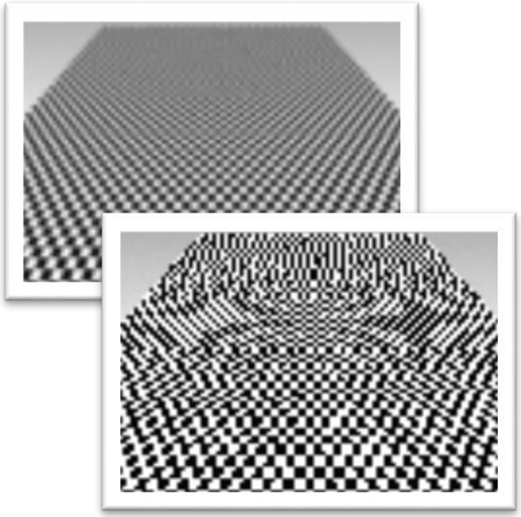
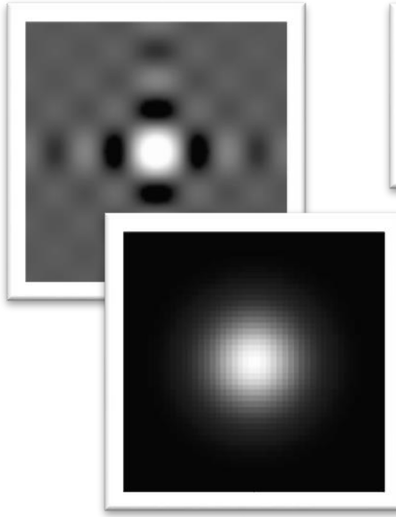
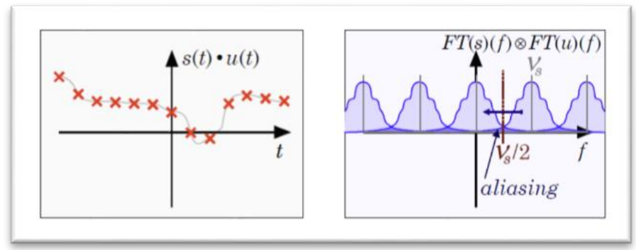
Modelling 1

SUMMER TERM 2020





$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \omega} dx$$

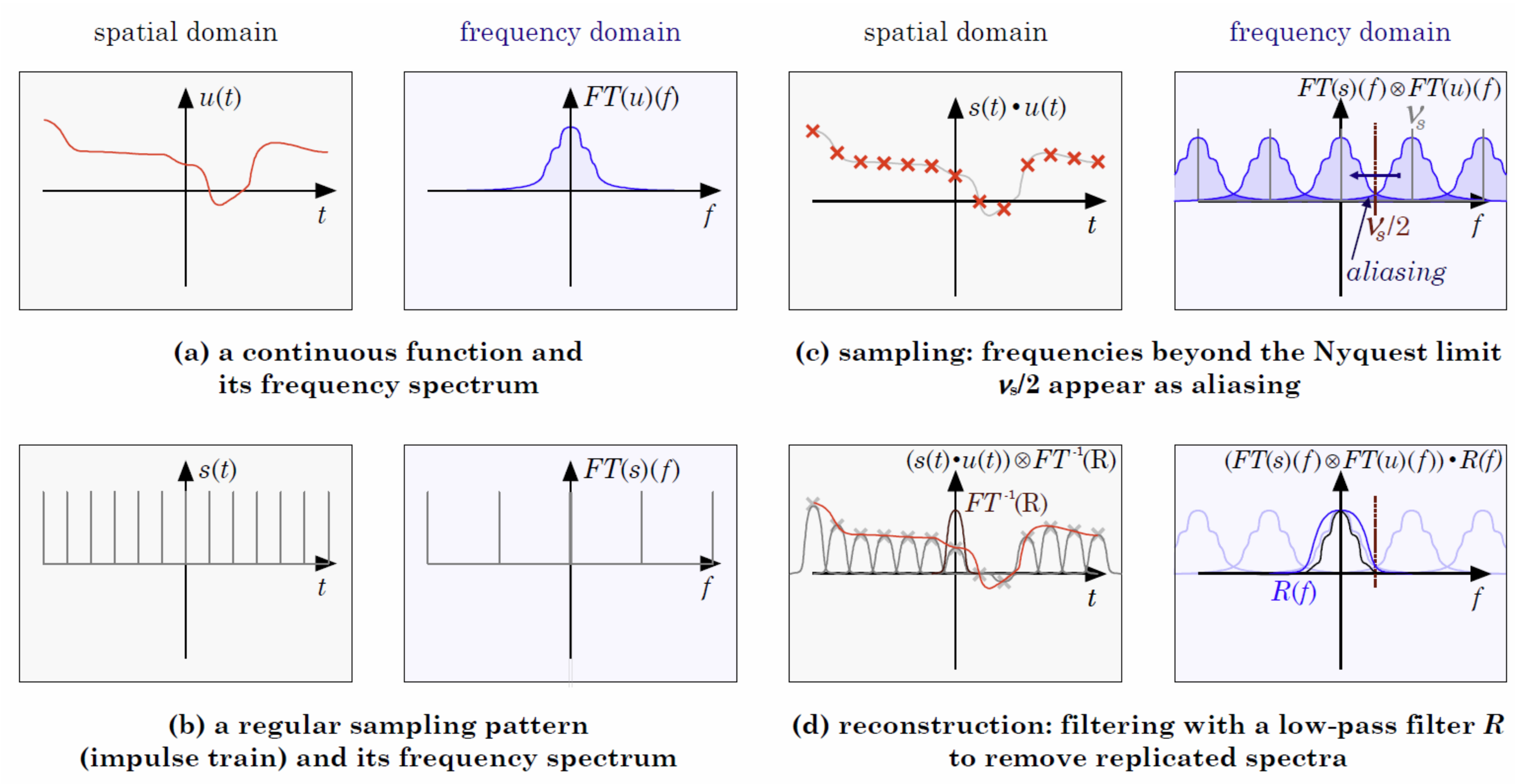


LECTURE 19

Regular and Irregular Sampling

Michael Wand · Institut für Informatik · Michael.Wand@uni-mainz.de

Sampling & Reconstruction



(a) a continuous function and its frequency spectrum

(c) sampling: frequencies beyond the Nyquest limit $\nu_s/2$ appear as aliasing

(b) a regular sampling pattern (impulse train) and its frequency spectrum

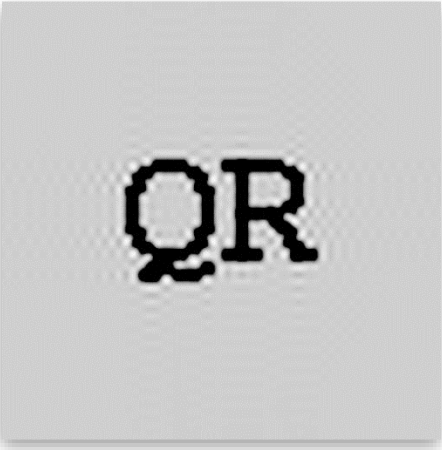
(d) reconstruction: filtering with a low-pass filter R to remove replicated spectra

Reference: Foley, van Dam, Feiner, Hughes
Computer Graphics - Principles & Practice, 2nd Edition, Addisson-Wesley, 1996
Chapter 14.10 “Aliasing and Antialiasing”

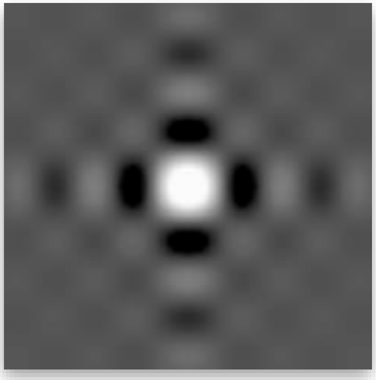
Regular Sampling

Reconstruction Filters

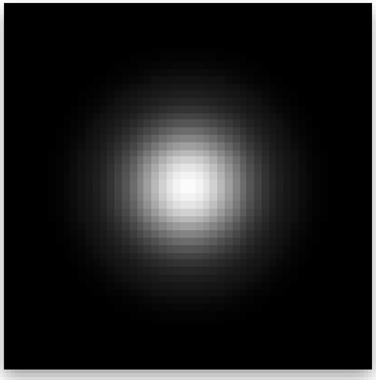
- Optimal filter: sinc
  (no frequencies discarded)
- However:
  - Ringing artifacts in spatial domain
  - Not useful for images
    (better for audio)
- Compromise
  - Gaussian filter
    (most frequently used)
  - There exist better ones,
    such as Mitchell-Netravalli,
    Lancos, etc...



Ringing by sinc reconstruction from [Mitchell & Netravali, Siggraph 1988]



2D sinc



2D Gaussian

Irregular Sampling

Irregular Sampling

Irregular Sampling

- No comparable formal theory
- However: similar idea
  - Band-limited by “sampling frequency”
  - Sampling frequency = mean sample spacing
    - Not as clearly defined as in regular grids
    - May vary locally (adaptive sampling)
- Aliasing
  - Random sampling creates noise as aliasing artifacts
  - Evenly distributed sample concentrate noise in higher frequency bands in comparison to purely random sampling

Consequences

When designing bases for function spaces

- Use band-limited functions
- Typical scenario:
  - Regular grid with spacing $\boldsymbol{\sigma}$
  - Grid points $\mathbf{g}_i$
  - Use functions: $\mathbf{exp}\left(-\frac{(\mathbf{x}-\mathbf{g}_i)^2}{\sigma^2}\right)$
- Irregular sampling:
  - Same idea
  - Use estimated sample spacing instead of grid width
  - Set $\boldsymbol{\sigma}$ to average sample spacing to neighbors

Random Sampling

Random sampling

- Aliasing gets replaced by noise
- Can we optimize this? – Yes!

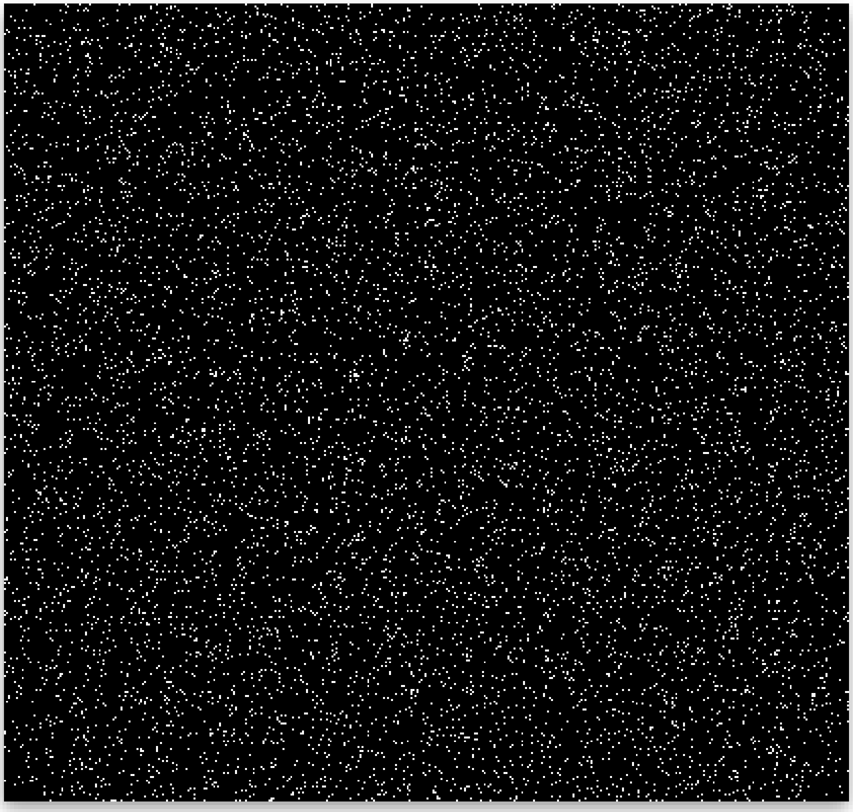
Different types of noise

- “White noise”: All frequencies equally likely
- “Blue noise”: Pronounced high-frequency content

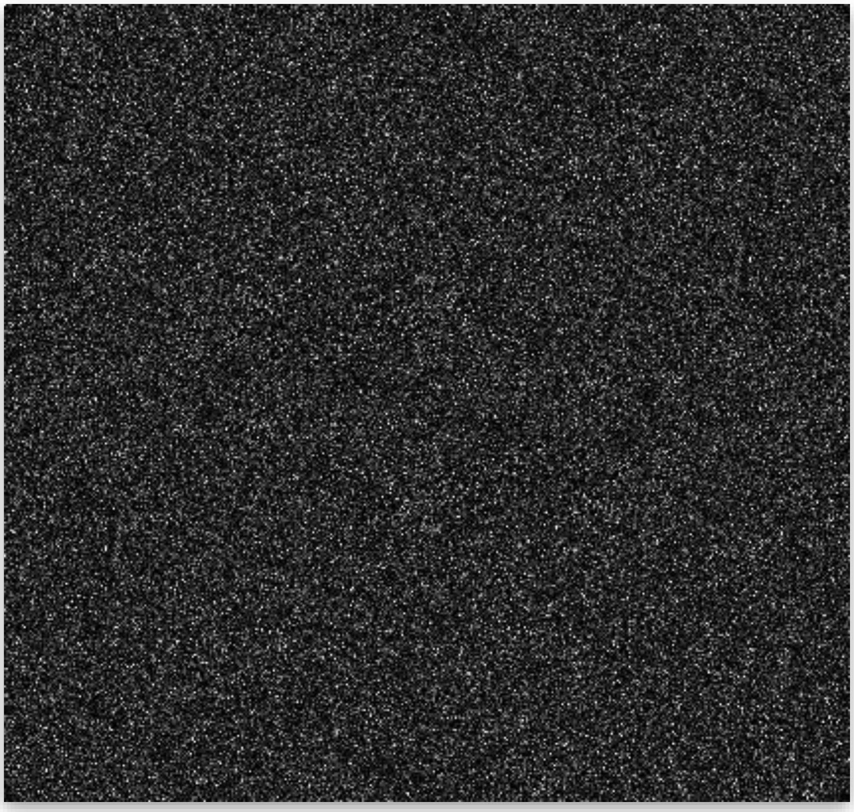
Depends on sampling

- Random sampling is “white”
- Poisson-disc sampling (uniform spacing) is “blue”

Random Noise

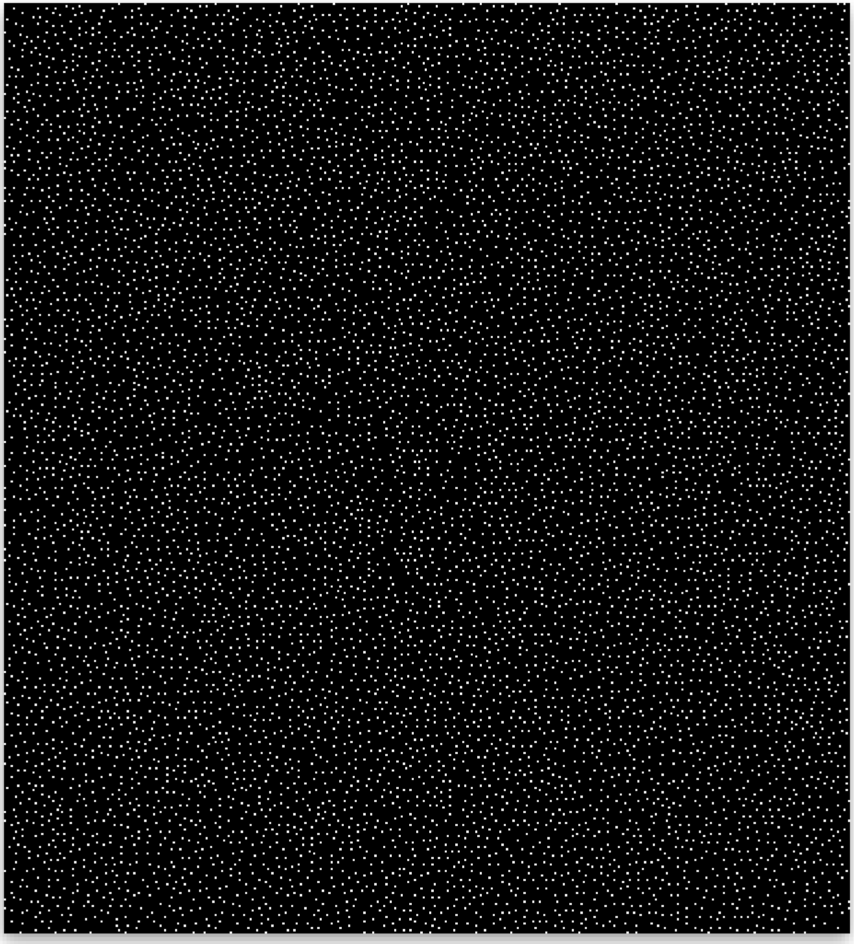


pixel image (b/w)

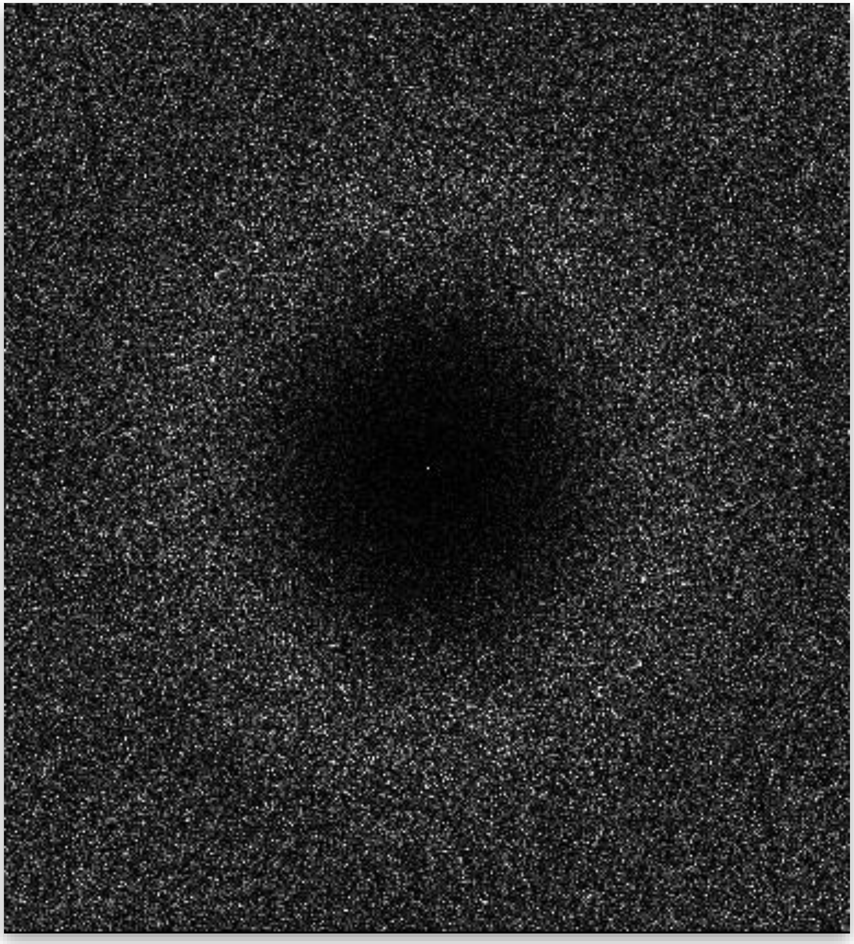


discrete Fourier transform
(power-spectrum)

Poisson Disc Sampling

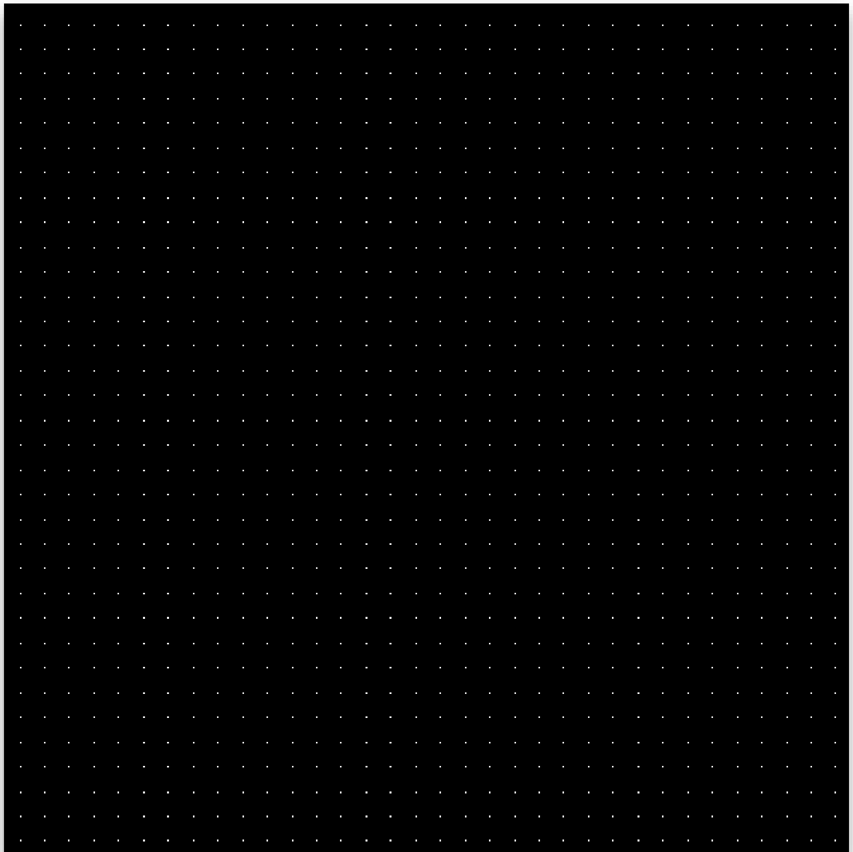


pixel image (b/w)

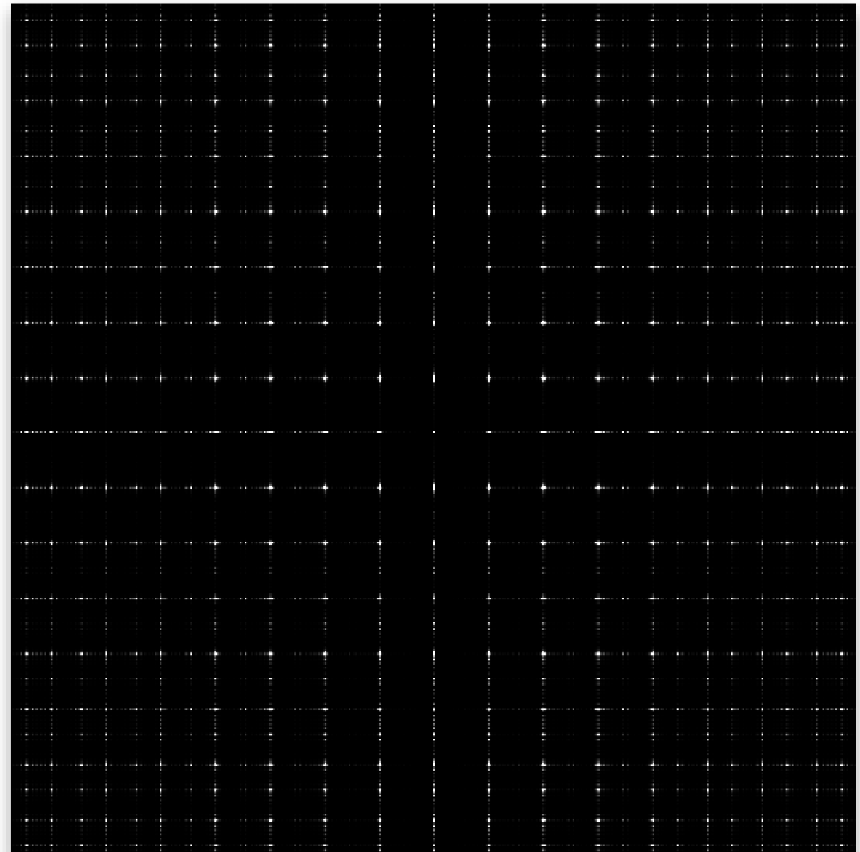


discrete Fourier transform (power-spectrum)

Regular Sampling



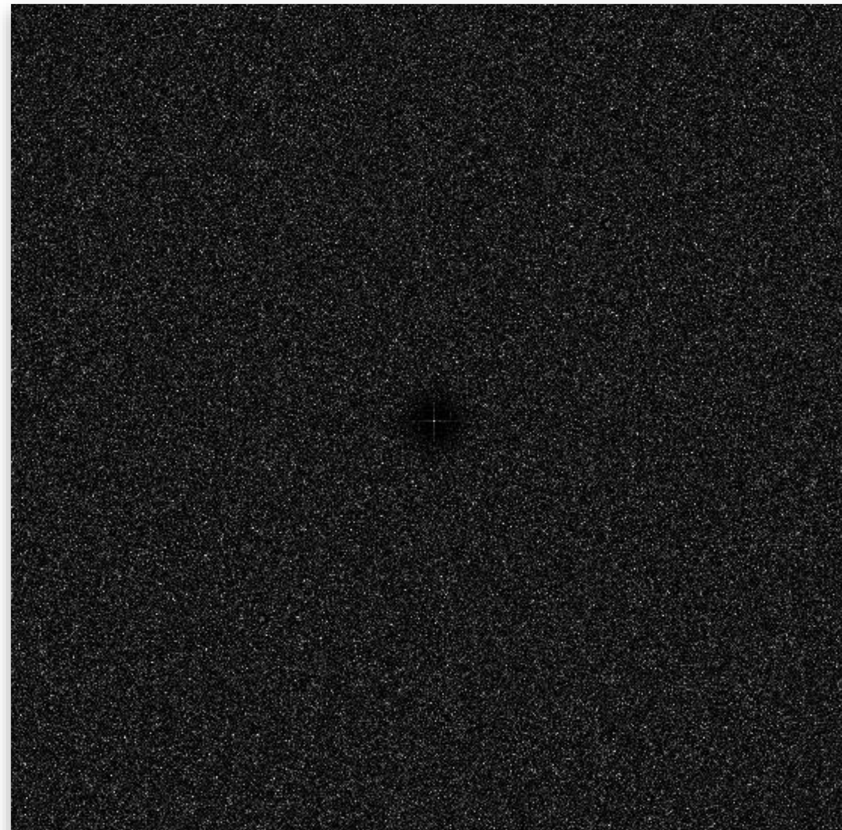
pixel image (b/w)



discrete Fourier transform
(power-spectrum)

Jittered Grid (Uniform Displacem.)

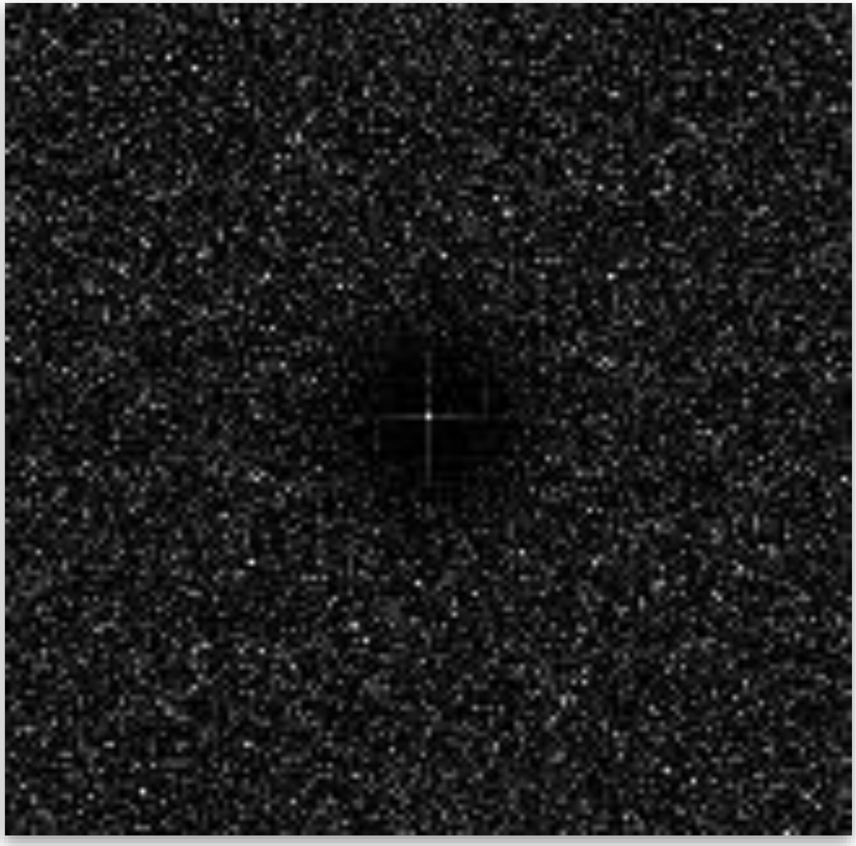
pixel image (b/w)



discrete Fourier transform
(power-spectrum)

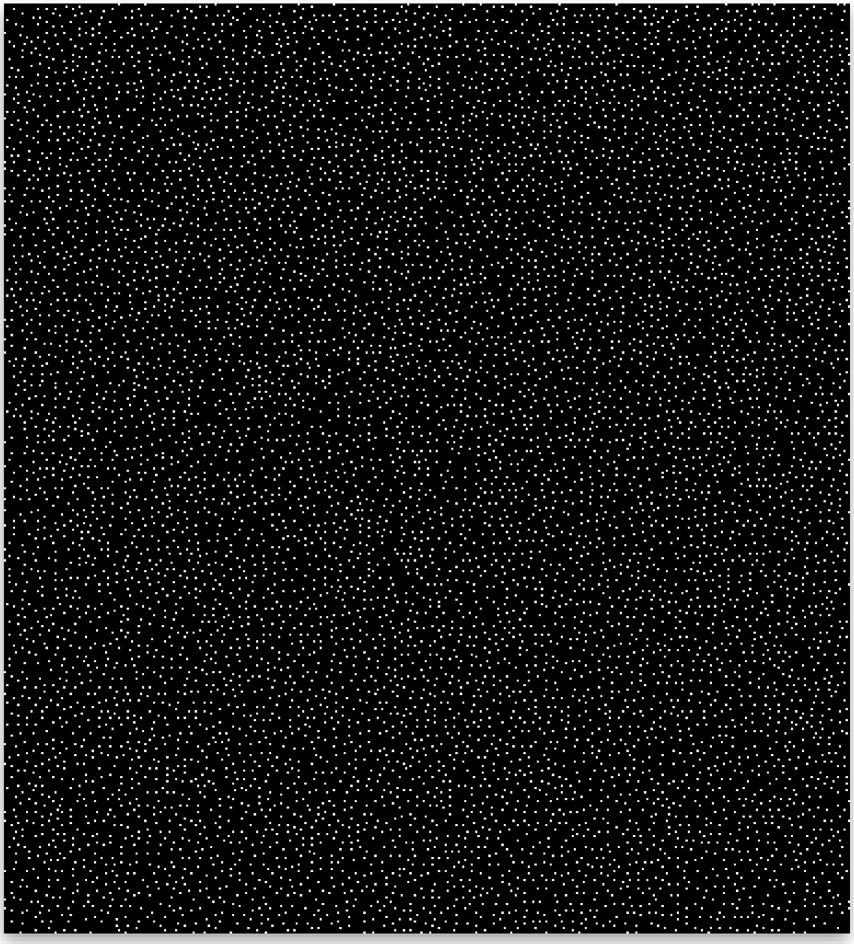
Jittered Grid (same density)

pixel image (b/w)

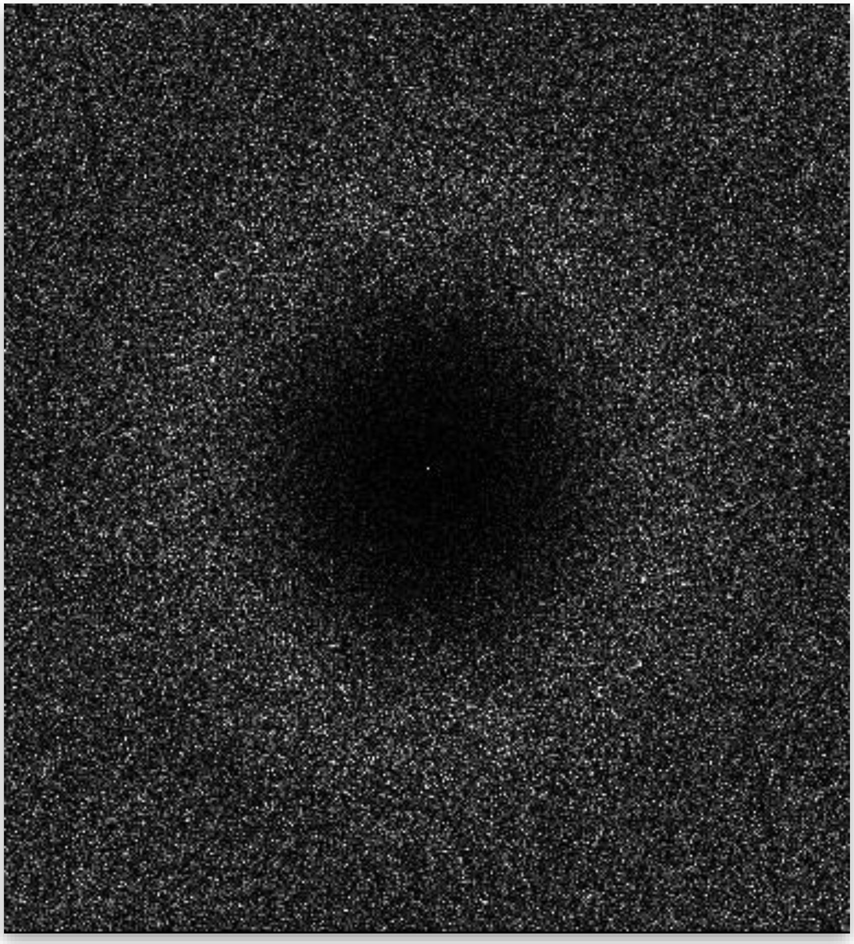


discrete Fourier transform
(power-spectrum)

Examples

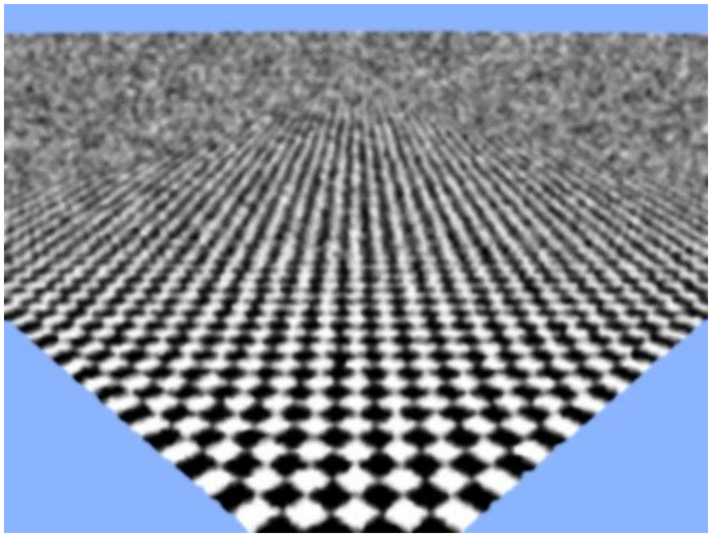


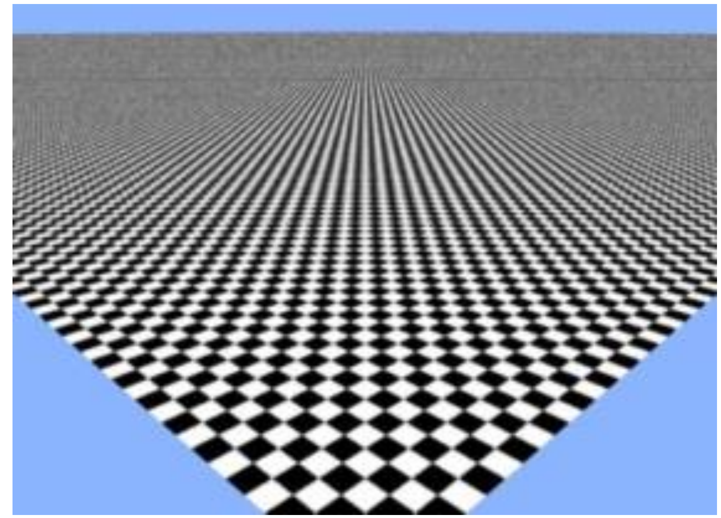
pixel image (b/w)



discrete Fourier transform
(power-spectrum)

Why should we care?





Example: Stochastic Raytracing

- Shoot random rays → random noise
- Low-pass filter → less noise
  - Low-frequency noise persists
  - LF-noise is particularly ugly!
  - Need many samples

Recipe:

Sampling Signals

How to Sample

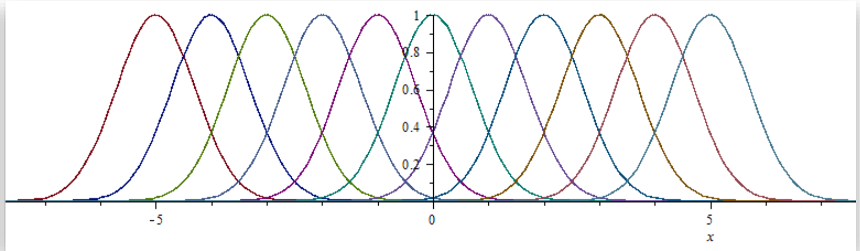
Given

- Function $f: \mathbb{R} \to \mathbb{R}$

Uniform sampling

- Sample spacing δ (given)

Choose filter kernel



- In case of doubt, try:

$$\omega(\mathrm{x}) = \exp(-\delta^{-1}x^2)$$

- Sample $(f \otimes \omega(\mathrm{x}))$ regularly
  - For example: Monte-Carlo integration

How to Sample

Given

- Function $f: \mathbb{R}^n \to \mathbb{R}^m$

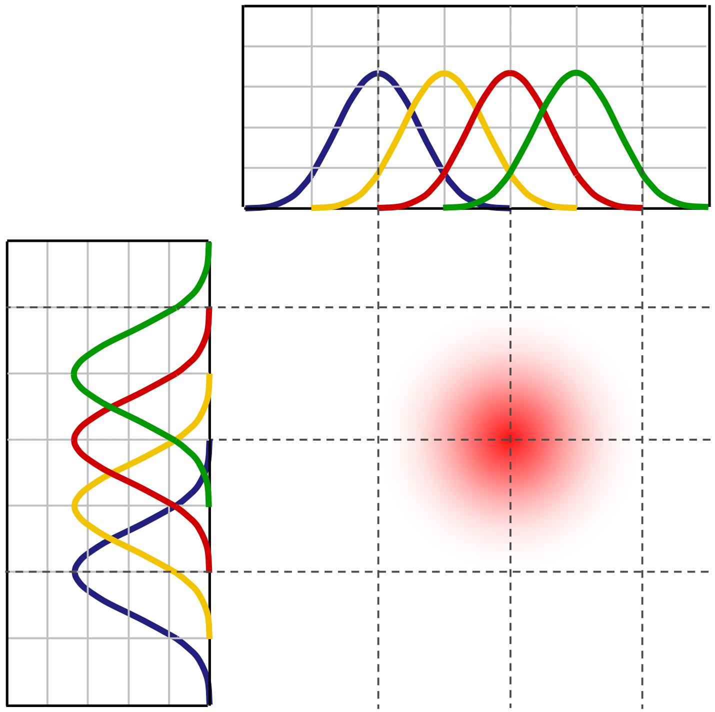
Multi-dimensional Gaussian

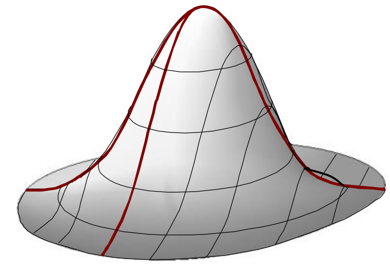
- In case of doubt, try:

$$\omega(\mathrm{x}) = \prod_{d=1}^{\mathrm{n}} \exp\left(-\frac{1}{\delta} x_d^2\right)$$

- Same procedure otherwise…

How to Sample





Multi-dimensional Gaussian

$$\omega(\mathrm{x}) = \prod_{d=1}^{n} \exp\left(-\frac{1}{\delta} x_d^2\right)$$

How to Sample

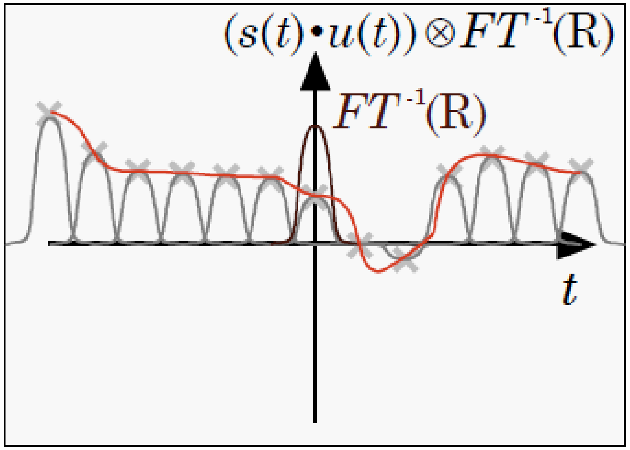
Non-Uniform Sampling

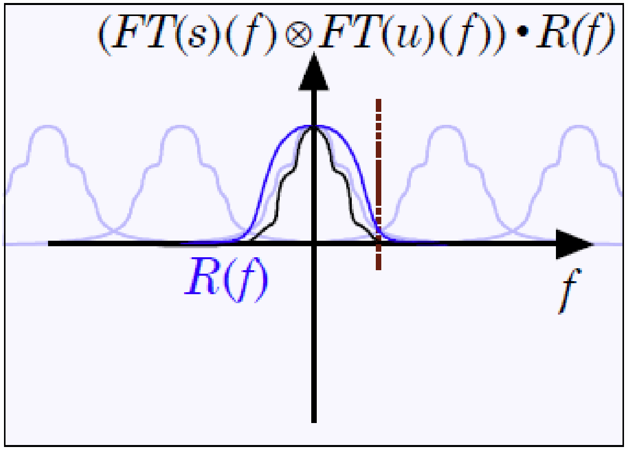
- Choose sample spacing $\delta(x)$
- Match level of detail
  - Nyquest limit
  - Spacing between two “ups” = frequency
- Filter adaptively
  - Varying filter width
- Sample adaptively
  - Sampling width varies accordingly

Recipe:

Reconstructing Signals

Signal Rec





Uniform

- Given samples $y_i = f(x_i), i = 1, \ldots, n$, spacing δ
- Chose reconstruction filter
- Try: $\omega(\mathrm{x}) = \exp(-\delta^{-1} x^2)$

Reconstruction: $\tilde{f} = \sum_{i=1}^{n} y_i \cdot \omega(\mathrm{x} - x_i)$

Non-Uniform

Non-Uniform

- Samples $y_i = f(x_i), i = 1, \dots, n,$
- Varying spacing δ_i
  - If unknown: average spacing of k-nearest neighbors
- Chose reconstruction filter
- Try: $\omega_i(\mathrm{x}) = \exp\left(-\delta_i^{-1}(x - x_i)^2\right)$

Reconstruction:

$$\tilde{f} = \frac{\sum_{i=1}^{n} y_i \cdot \omega_i(\mathrm{x} - x_i)}{\sum_{i=1}^{n} \omega_i(\mathrm{x} - x_i)}$$

"Partition of Unity" just to be save...

Reconstruction: Implementation

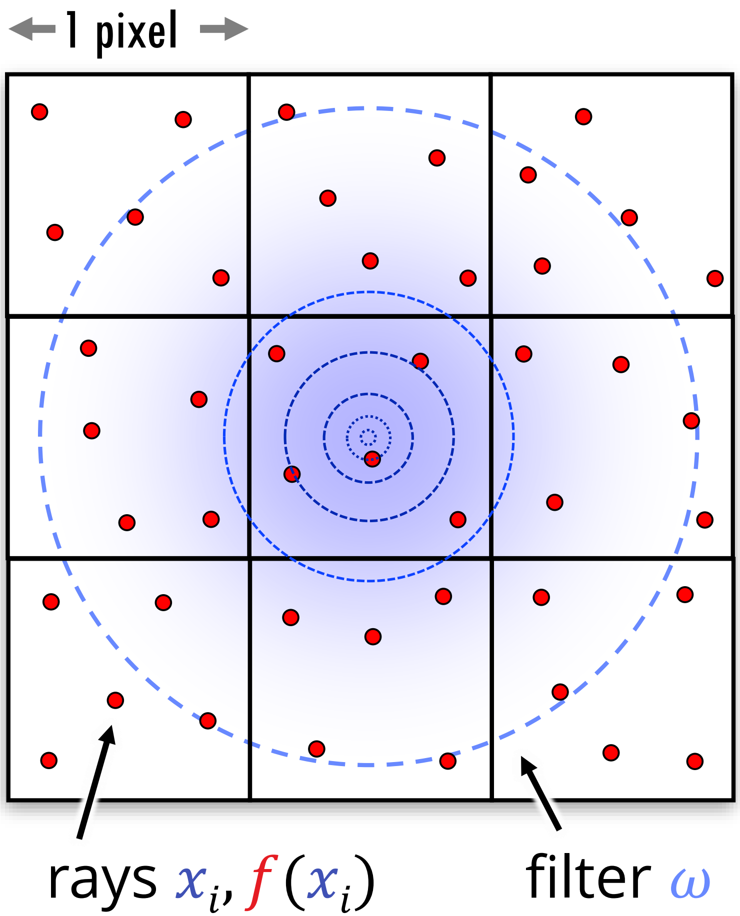
Variant 1: Gathering

- Record *samples in list* (plus kD Tree, Octree, grid)
- For each *pixel*:
  - Range query: kernel support radius
  - Compute weighted sum (last slide)

Variant 2: Splatting

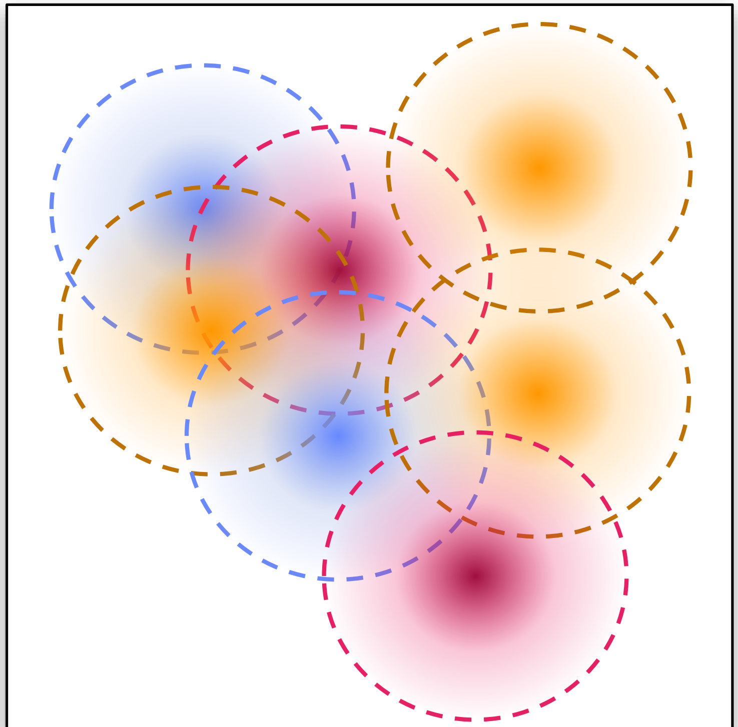
- Two *pixel* buffers: Color (3D), weight (1D)
- Iterate over *samples*:
  - Add Gaussian splat to weight buffer
  - Add 3× Gaussian splat scaled by RGB to color buffer
- In the end: Divide *color buffer* by *weight buffer*.

Gathering

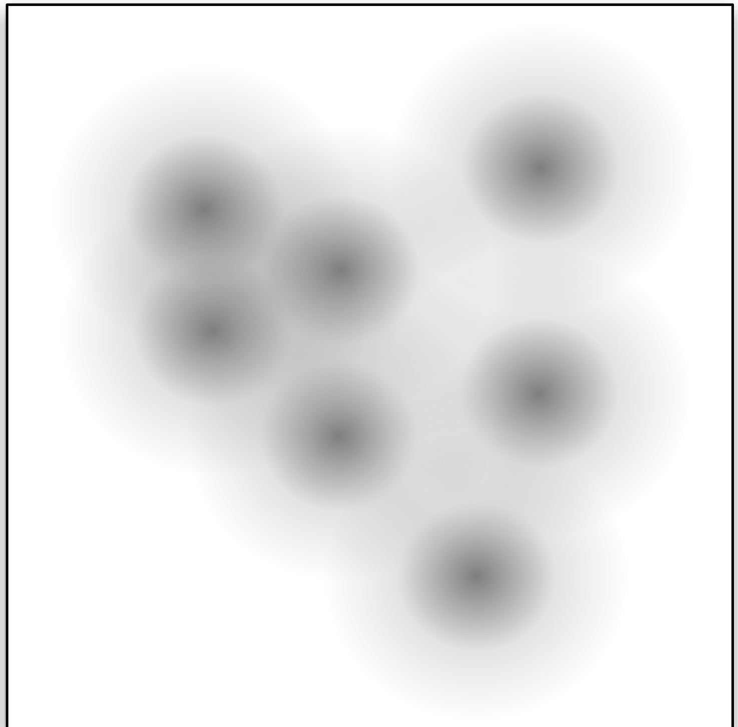


$$\tilde{f} = \frac{\sum_{i=1}^{n} y_i \cdot \omega(\mathrm{x} - x_i)}{\sum_{i=1}^{n} \omega(\mathrm{x} - x_i)}$$

Splatting



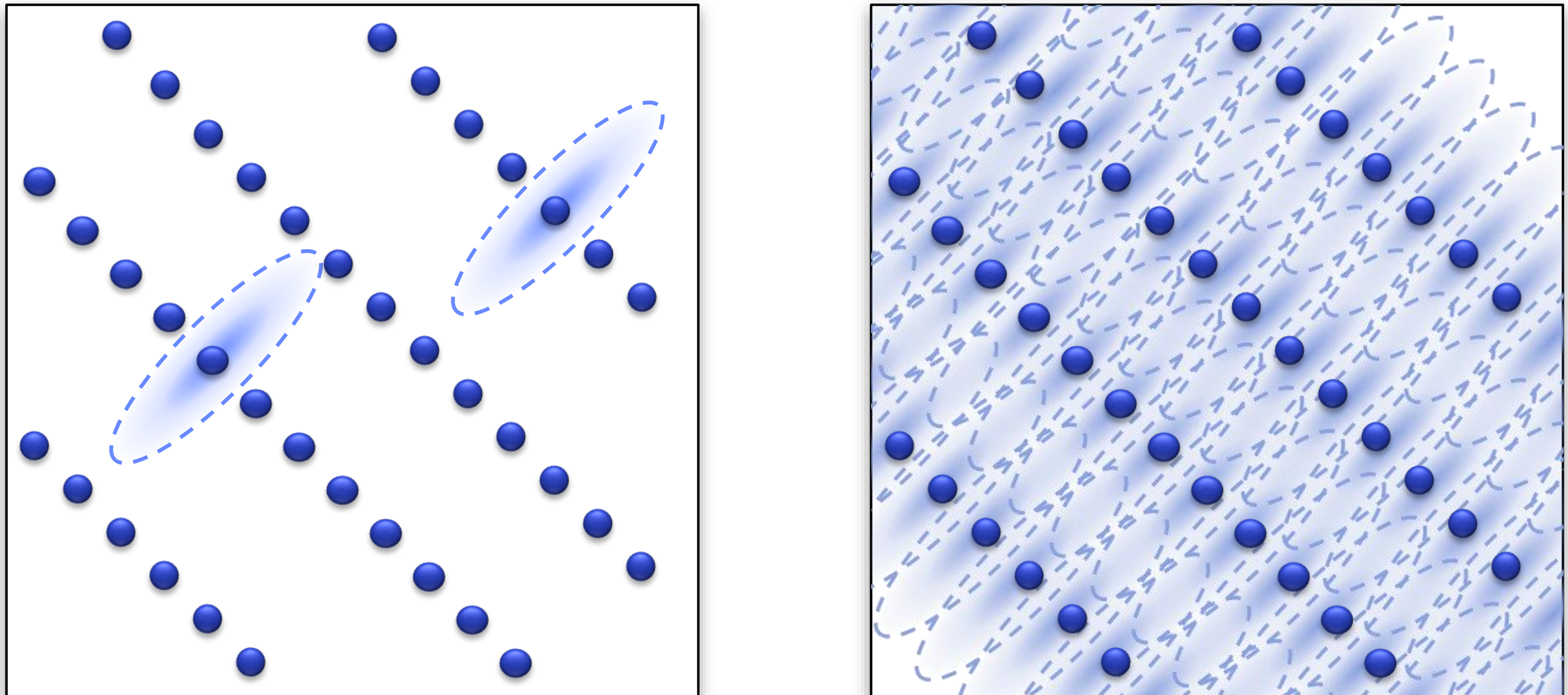
color buffer



weight buffer

$$\tilde{f} = \frac{\sum_{i=1}^{n} y_i \cdot \omega(\mathrm{x} - x_i)}{\sum_{i=1}^{n} \omega(\mathrm{x} - x_i)}$$

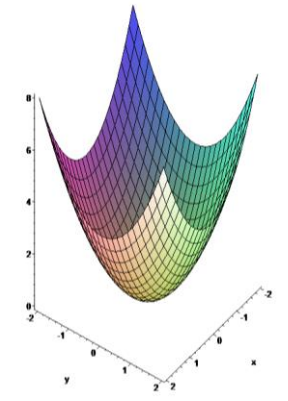
Remark: Anisotropic Filtering

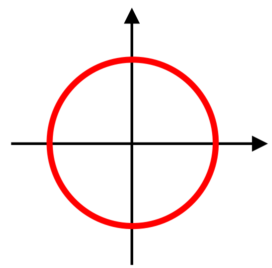


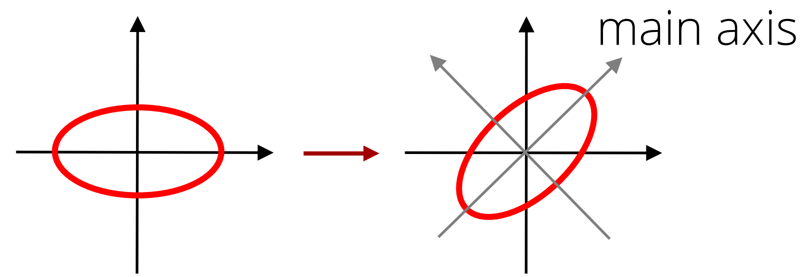
$$\tilde{f} = \frac{\sum_{i=1}^{n} y_i \cdot \omega(\mathrm{x} - x_i)}{\sum_{i=1}^{n} \omega(\mathrm{x} - x_i)}$$

Building Anisotropic Filters

$\mathbf{x}^{\mathrm{T}}\mathbf{x}$







$$\mathbf{x}^{\mathrm{T}}[\mathbf{T}^{\mathrm{T}} \cdot \mathbf{T}]\mathbf{x}$$

How to construct?

- Given: Kernel $w(\mathbf{x})$
  - For example: $w(\mathbf{x}) = \exp\left(-\frac{1}{2\sigma}\mathbf{x}^{\mathrm{T}}\mathbf{x}\right)$
- Coordinate transformation:
  - $w(\mathbf{x}) \to w(\mathbf{T}\mathbf{x})$
  - Gaussian: $w(\mathbf{x}) = \exp\left(-\frac{1}{2\sigma}\mathbf{x}^{\mathrm{T}}[\mathbf{T}^{\mathrm{T}} \cdot \mathbf{T}]\mathbf{x}\right)$

Advanced
Reconstruction

Push-Pull Algorithm

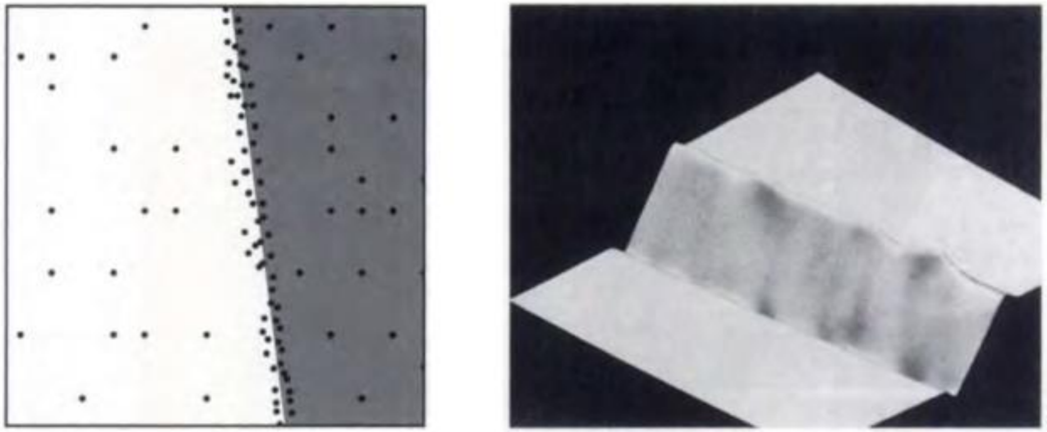
(a)  (b)

FIGURE 10.101
(a) The test situation: a straight edge between black and white regions. (b) A failure of weighted-average reconstruction. Reprinted, by permission, from Mitchell in *Computer Graphics (Proc. Siggraph '87)*, fig. 11, p. 72.

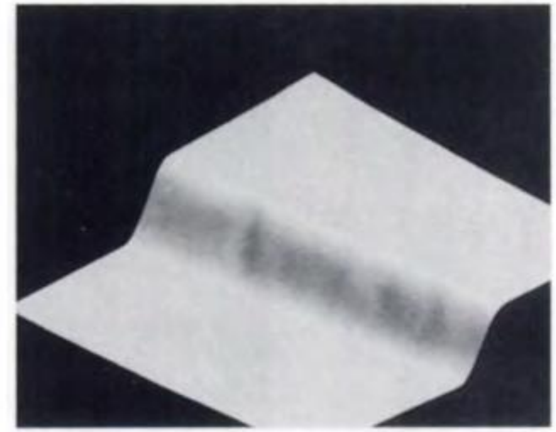


FIGURE 10.103
Reconstruction with the Mitchell multistage filter. Reprinted, by permission, from Mitchell in *Computer Graphics (Proc. Siggraph '87)*, fig. 14, p. 72.

Source: [Glassner 1995, Principles of digital image synthesis, CC license]

Remedy

Push-Pull-Algorithm

- Reconstruct at multiple levels (stratification)
  - Build quadtree
  - Keep one sample per cell
  - Creates different levels
- Add results together
  - Do not reconstruct in empty cells

Reduced bias

Advanced Reconstruction

Moving Least-Squares

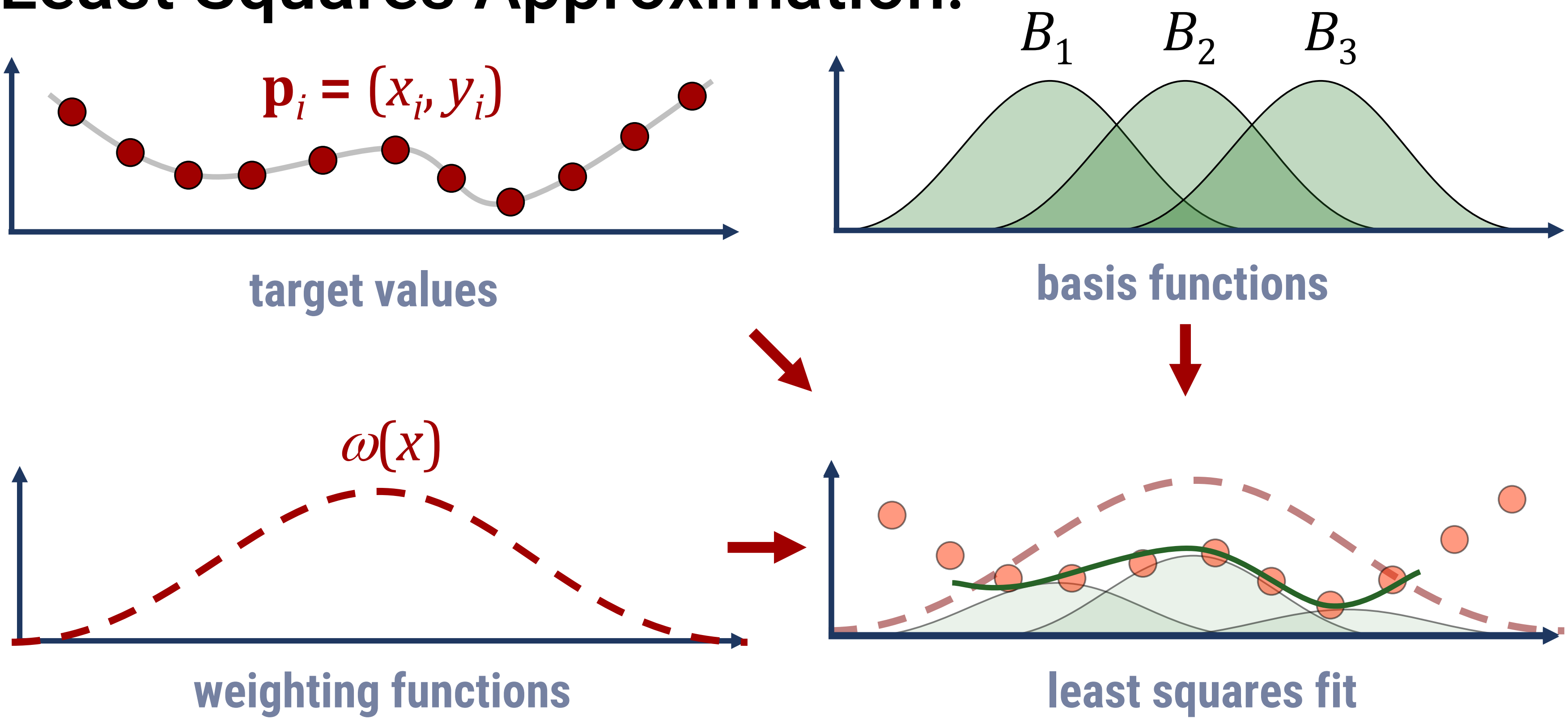
Moving Least Squares

Moving least squares (MLS):

- MLS is a standard technique for scattered data interpolation.
- Generalization of partition-of-unity method

Weighted Least-Squares

Least Squares Approximation:



Least-Squares

Least Squares Approximation:

$$\tilde{y}(x) = \sum_{i=1}^{n} \lambda_i B_i(x)$$

Best Fit (weighted):

$$\underset{c_i}{\operatorname{argmin}} \sum_{i=1}^{n} \left\| \left(\tilde{y}(x_i) - y_i \right) \omega(x_i) \right\|^2$$

Least-Squares

Notation:

Normal Equations: $\left(\mathbf{B}^T\mathbf{W}^2\mathbf{B}\right)\boldsymbol{\lambda} = \left(\mathbf{B}^T\mathbf{W}^2\right)\mathbf{y}$

Solution: $\boldsymbol{\lambda} = \left(\mathbf{B}^T\mathbf{W}^2\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{W}^2\mathbf{y}$

Evaluation: $\tilde{y}(x) = \langle \mathbf{b}(x), \boldsymbol{\lambda} \rangle = \mathbf{b}(x)^{\mathrm{T}}\left(\mathbf{B}^{\mathrm{T}}\mathbf{W}^2\mathbf{B}\right)^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{W}^2\mathbf{y}$

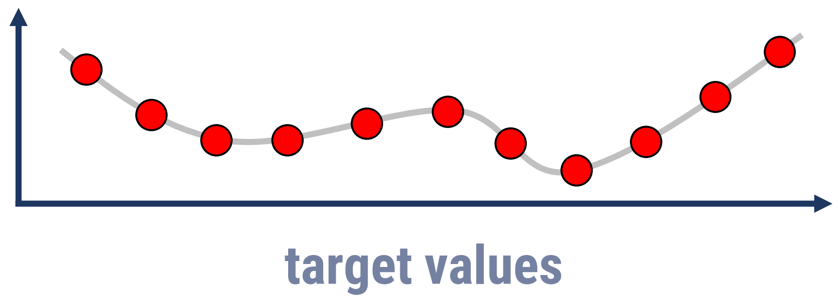
MLS approximation

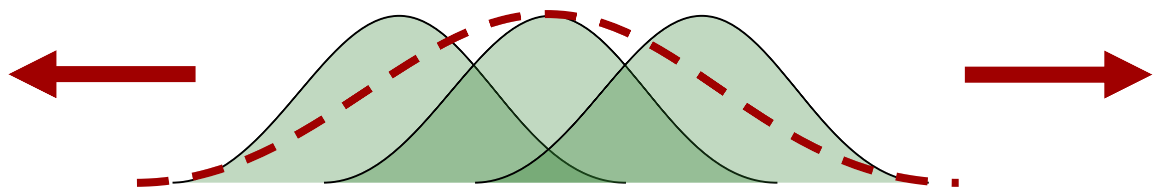
$$\mathbf{b} := \left[B_1, \ldots, B_n\right]$$

$$\mathbf{B} := \begin{bmatrix} -\mathbf{b}(x_1)- \\ \vdots \\ -\mathbf{b}(x_n)- \end{bmatrix} \qquad \mathbf{y} := \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \mathbf{W} := \begin{bmatrix} \omega(x_1) & & \\ & \ddots & \\ & & \omega(x_n) \end{bmatrix}$$

Moving Least-Squares

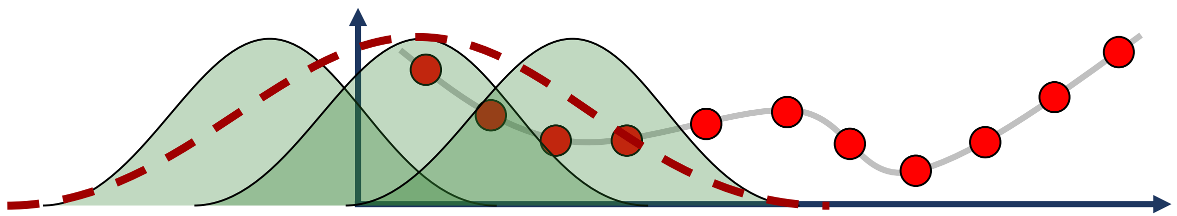
Moving Least Squares Approximation:



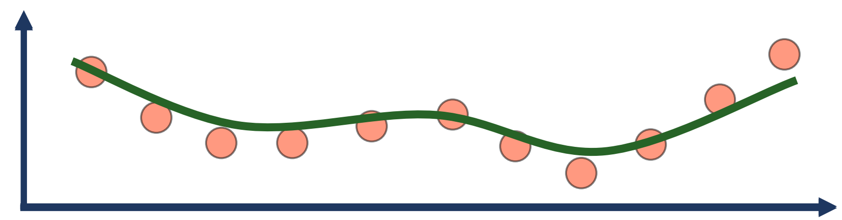


move basis and weighting function,
recompute approximation $\tilde{y}(x)$

Moving Least Squares Approximation:



target values



approximation

Summary: MLS

Standard MLS approximation:

- Choose set of basis functions
  - Typically monomials of degree 0,1,2
- Choose weighting function
  - Typical choices: Gaussian, Wendland function, B-Splines
  - Solution will have the same continuity as the weighting function.
- Solve a weighted least squares problem at each point:

$$\tilde{y}(x) = \mathbf{b}(x)^{\mathrm{T}} \Big(\underbrace{\mathbf{B}(x)^{\mathrm{T}} \mathbf{W}(x)^2 \mathbf{B}(x)}_{\textbf{moment matrix}} \Big)^{-1} \mathbf{B}(x)^{\mathrm{T}} \mathbf{W}(x)^2 \mathbf{y}$$

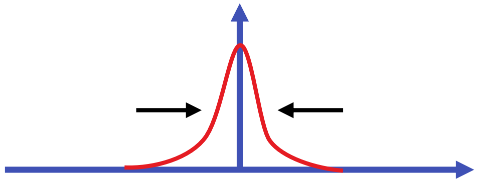
- Need to invert the “moment matrix” at each evaluation.
- Use SVD if sampling requirements are not guaranteed.

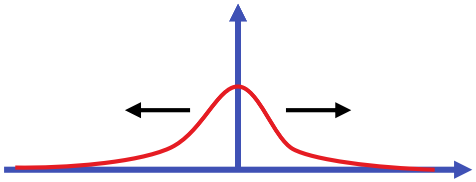
Remark

Uncertainty Relation(s)

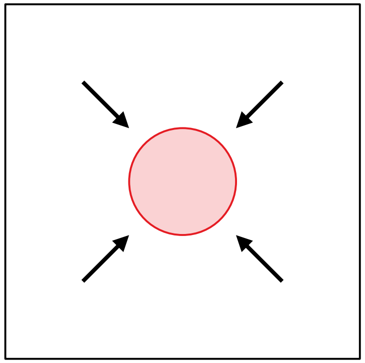
Fourier Transform Pairs

Gaussians

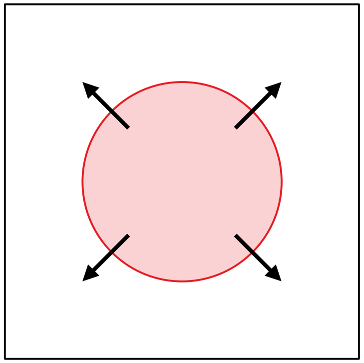




$$f(x) = e^{-ax^2} \quad \rightarrow \quad F(\omega) = \sqrt{\frac{\pi}{a}} \cdot e^{-\frac{(\pi\omega)^2}{a}}$$



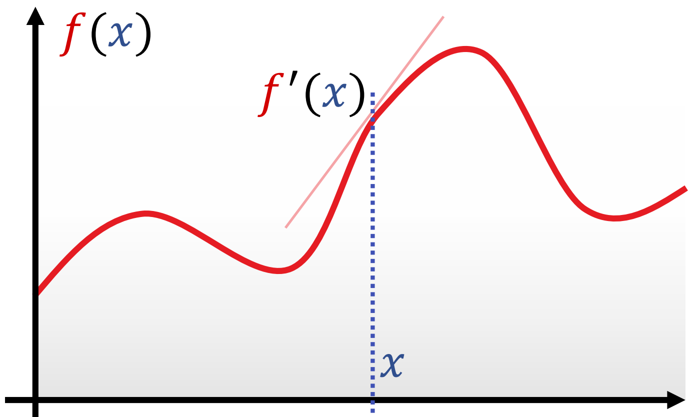
x



ω

Taylor-Approximation

Function f

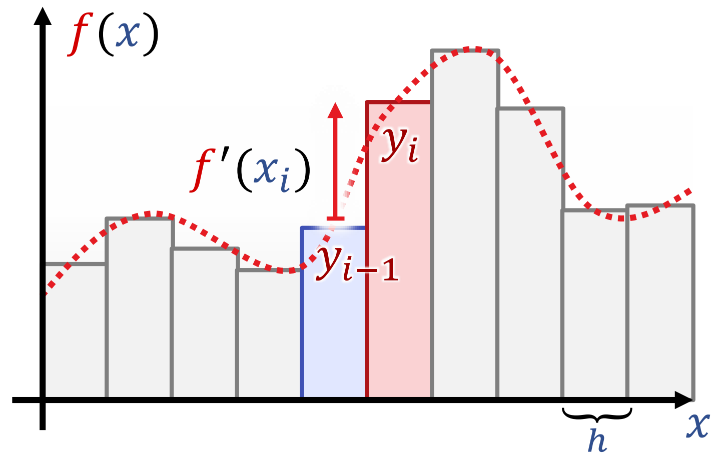


tangent slope

$$f: \mathbb{R} \to \mathbb{R}$$

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Think of this:



neighborhood differences

$$f = (y_1, \dots, y_n)$$

$$f'(x_i) \approx \frac{y_i - y_{i-1}}{h}$$