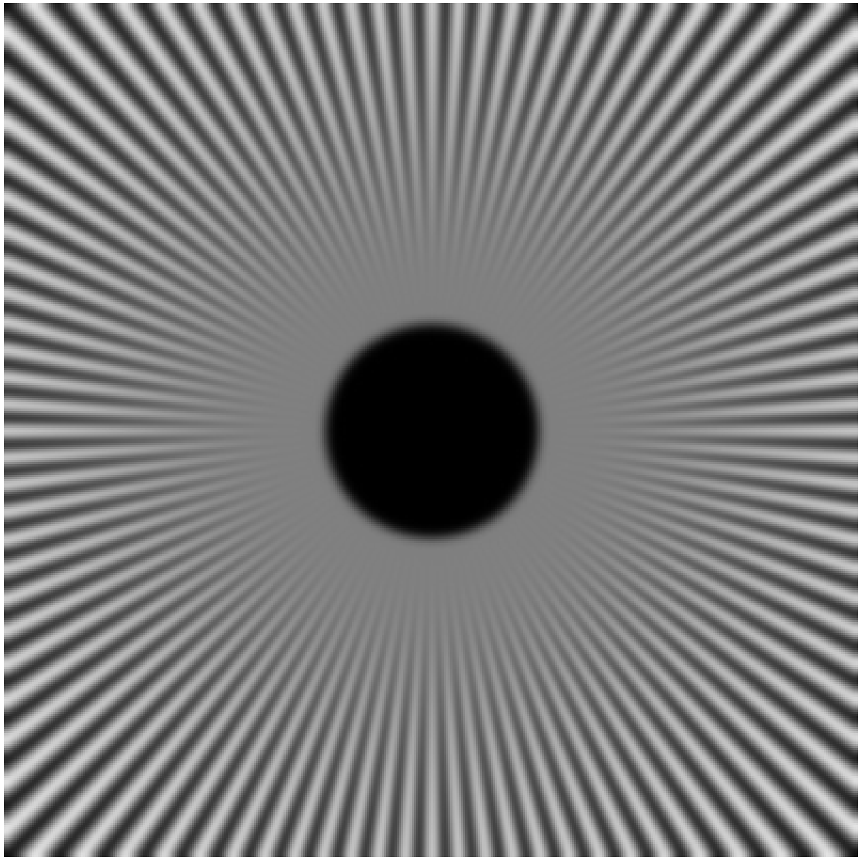
Avoiding aliasing



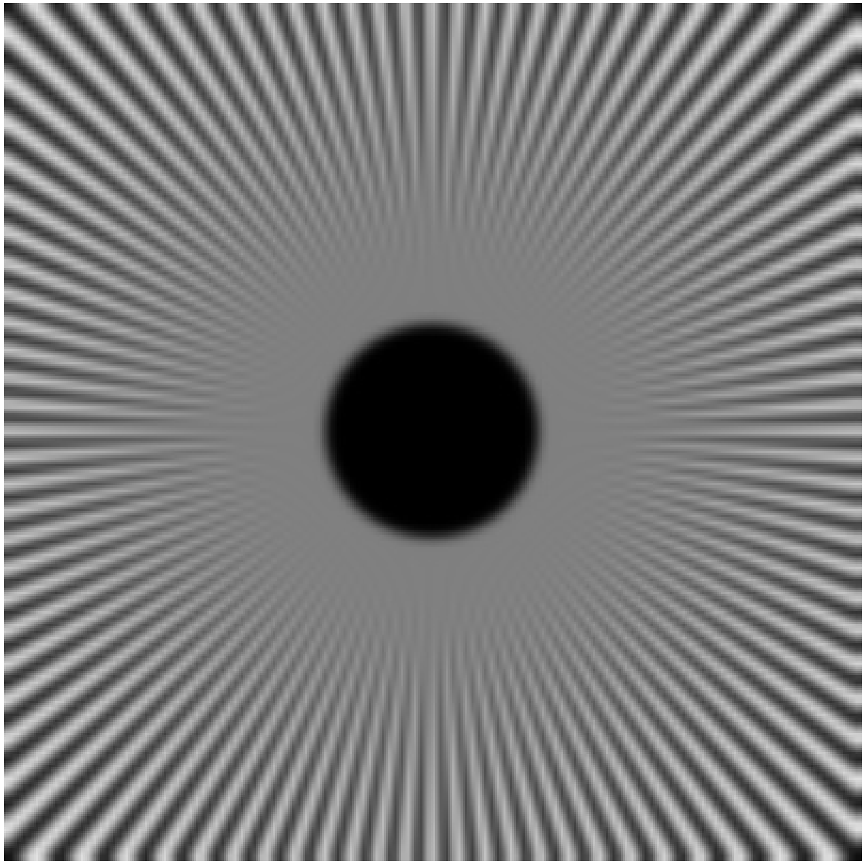


Discarded 8 of every 9 pixels

Avoiding aliasing

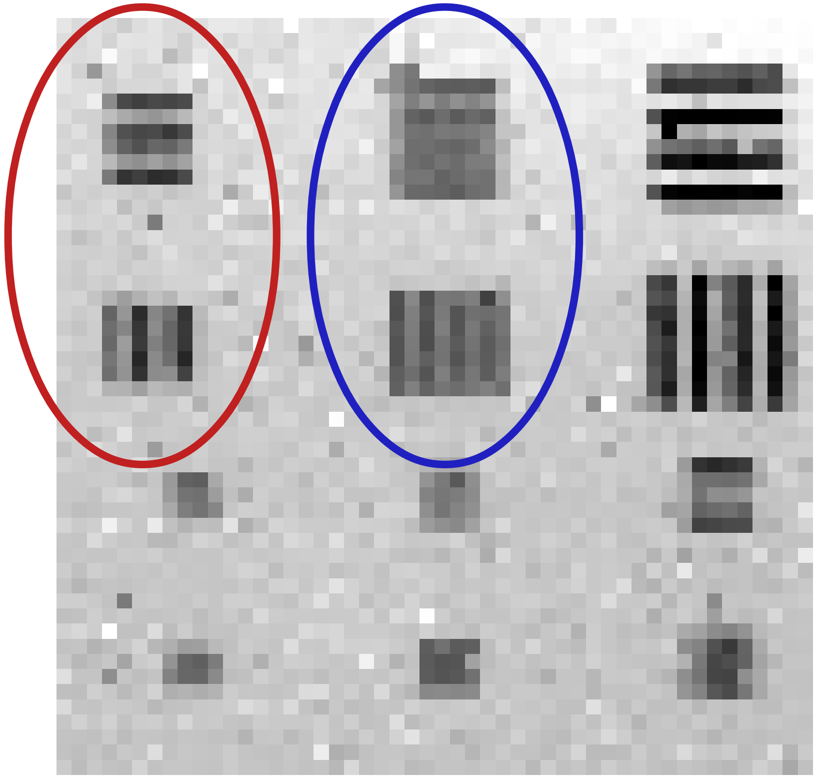


Low-pass filtered image does not contain high frequencies

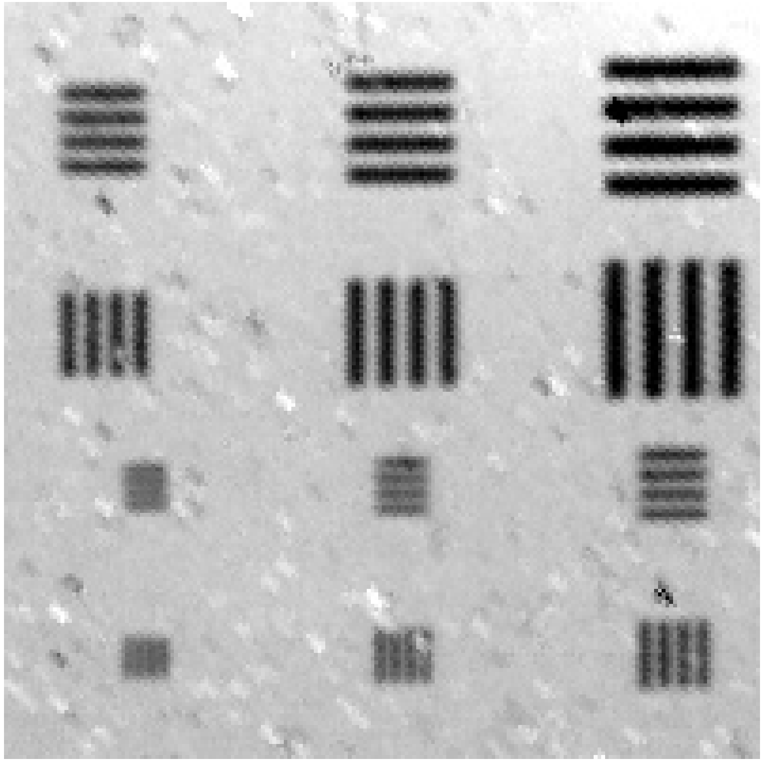


Discarded 8 of every 9 pixels

Aliasing in practice

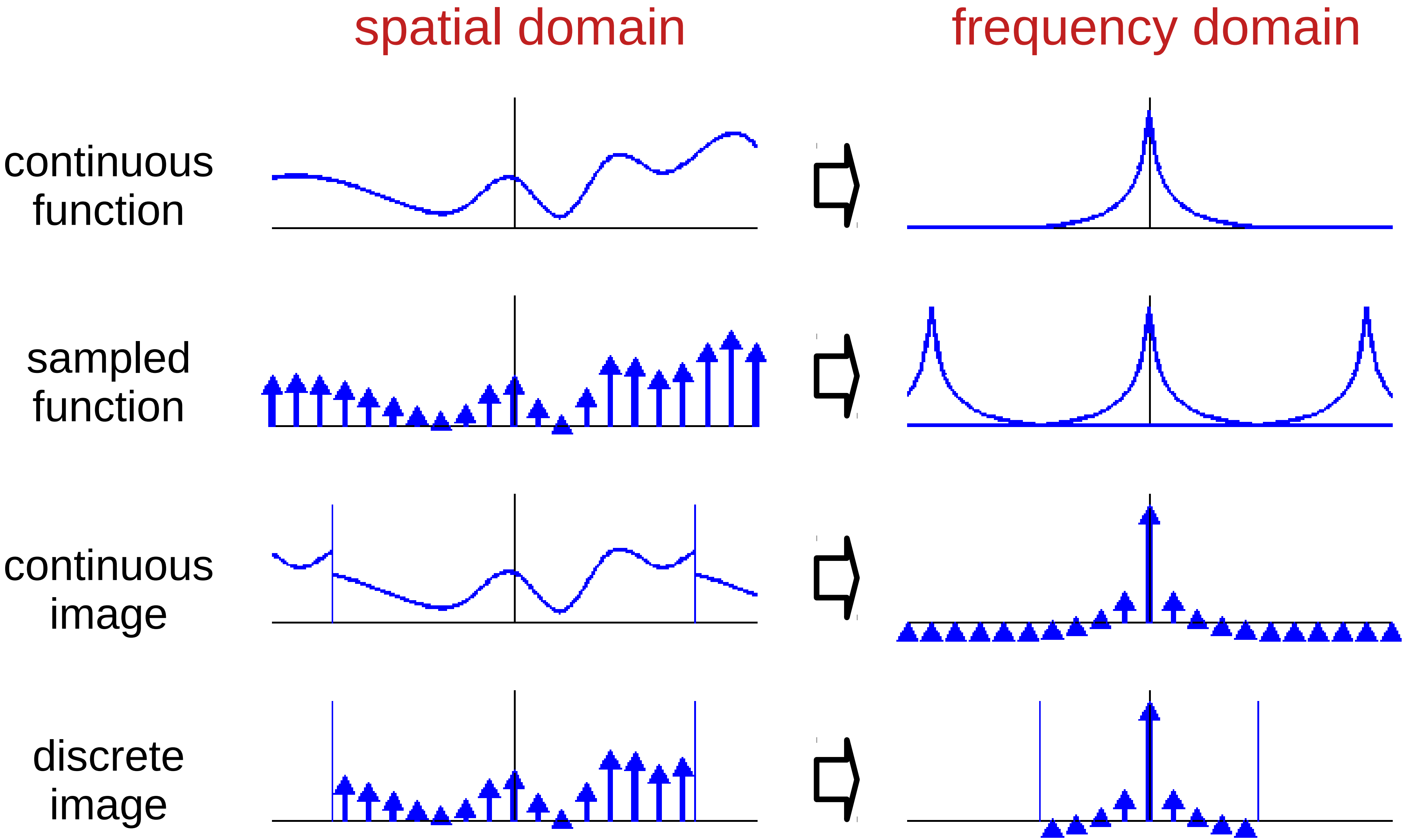


Undersampled

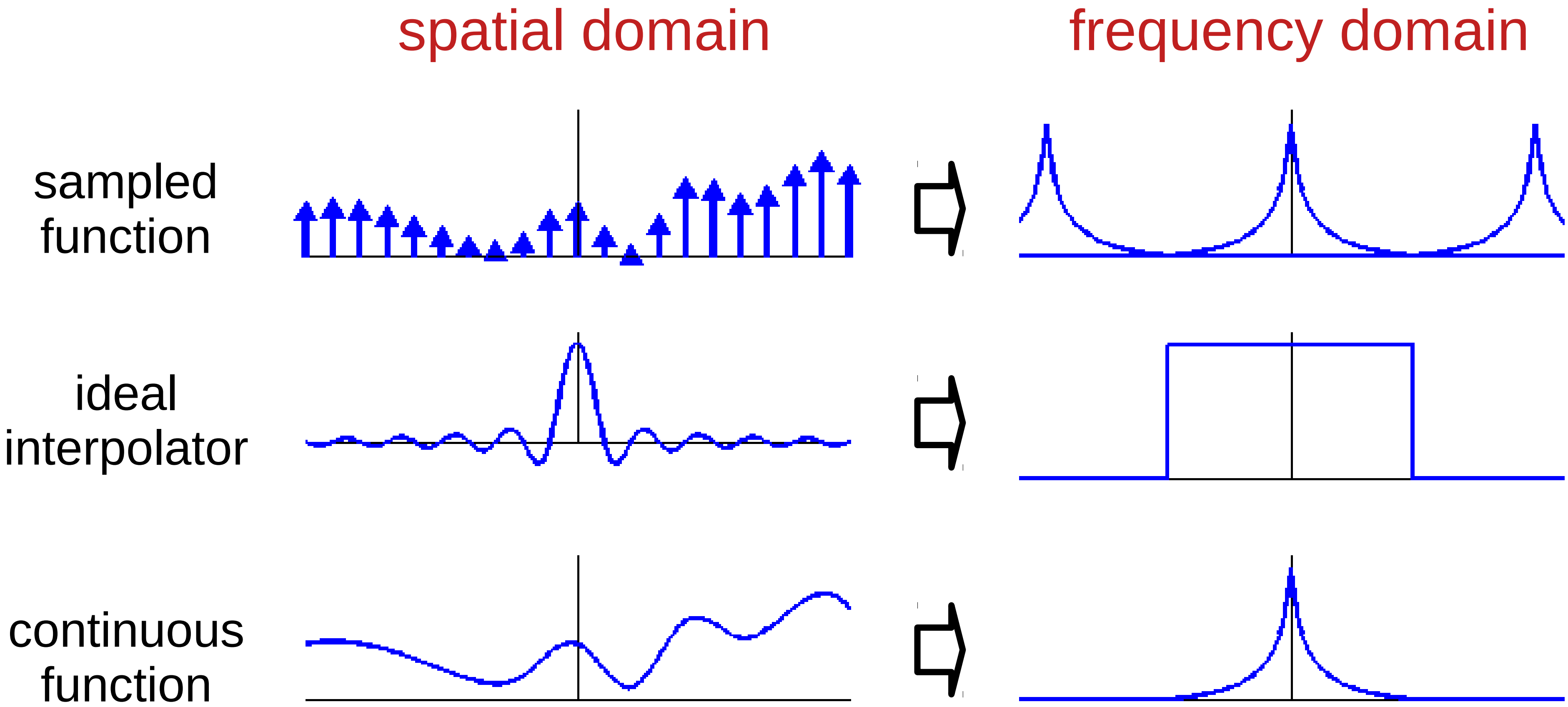


Not undersampled

Revisiting sampling

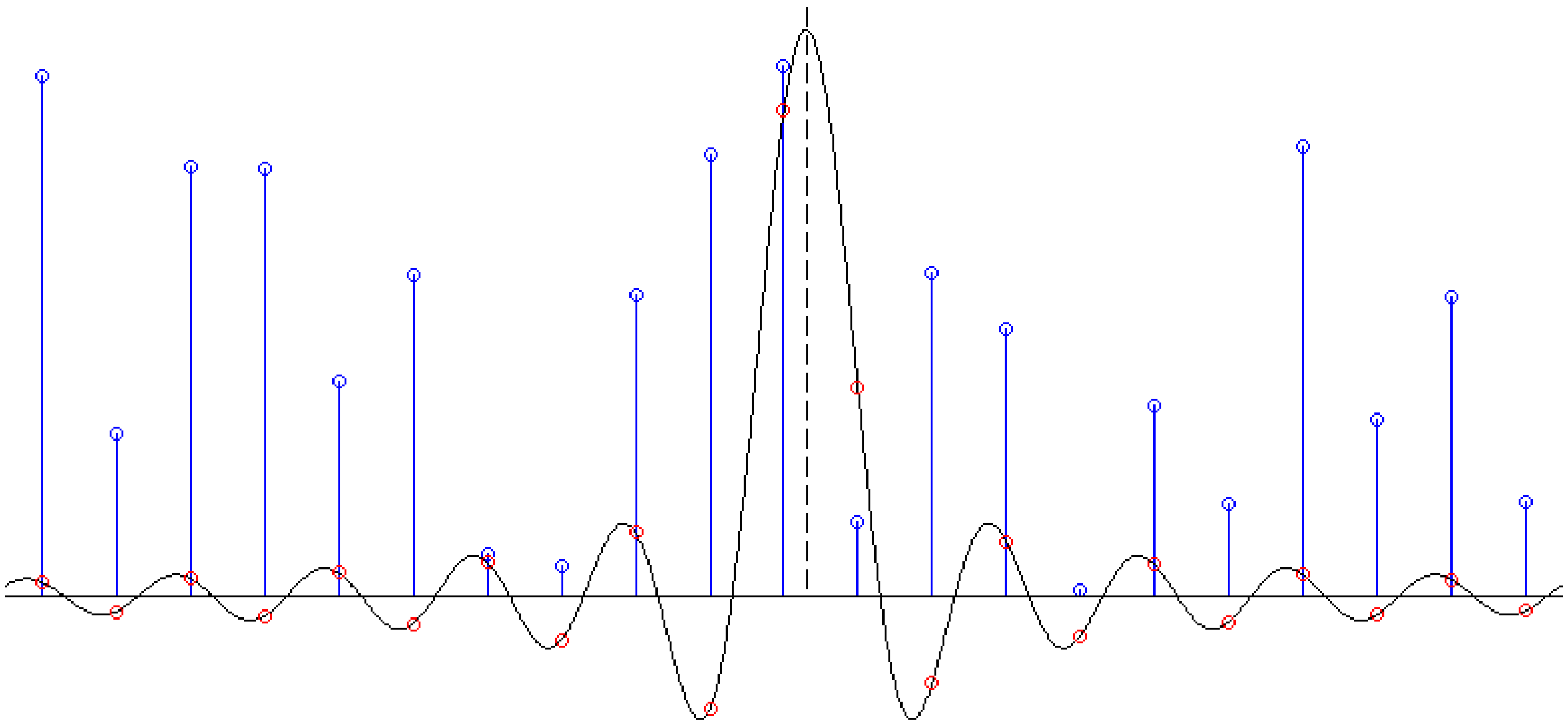


Reconstruction



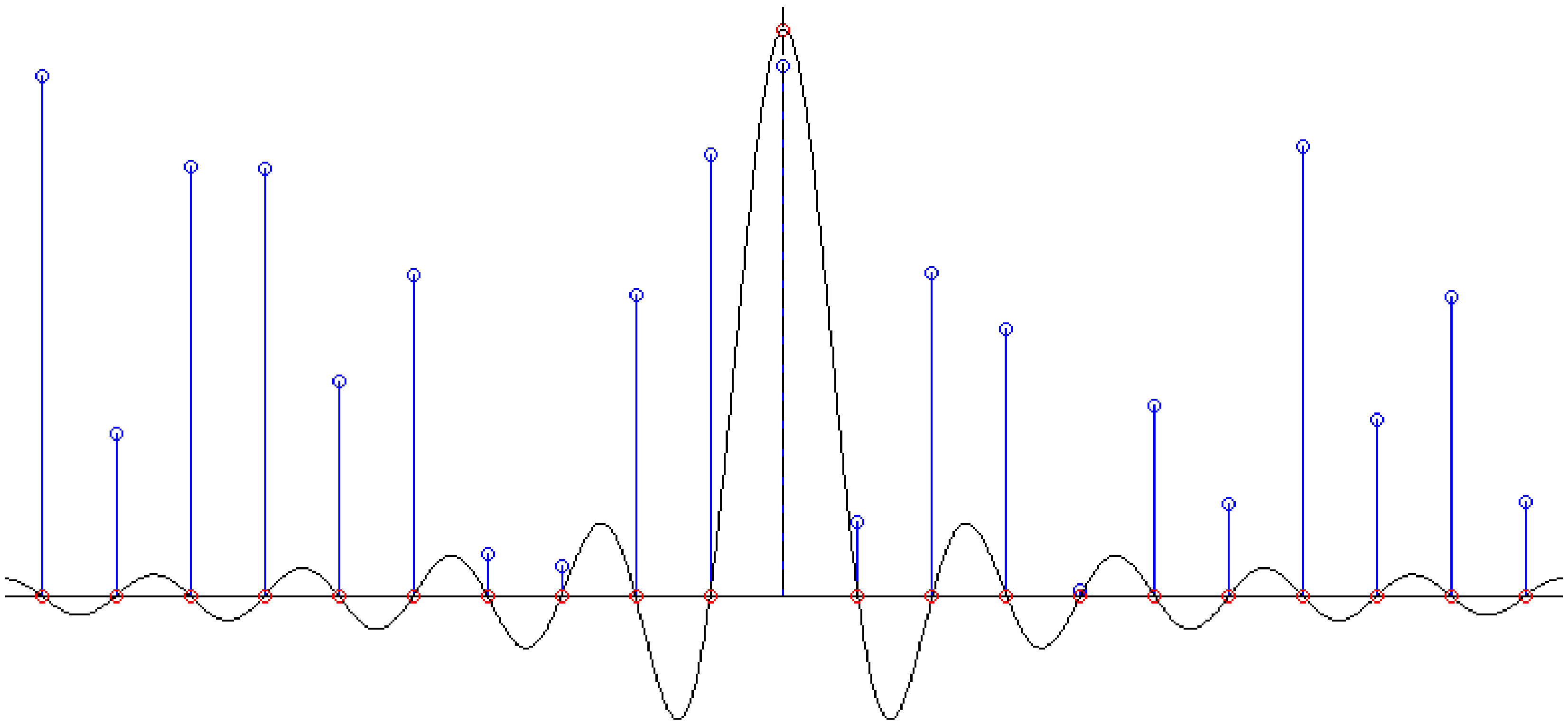
Ideal interpolator: $\mathrm{sinc}(x) = \sin(x)/x$

Interpolation



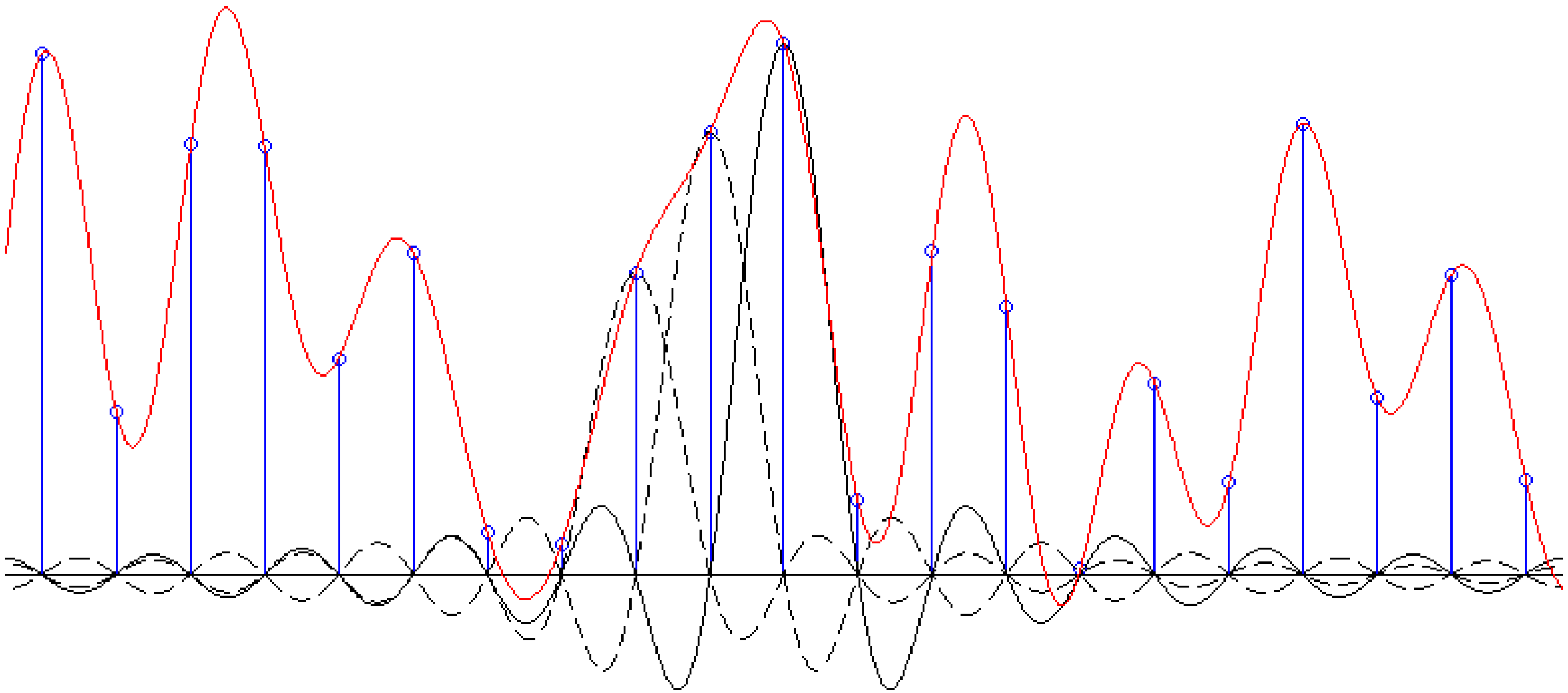
Sinc function gives weights for each pixel in image.
Computing value at one point requires knowledge
of all samples.

Interpolation



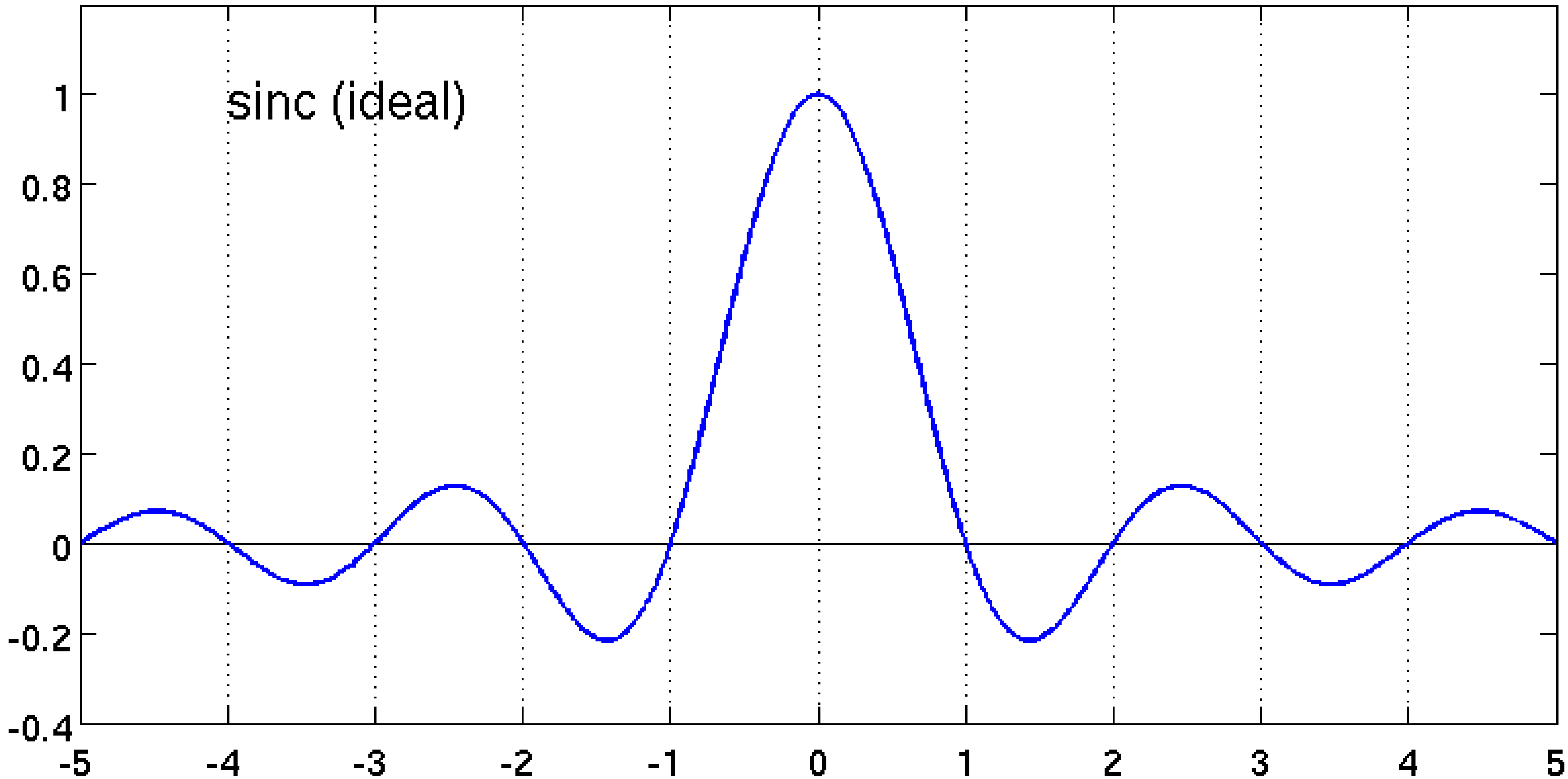
Sinc function is zero at all other grid points when origin matches a grid point: output is identical to sample value.

Interpolation



An alternative interpretation of the convolution integral places a sinc function centered at each sample point, and adds them all together.

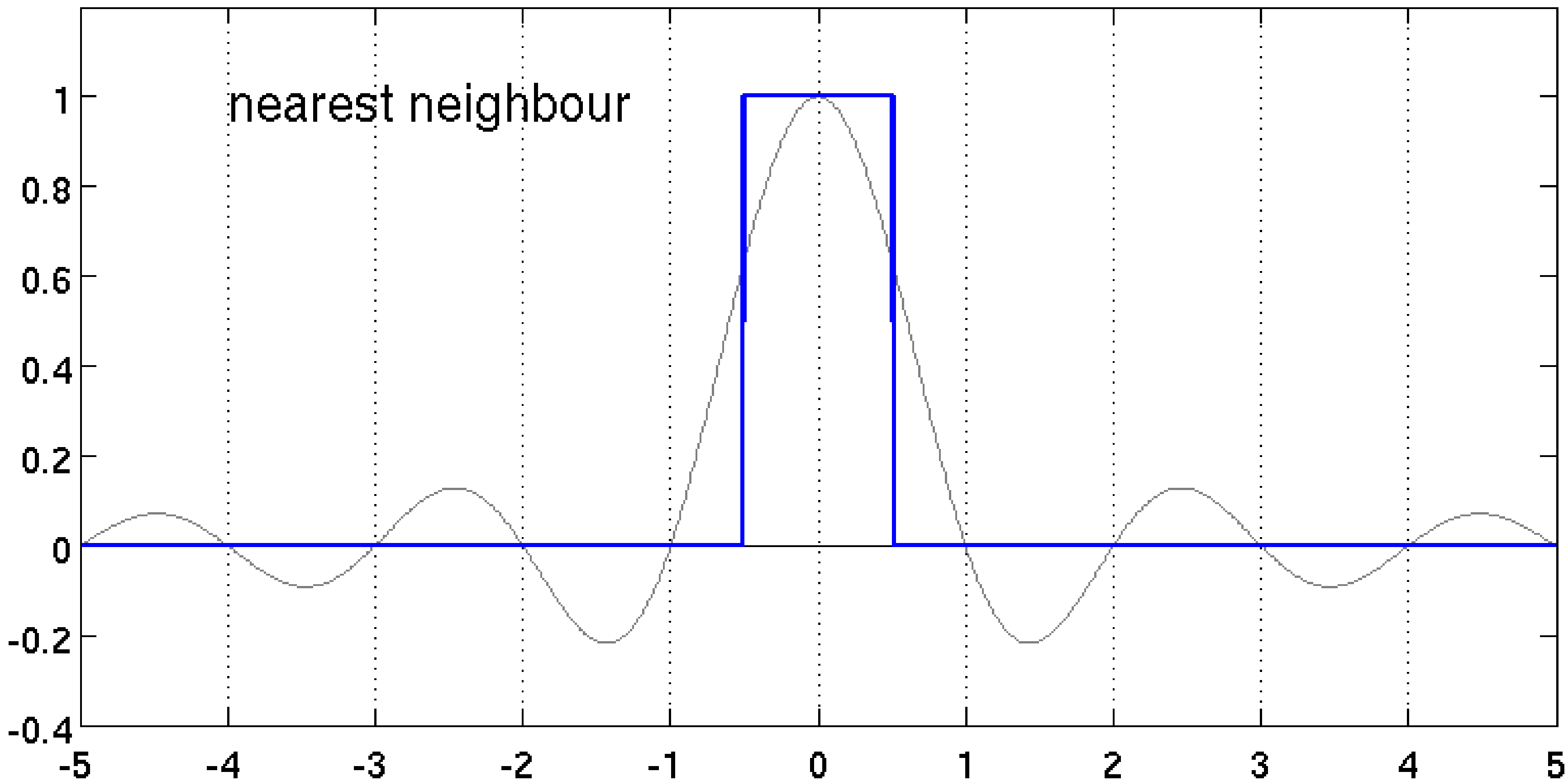
Alternative interpolation kernels



sin(πx)/πx

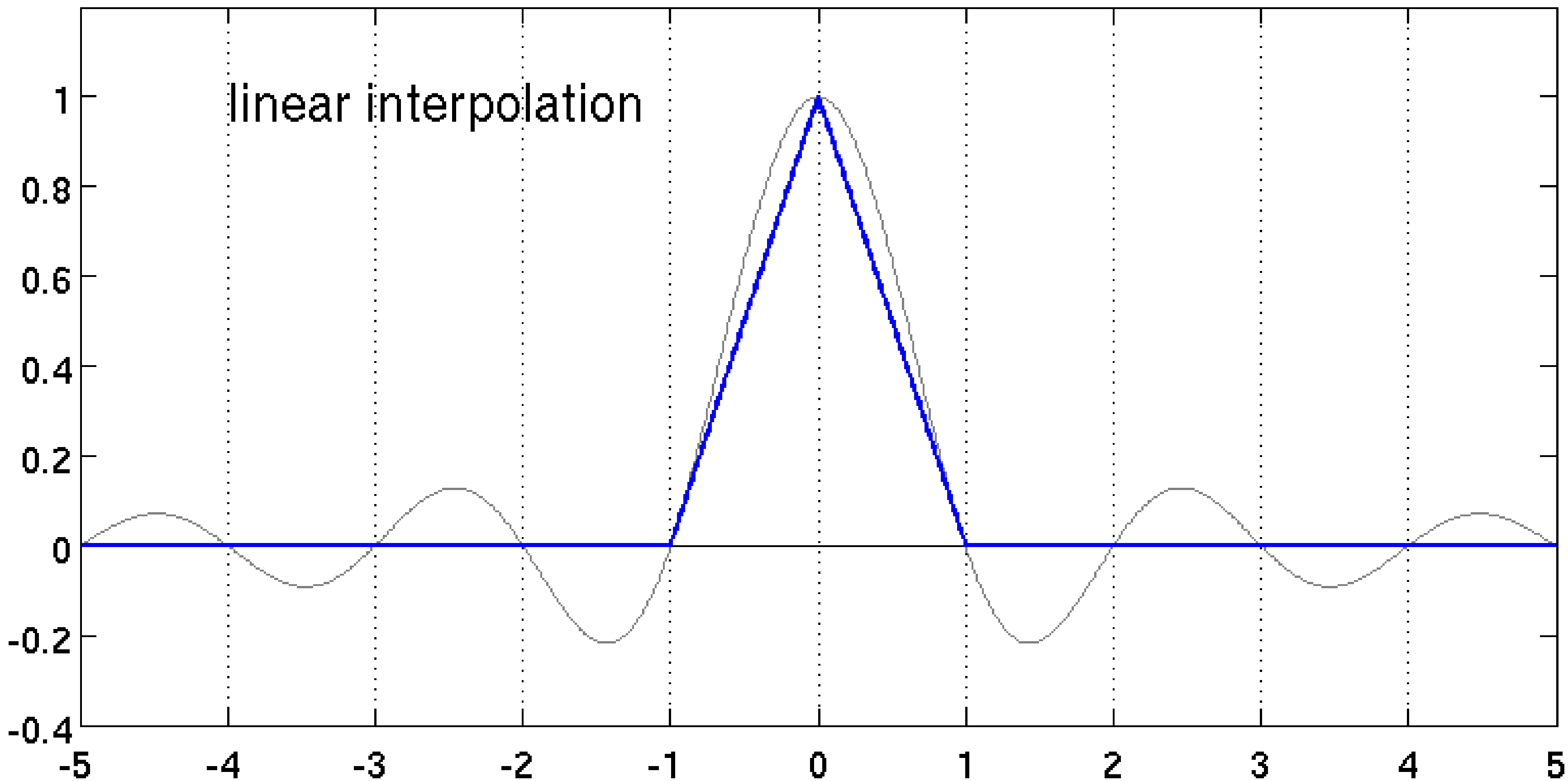
uses all data points

Alternative interpolation kernels



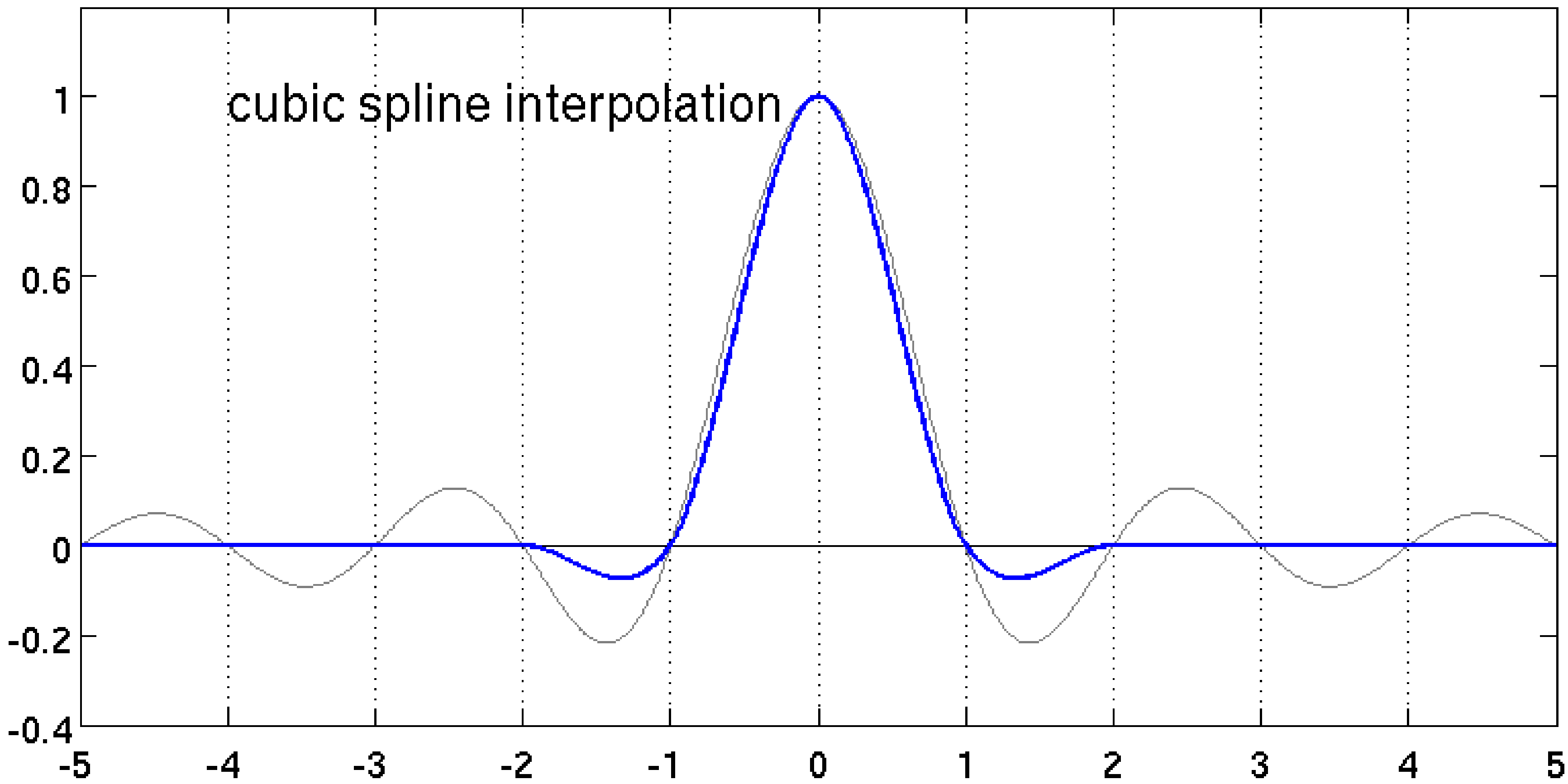
uses 1 data point

Alternative interpolation kernels



uses 2 data points

Alternative interpolation kernels

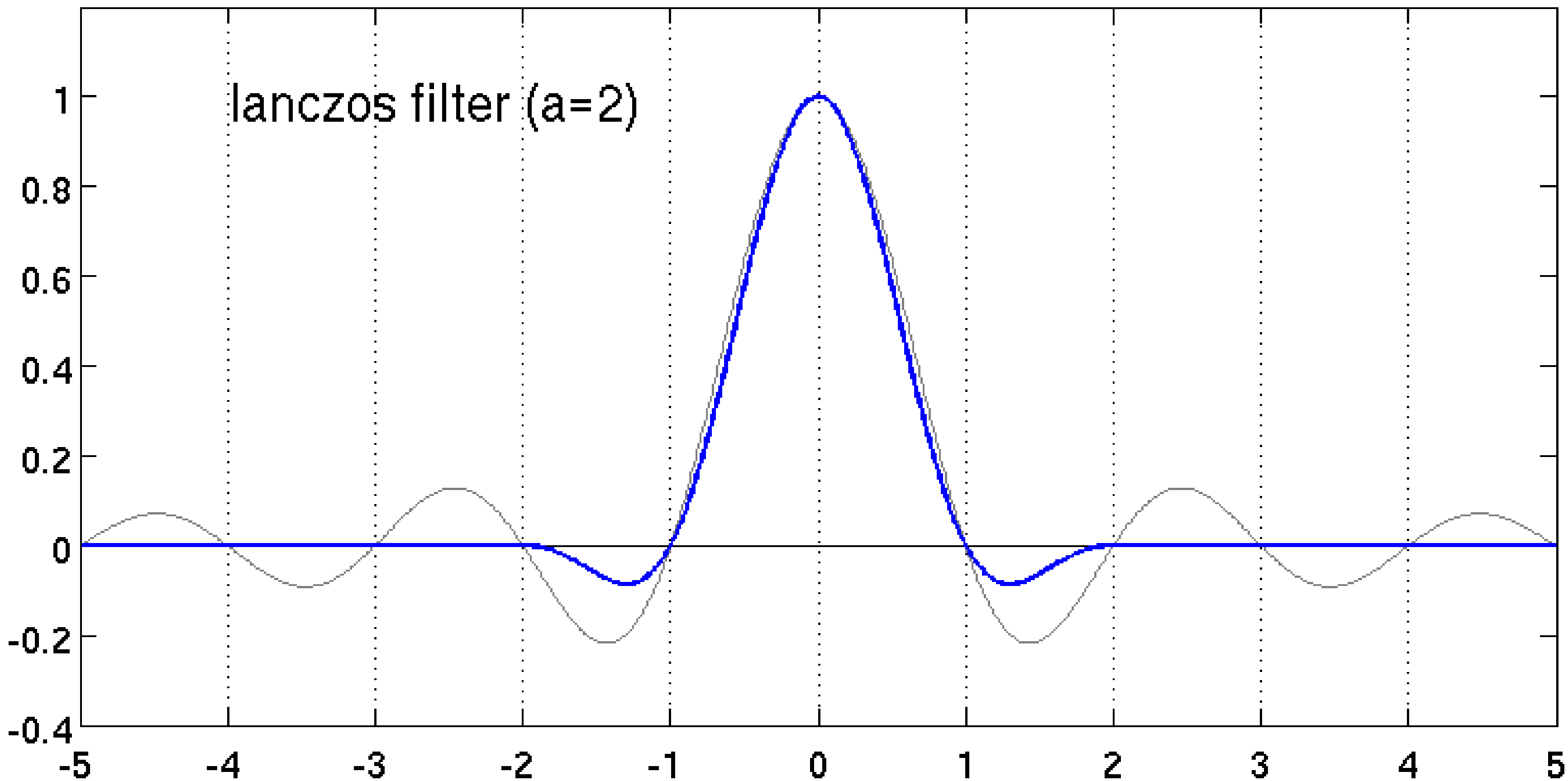


4 cubic polynomials pieced together

uses 4 data points

(don't confuse with b-splines!)

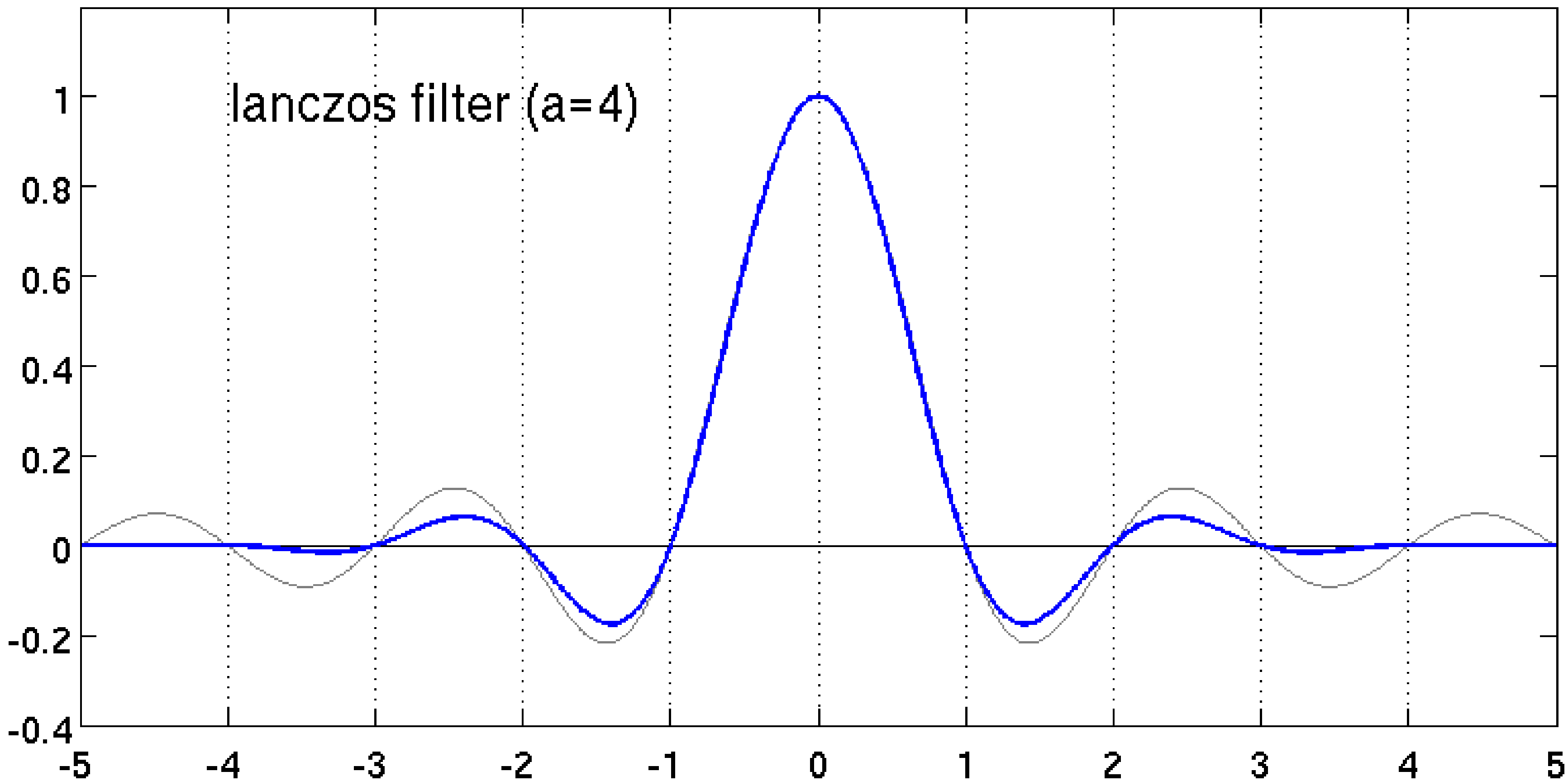
Alternative interpolation kernels



sin(πx) / (πx) * sin(πx/2) / (πx/2)

uses 4 data points

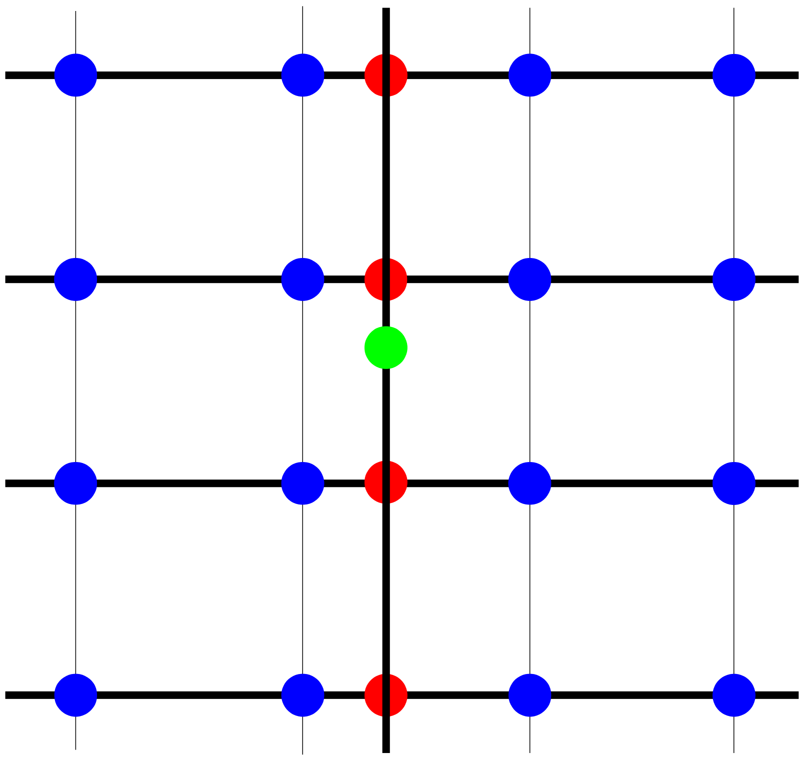
Alternative interpolation kernels



$\sin(\pi x) / (\pi x) \;*\; \sin(\pi x/4) / (\pi x/4)$ uses 8 data points

What is trilinear interpolation?

- In *n*-D images, interpolation can be done on each dimension independently:



  - “bilinear” means linear interpolation in 2-D image
  - “trilinear” means linear interpolation in 3-D image

Basic image operations

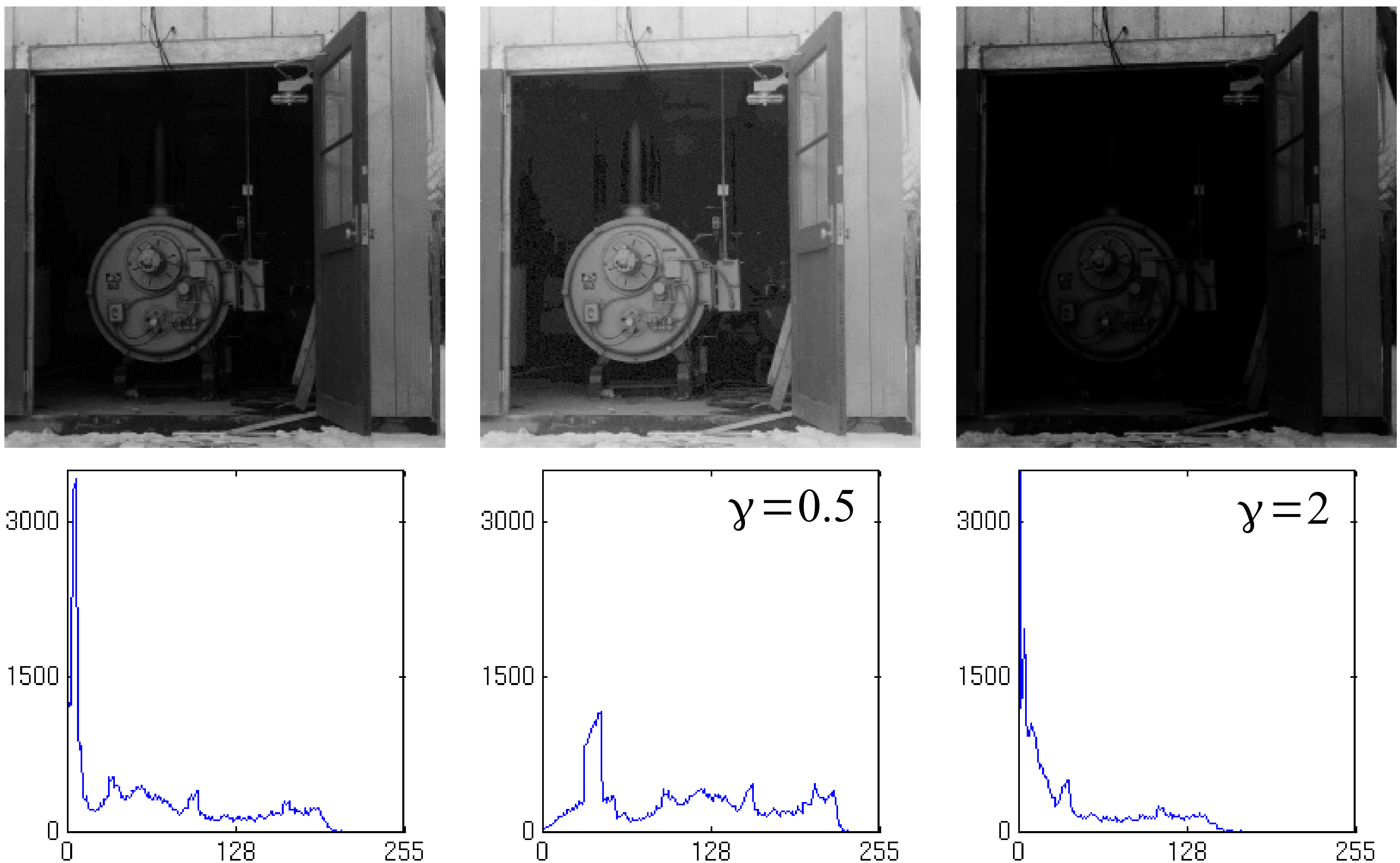
- Image arithmetic:
  - So trivial it's not even mentioned in the book
- Point operations (next):
  - Function that maps image values
  - Independent of spatial location
- Geometric transforms (Anders' lecture on Feb 4):
  - Function that maps image coordinates
  - Independent of image values
- Filtering (next 2 lectures):
  - Function that changes image values based on local neighbourhood

Point operations

- Apply a function (mapping) to each pixel in the image, independent of pixel location
  - Increase contrast
  - Bring interesting grey-value range in view
  - Make details visible
- Common:
  - Change gamma $f(y) = y^{\gamma}$
  - Contrast stretch $f(y) = ay$
  - Logarithmic stretch $f(y) = a\log(y)$
  - Clipping $f(y) = \begin{cases} a \ , & y \leqslant a \\ y \ , & otherwise \end{cases}$
  - Histogram equalization $f(y) = \langle data - dependent \rangle$
  - Thresholding $f(y) = y > a$

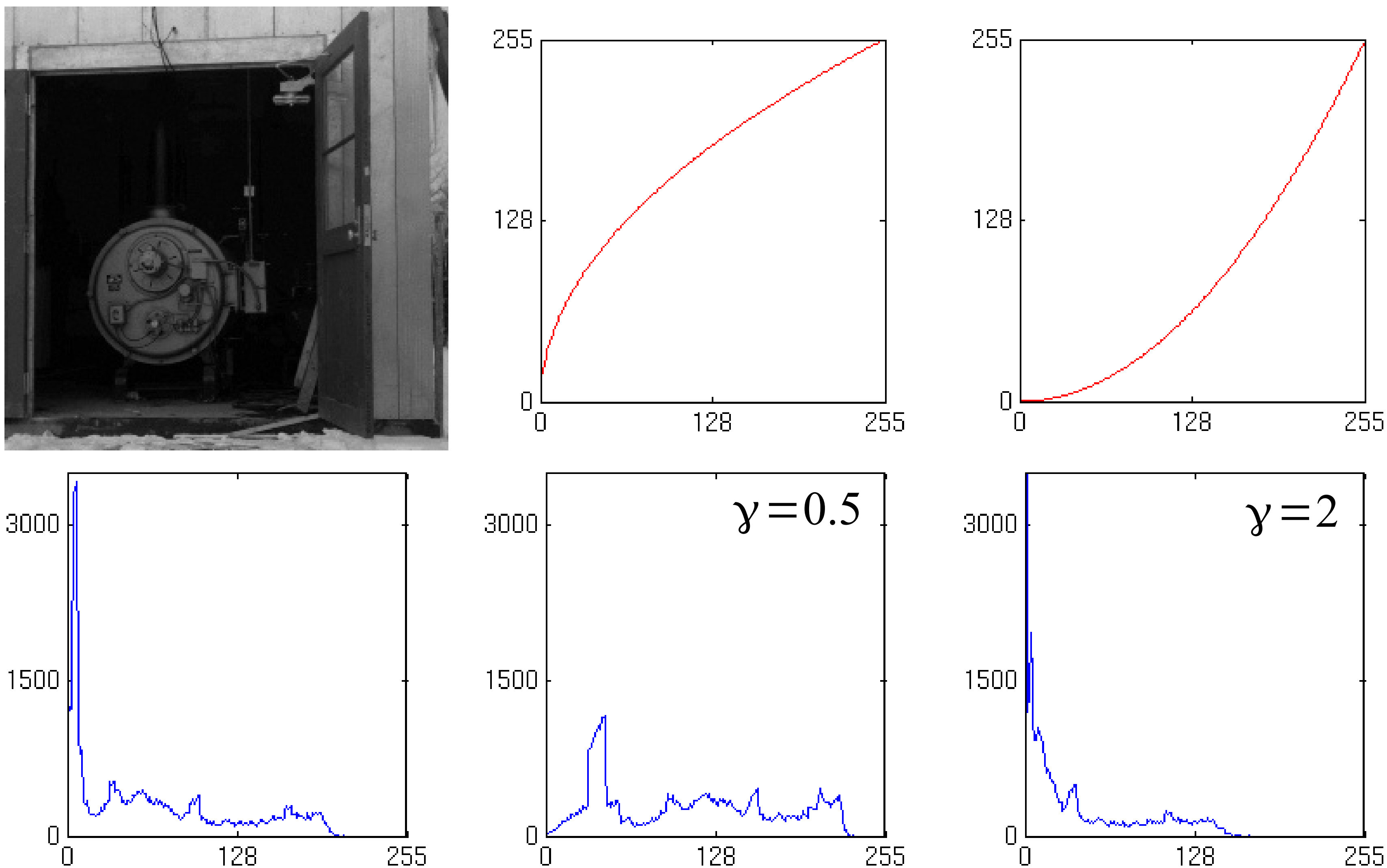
Gamma

- Increases contrast at one end of the range at the expense of the other end of the range $f(y) = y^{\gamma}$



Gamma

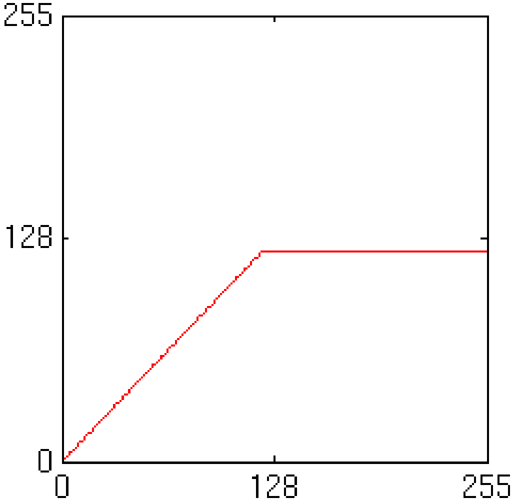
- Increases contrast at one end of the range at the expense of the other end of the range $f(y) = y^{\gamma}$

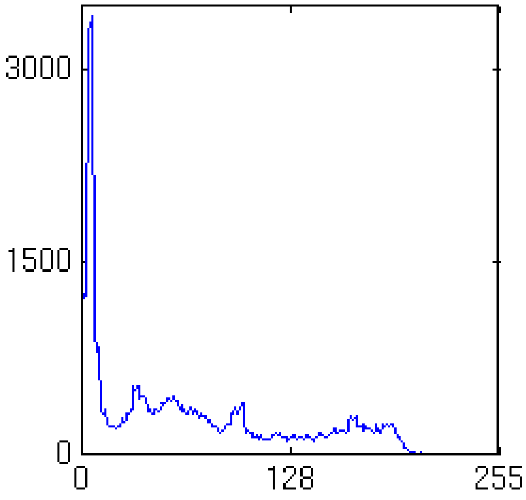


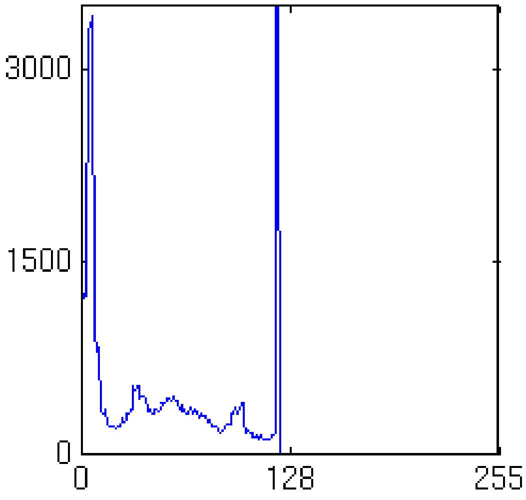
Clipping

- Brings values outside of the range to the range boundary

Mapping:

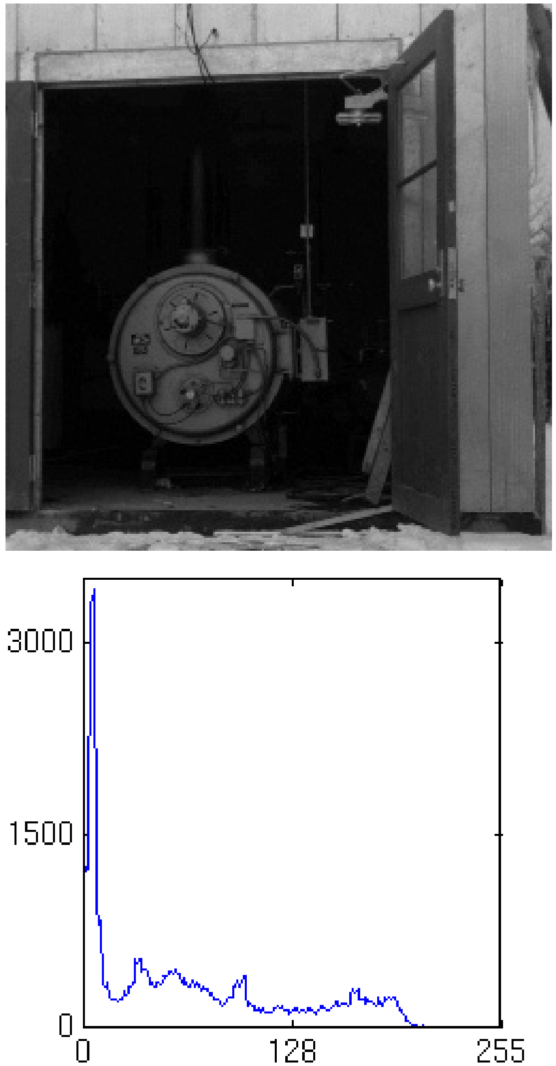


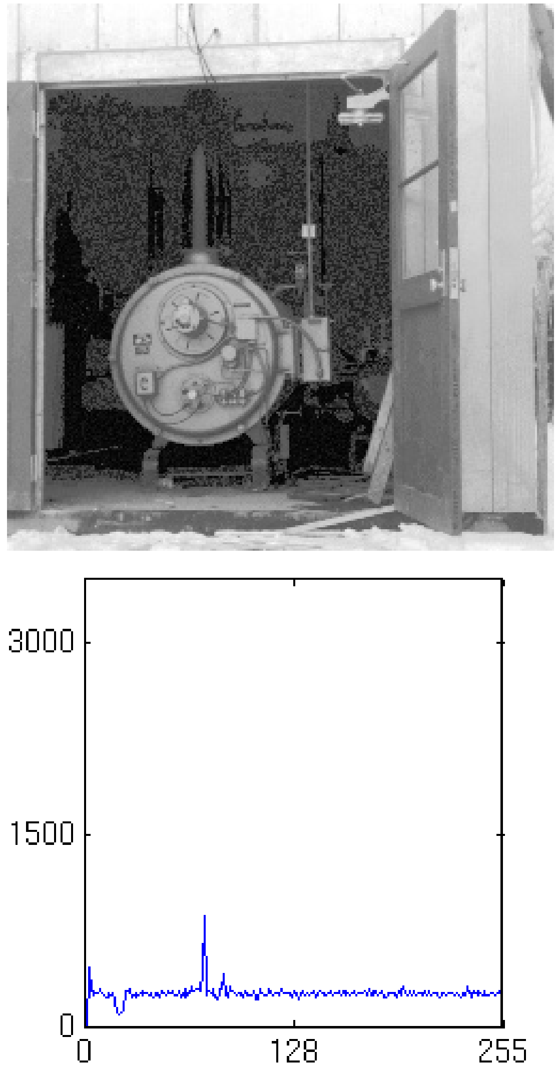


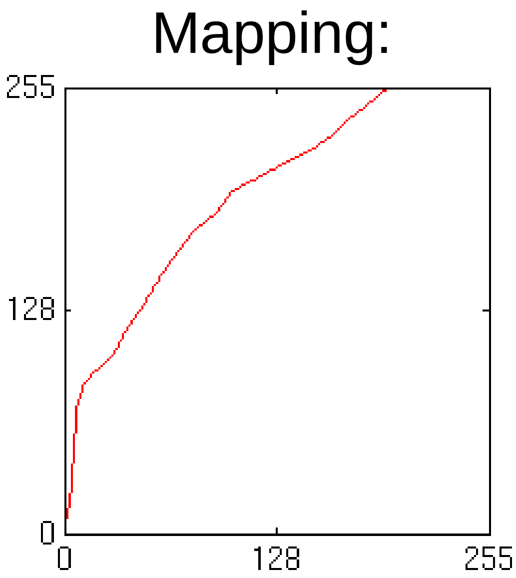


Histogram equalization

- Mapping derived from histogram: tries to make histogram as flat as possible



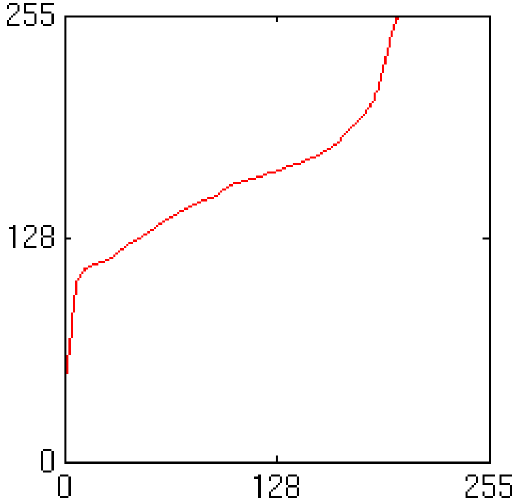


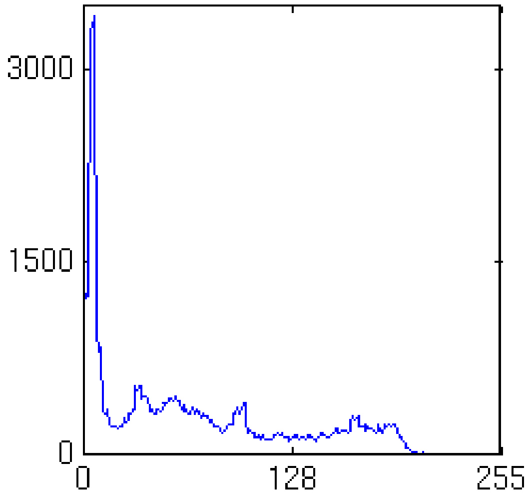


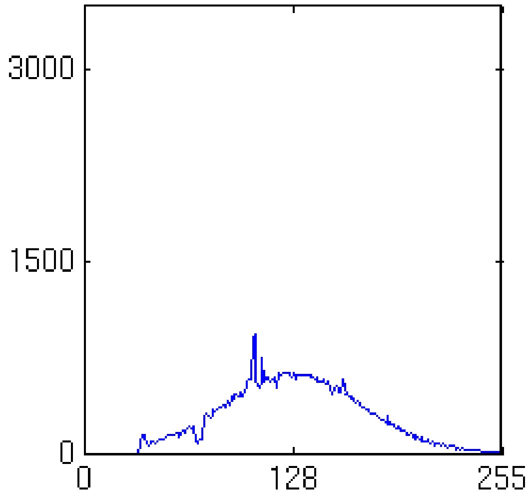
Histogram equalization

- Mapping derived from histogram: tries to make histogram as flat as possible

Mapping:







Threshold

- Simplest form of segmentation
- Associates each pixel to object or background based on the pixel's grey value
  - Static or global threshold: same threshold for all pixels

$$g(\vec{x})=\begin{cases}1 & \text{if} \quad f(\vec{x})>T \\ 0 & \text{otherwise}\end{cases}$$

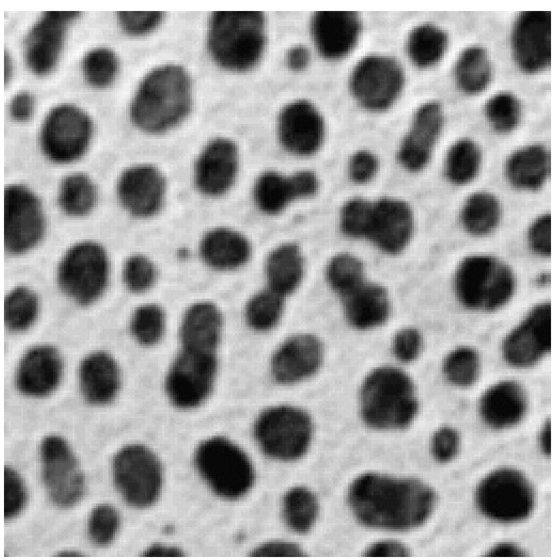
  - Adaptive or local threshold: threshold depends on local neighbourhood

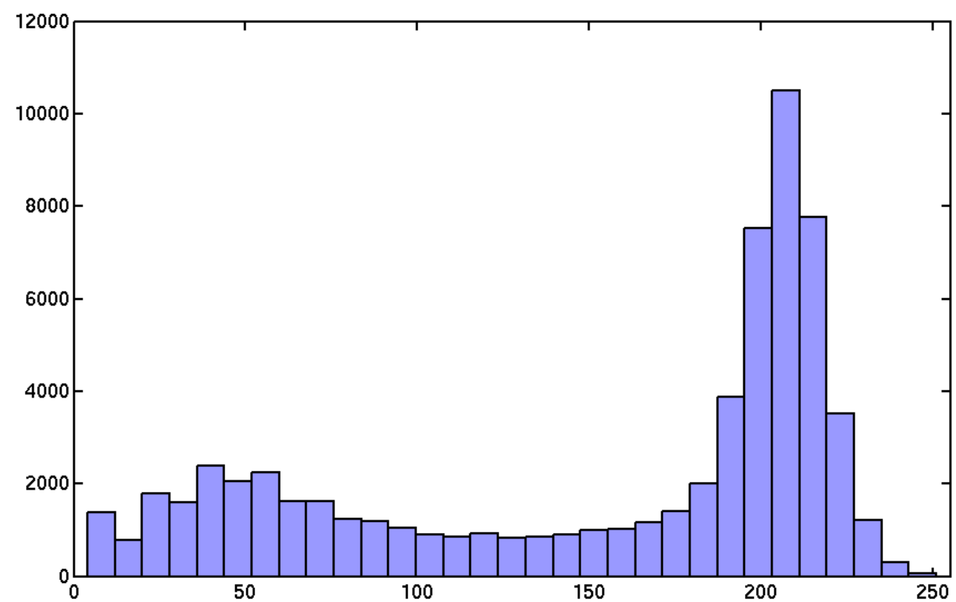
$$g(\vec{x})=\begin{cases}1 & \text{if} \quad f(\vec{x})>T(\vec{x}) \\ 0 & \text{otherwise}\end{cases}$$

  - Hysteresis threshold: combine results of two thresholds

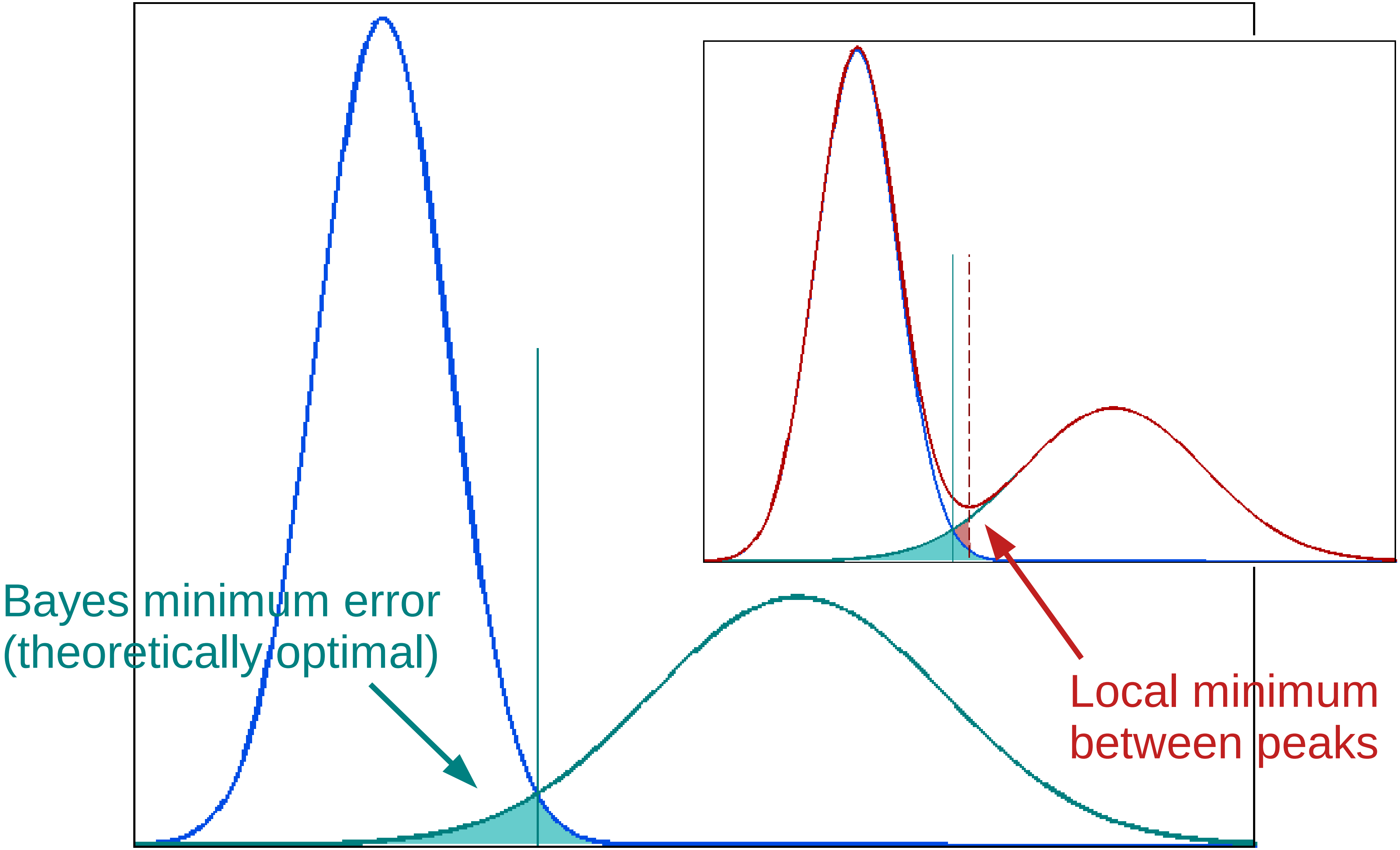
Finding a threshold level

- For a global threshold, all relevant information is in the image's histogram
  - Because no neighbourhood information is used, only each individual pixel’s grey value
  - Histogram shows distribution of grey values
  - Object and background often have separate peaks





Bimodal histograms



Bimodal histograms

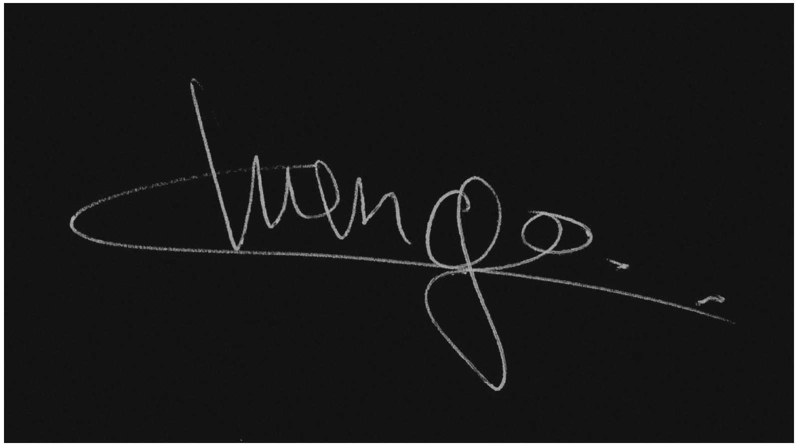
- *k*-means clustering (“isodata” method)
  - Ridler and Calvard (1978)
  - Assumes the two modes are of similar width and height
  - Iterative method, depends on initialisation
- Minimizing intra-class variance (“Otsu” method)
  - Otsu (1979)
  - Equivalent to maximizing inter-class variance (easy to compute)

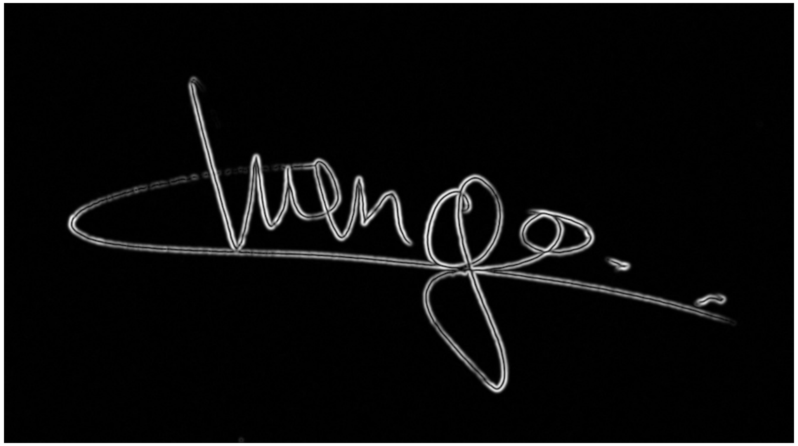
$$\sigma_b^2(t) = P_1(t) P_2(t) \left[\mu_1(t) - \mu_2(t) \right]^2$$

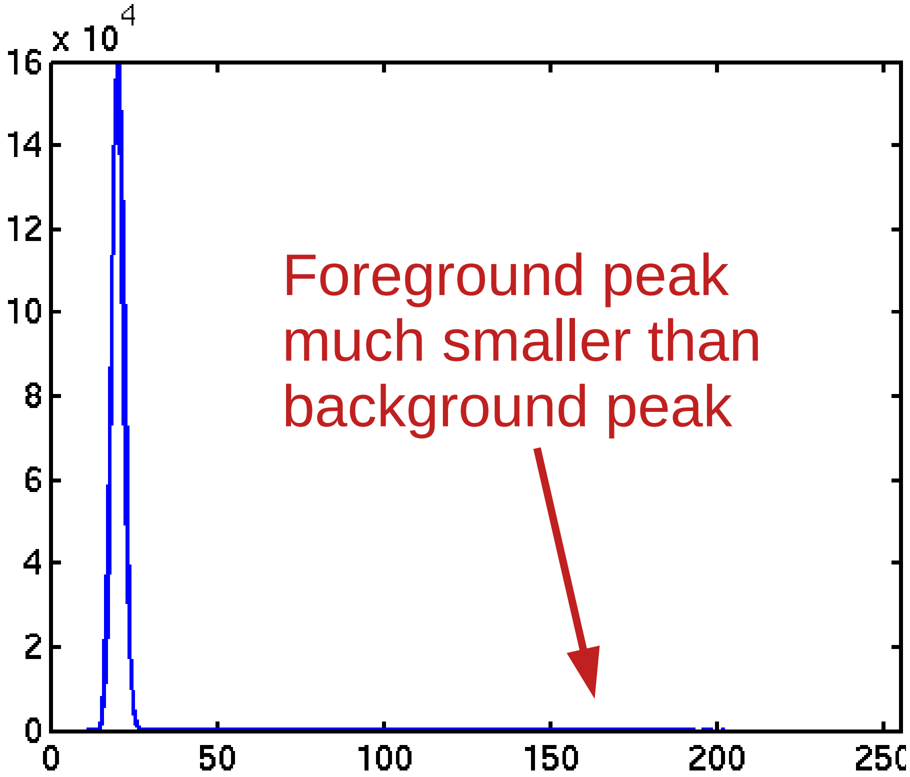
- Minimizing error
  - Kittler and Illingworth (1986)
  - Assumes 2 Normal distributions

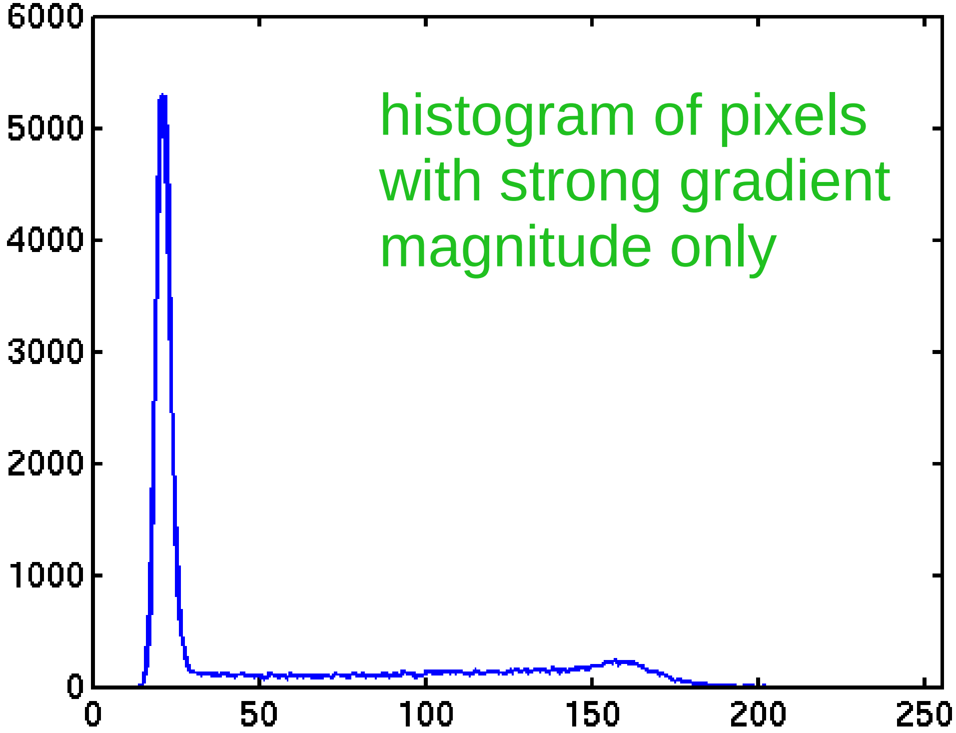
$$J(t) = 1 + 2\left[P_1(t) \log \sigma_1(t) + P_2(t) \log \sigma_2(t) \right] - 2\left[P_1(t) \log P_1(t) + P_2(t) \log P_2(t) \right]$$

Improving the histogram

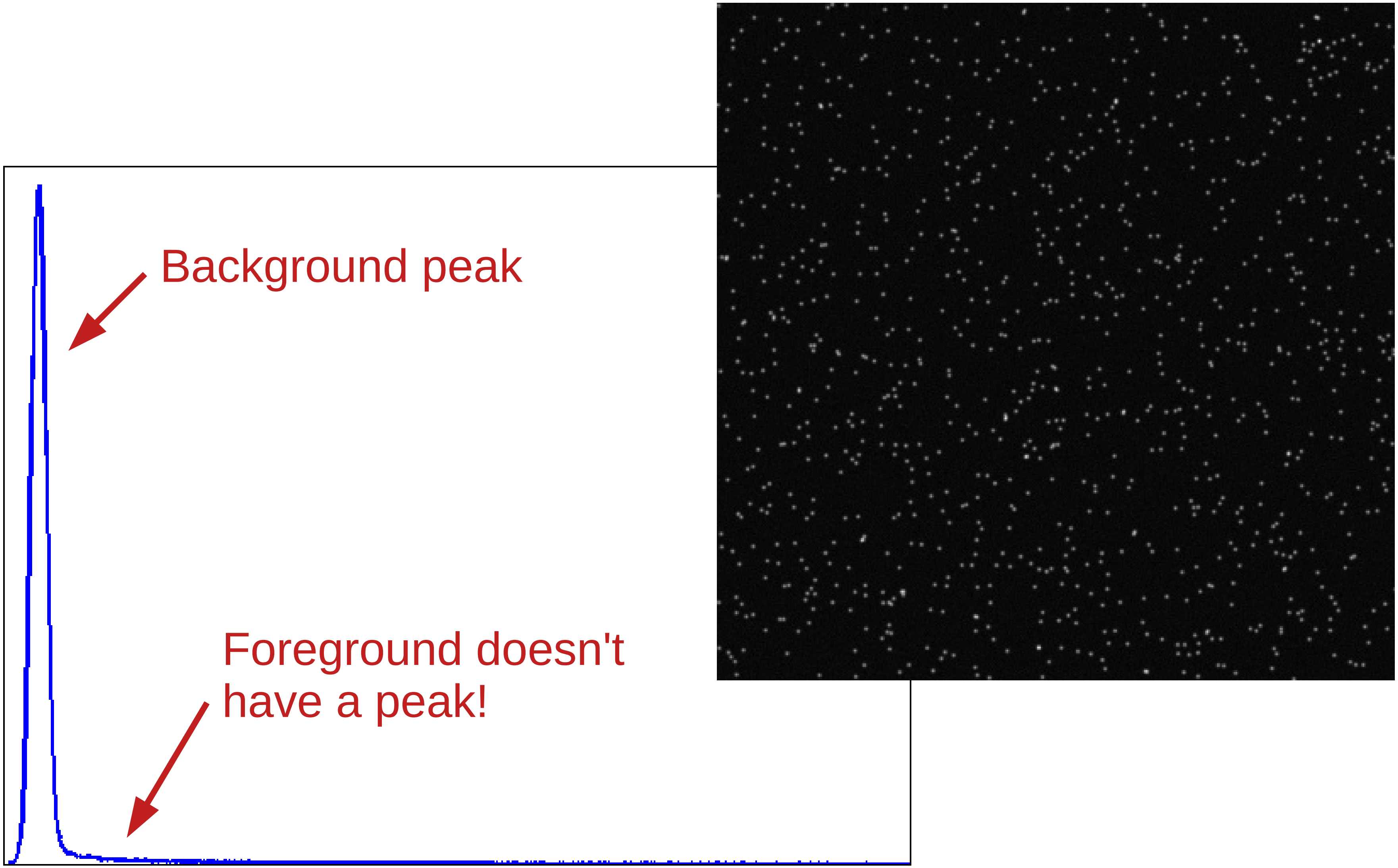






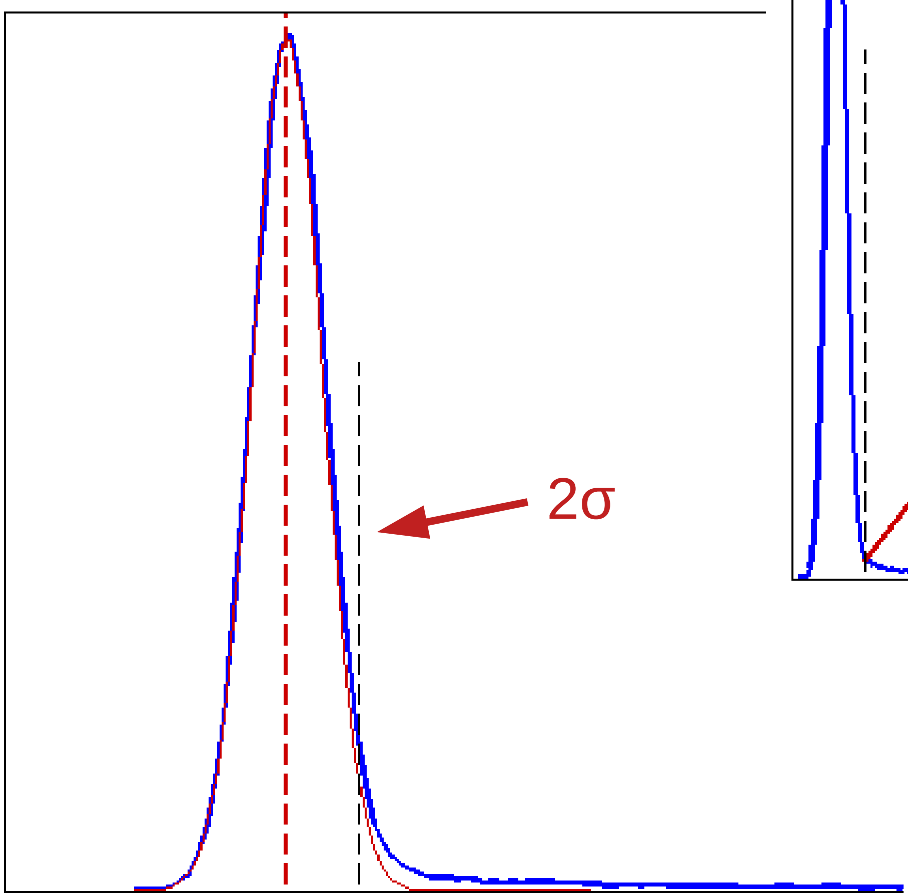


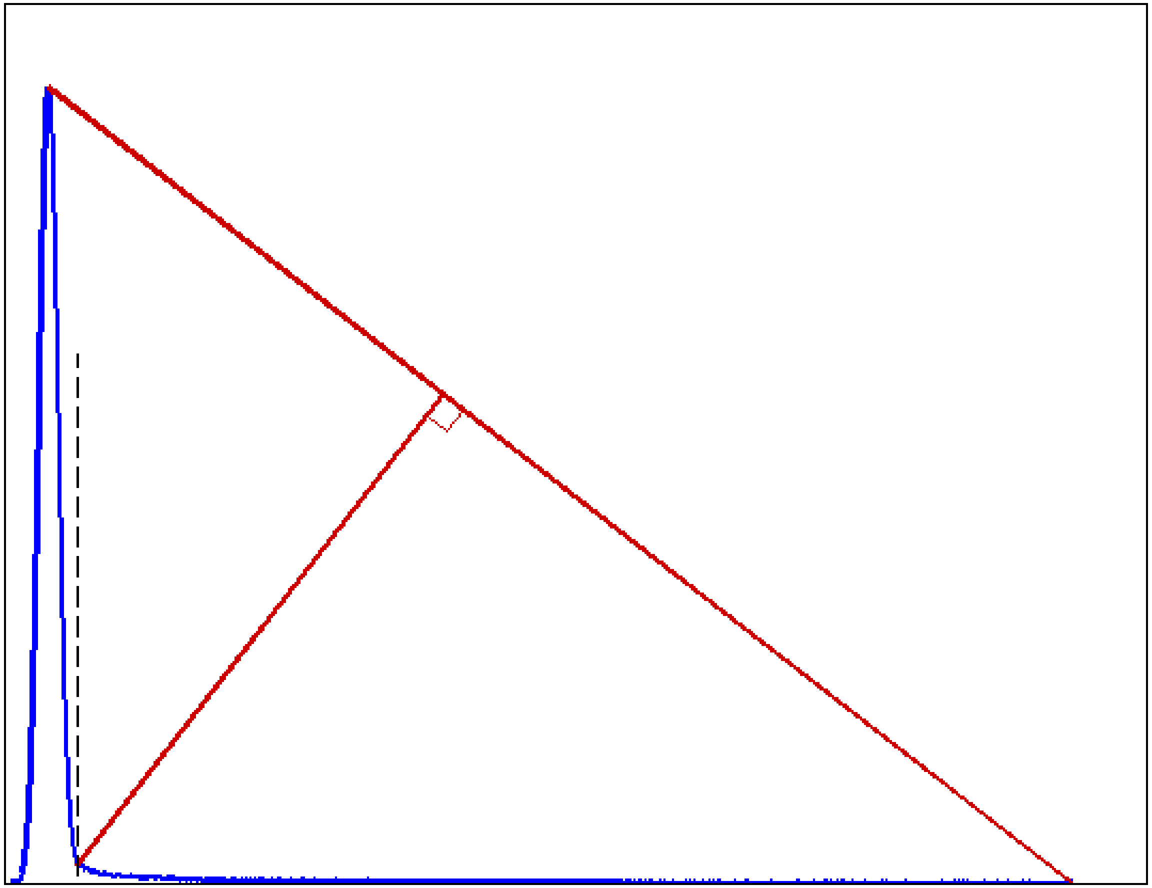
Unimodal histograms



Unimodal histograms

Fitting a Gaussian to the background peak





Chord method, a.k.a.:
- skewed bi-modality
- maximum distance to triangle

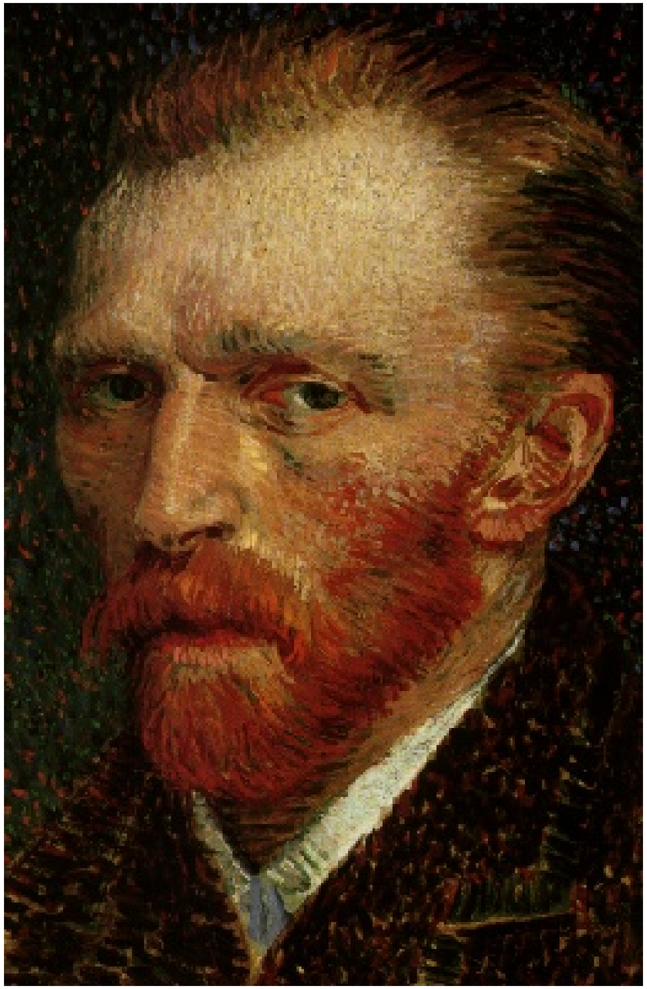
Zack, Rogers and Latt (1977)
Rosin (2001)

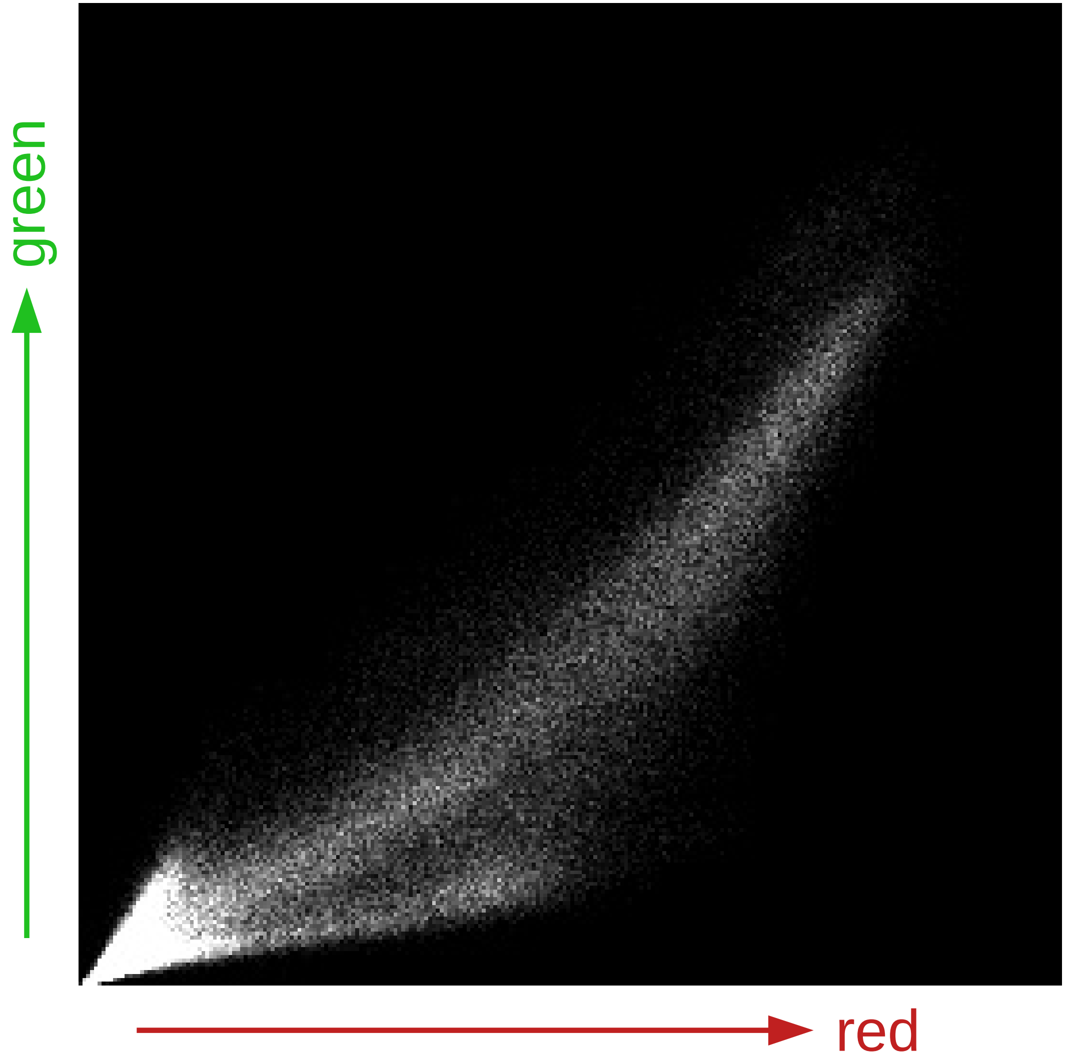
Finding a threshold level

Other methods used besides histogram analysis:

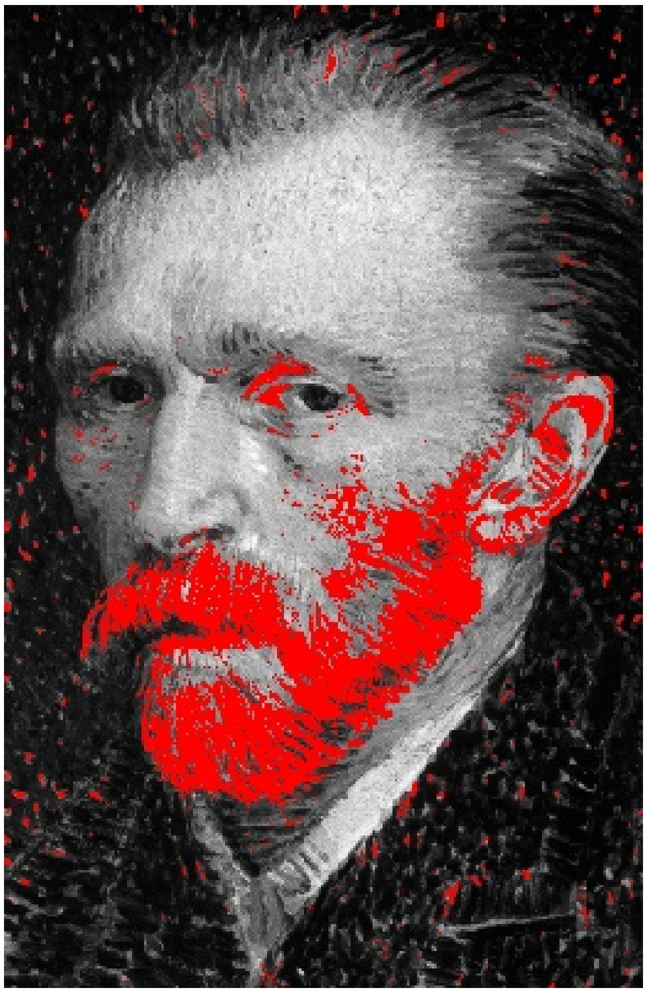
- Manual determination on a training set of images
  - Threshold becomes a "magic number"
  - Results useless if imaging circumstances change
  - E.g. in CT the grey-value is an absolute measure
- Using *a priori* knowledge:
  - Volume: if it is known that 25% of the image is foreground, choose a threshold value so that 25% of the pixels are above it
  - Shape: if round objects are expected, choose a threshold value that maximizes some roundness measure of the result
  - …

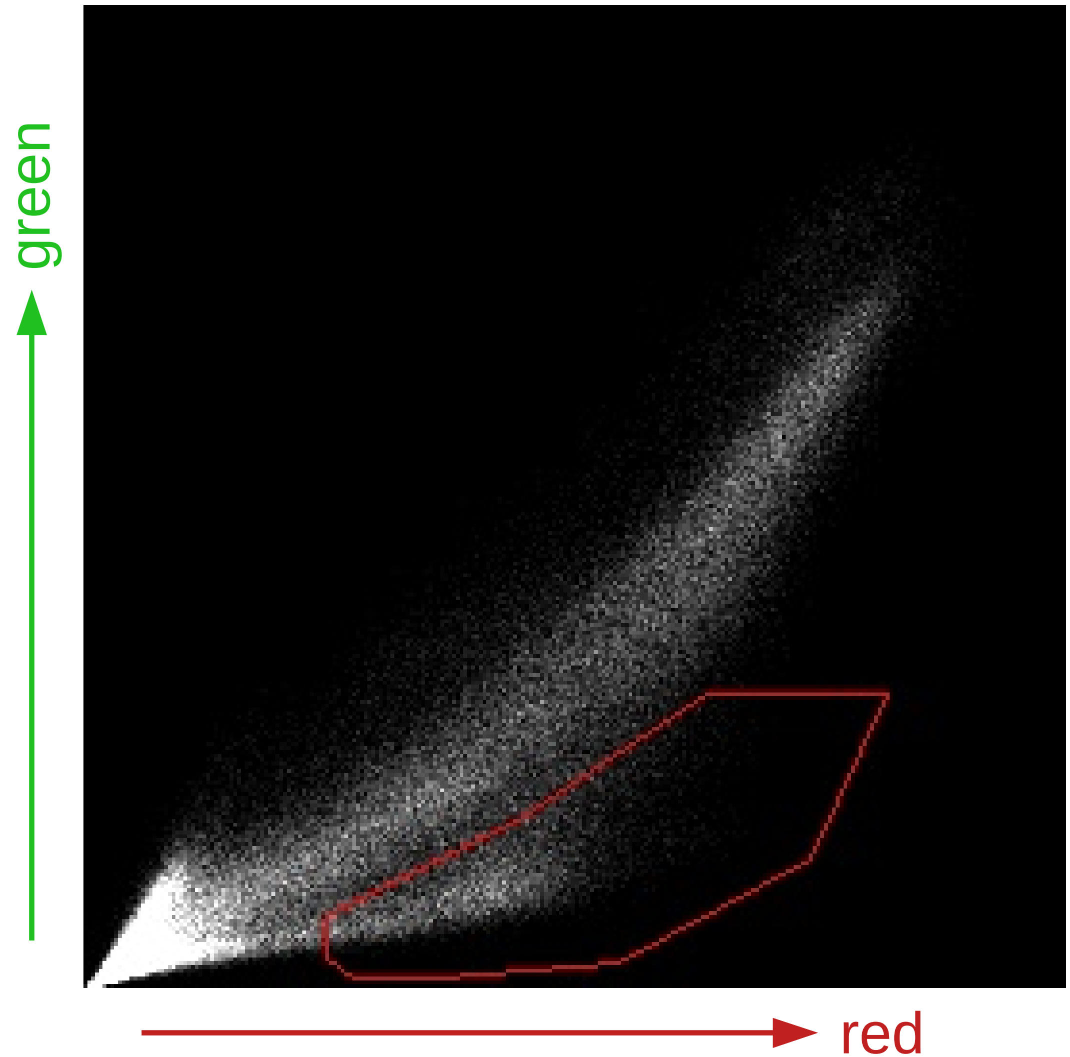
Multi-channel threshold





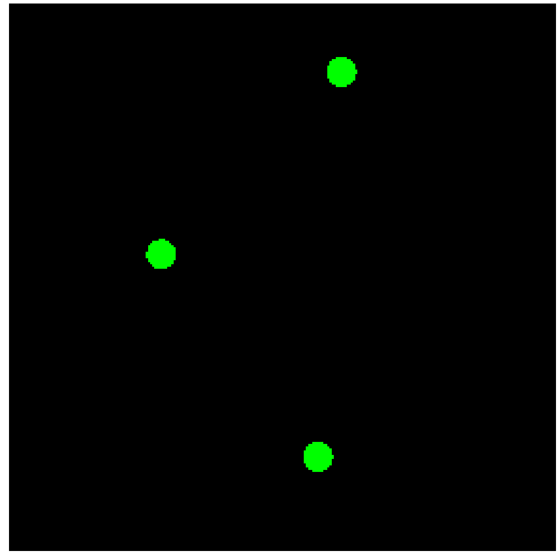
Multi-channel threshold

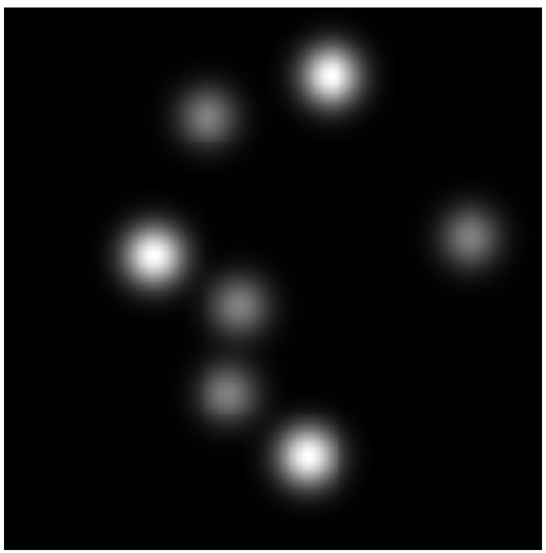


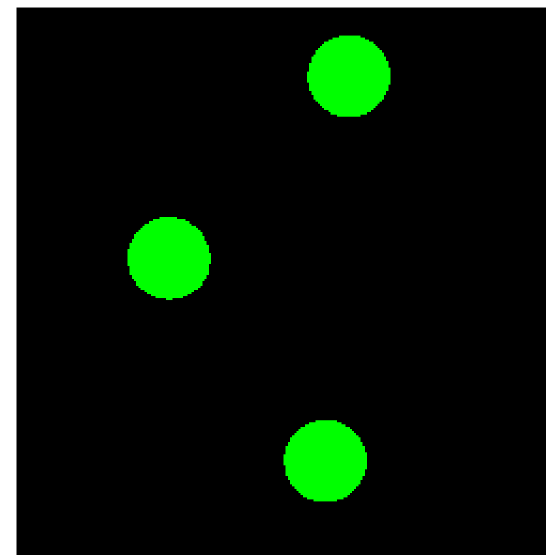


Hysteresis threshold

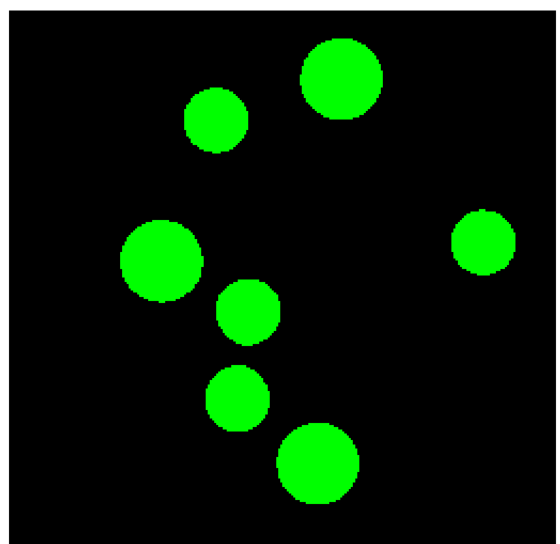
High threshold







Regions in “low” that have some pixels set in “high”



Low threshold

Summary of today's lecture

- Convolution is important!
- Sampling property of convolution
- Fourier Domain useful for understanding convolution
- Convolution in Spatial Domain is multiplication in Fourier Domain

$$f(x) \otimes h(x) \overset{\mathscr{F}}{\Rightarrow} F(\omega)H(\omega)$$

- Images can be interpolated with a convolution
- New threshold finding techniques