Review of Sampling, Deconvolution & Linear Systems

EE367/CS448I: Computational Imaging

stanford.edu/class/ee367

Lecture 5

Gordon Wetzstein

Stanford University

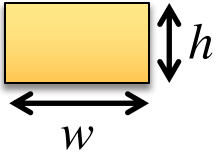
What's a Discrete Image?

- continuous 2D visual signal on sensor: $i(x,y)$

- integration over pixels: $\tilde{i}(x,y) = i(x,y) * \left(rect\left[\frac{x}{w}\right] \cdot rect\left[\frac{y}{h}\right] \right)$

(detector footprint modulation transfer function, Boreman 2001)

sensor pixel:



What's a Discrete Image?

- continuous 2D visual signal on sensor: $i(x,y)$

- integration over pixels: $\tilde{i}(x,y) = i(x,y) * \left(rect\left[\frac{x}{w}\right] \cdot rect\left[\frac{y}{h}\right] \right)$

  (detector footprint modulation transfer function, Boreman 2001)

- discrete sampling: $E[i,j] = sample\left(\tilde{f}(x,y)\right) = \tilde{f}(x,y) \cdot \sum_m \sum_n \delta(i,j)$
  (in irradiance $\frac{W}{m^2}$)

Fourier Transform

- any continuous, integrable, periodic function can be represented as an infinite sum of sines and cosines:

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi) e^{2\pi i \xi x} \, d\xi \quad \longleftrightarrow \quad \hat{f}(\xi) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i \xi x} \, dx$$

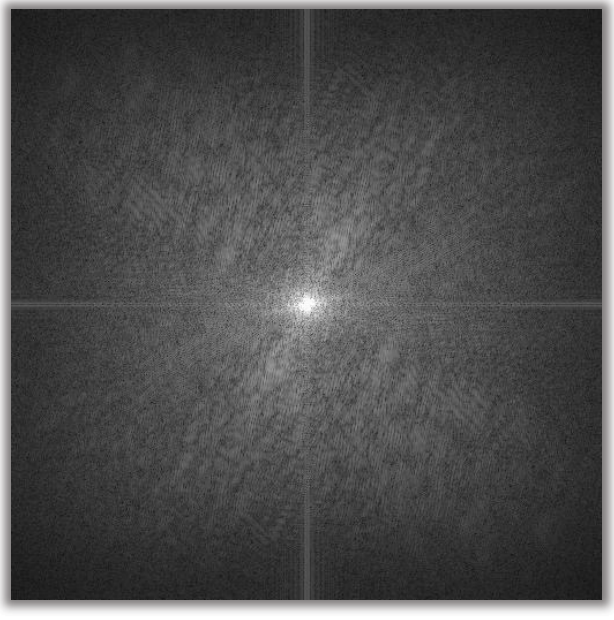
- most important for us: discrete Fourier transform

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} \hat{x}[k] e^{2\pi i k n / N} \quad \longleftrightarrow \quad \hat{x}[k] = \sum_{n=0}^{N-1} x[n] e^{-2\pi i k n / N}$$

- convolution theorem (critical): $\boxed{x * g = F^{-1}\left\{ F\{x\} \cdot F\{g\} \right\}}$

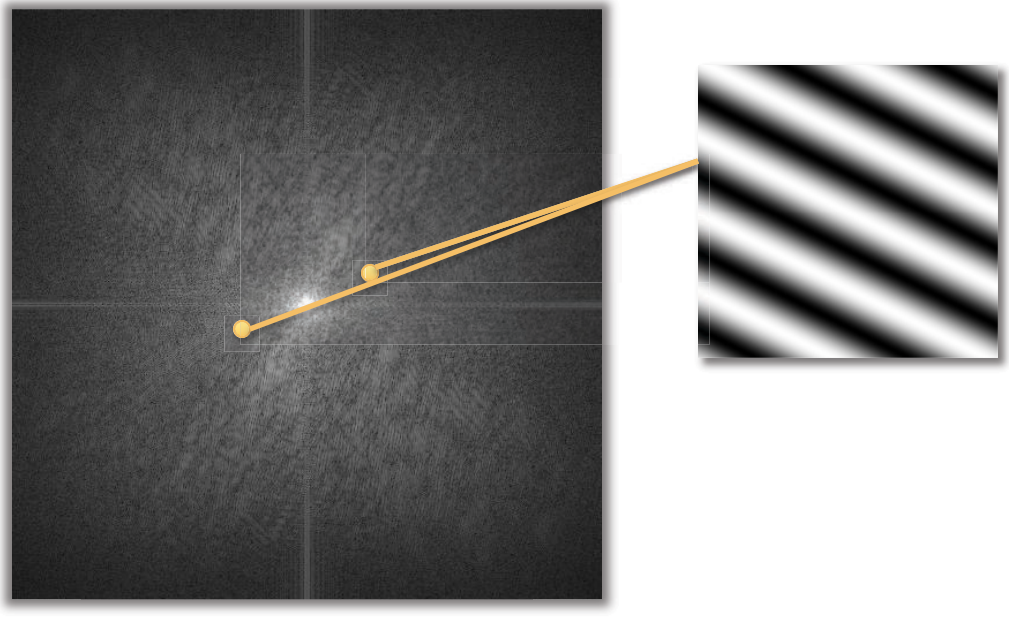
Discrete Fourier Transform

- What is this?



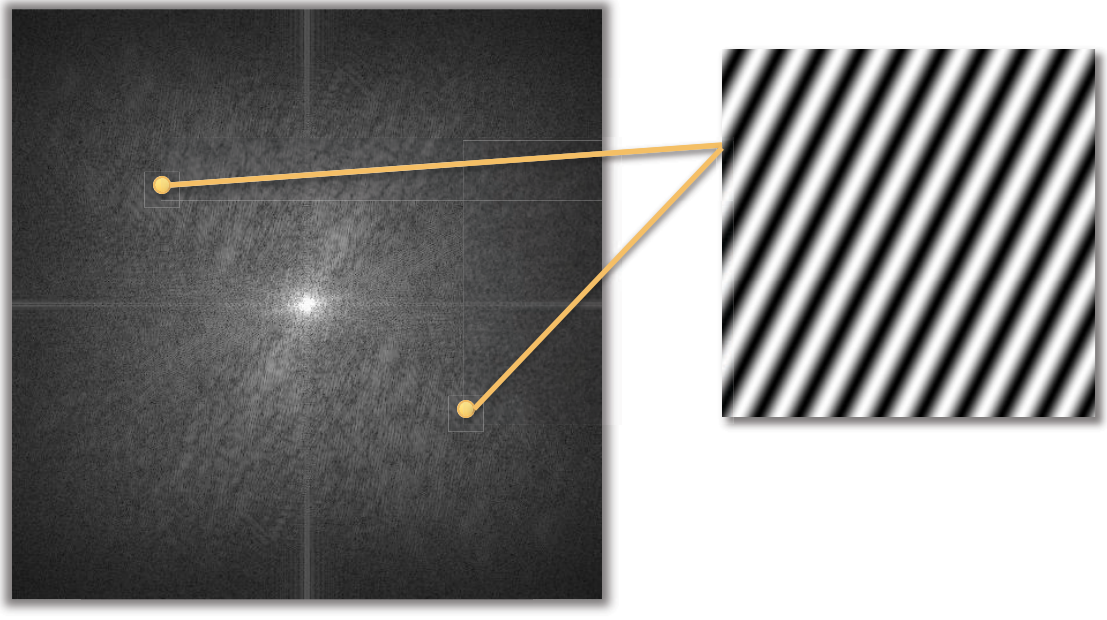
Discrete Fourier Transform

- What is this?



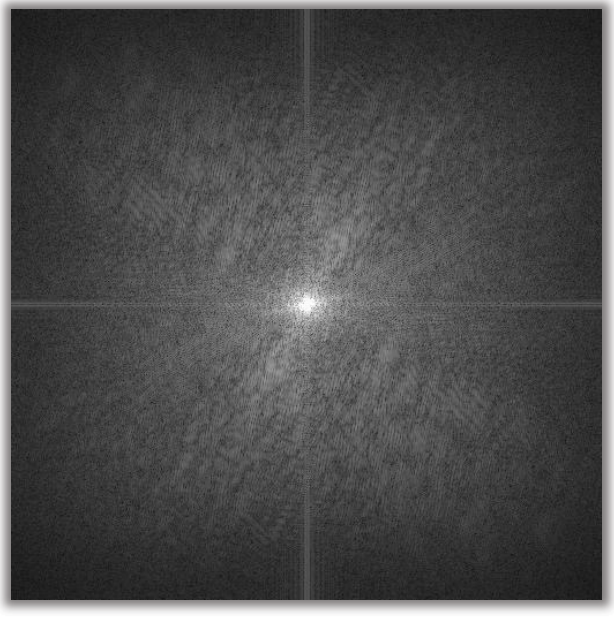
Discrete Fourier Transform

- What is this?



Discrete Fourier Transform

- What is this?



Filtering – Low-pass Filter

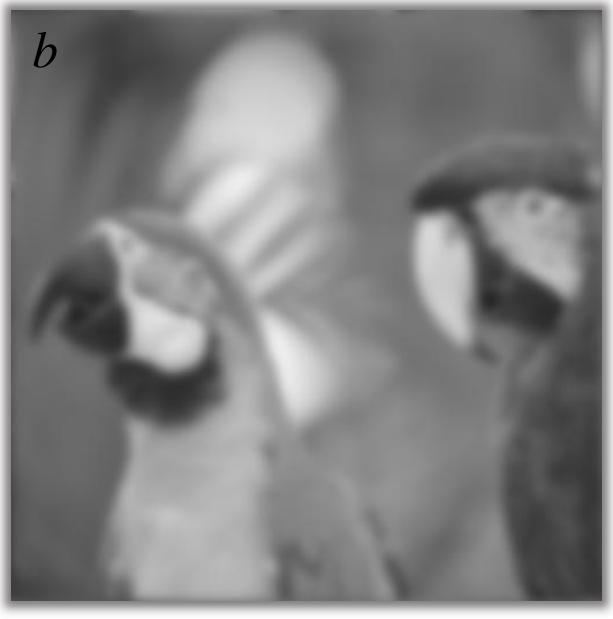
- low-pass filter: convolution in primal domain $b = x * c$
- convolution kernel c is also known as point spread function (PSF)

c

$*$

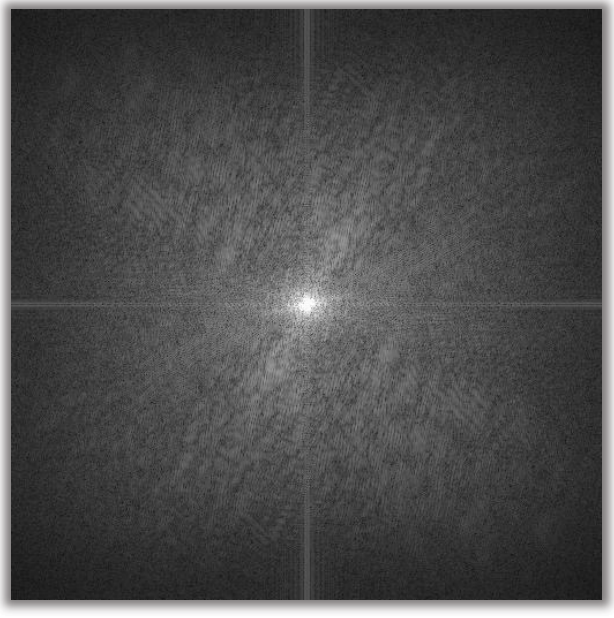
small kernel

$=$



Filtering – Low-pass Filter

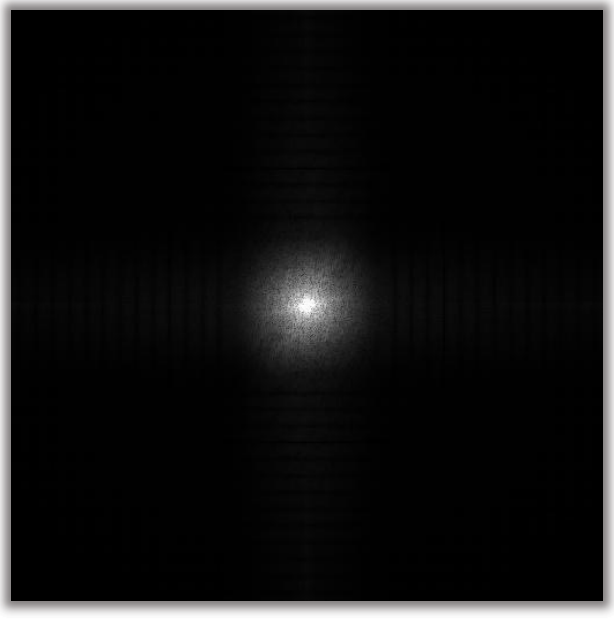
- low-pass filter: multiplication in frequency domain $F\{b\} = F\{x\} \cdot F\{c\}$



·

big

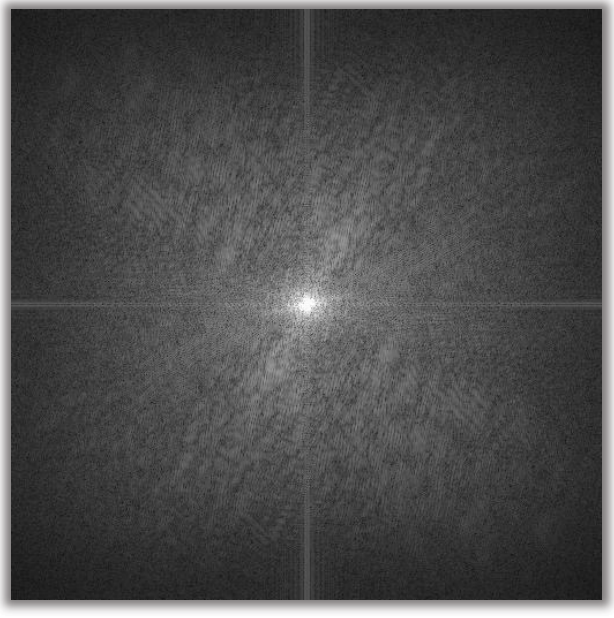
=



Filtering – Low-pass Filter

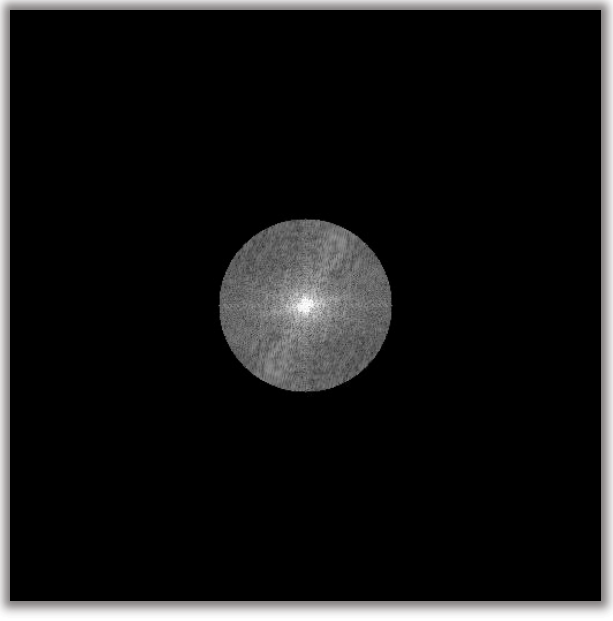
$$F\{b\} = F\{x\} \cdot F\{c\}$$

- low-pass filter: hard cutoff



$\cdot$

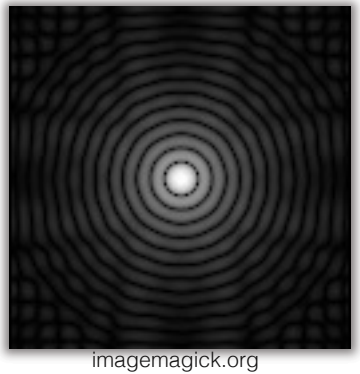
$=$



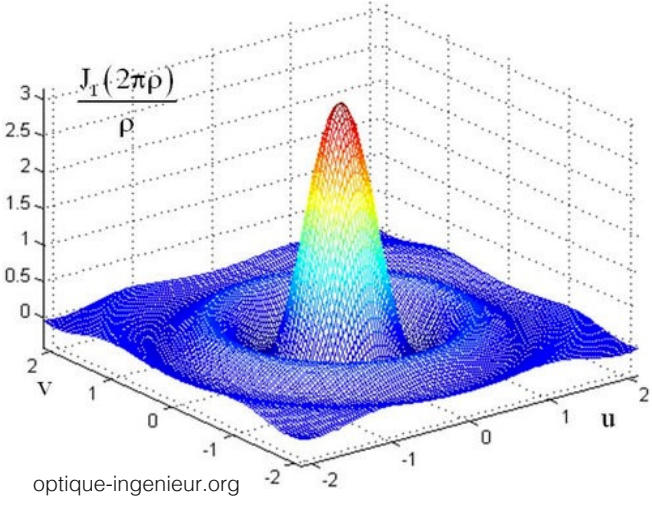
Filtering – Low-pass Filter

- Bessel function of the first kind or “jinc”

$F^{-1}\{$  $\}$



imagemagick.org

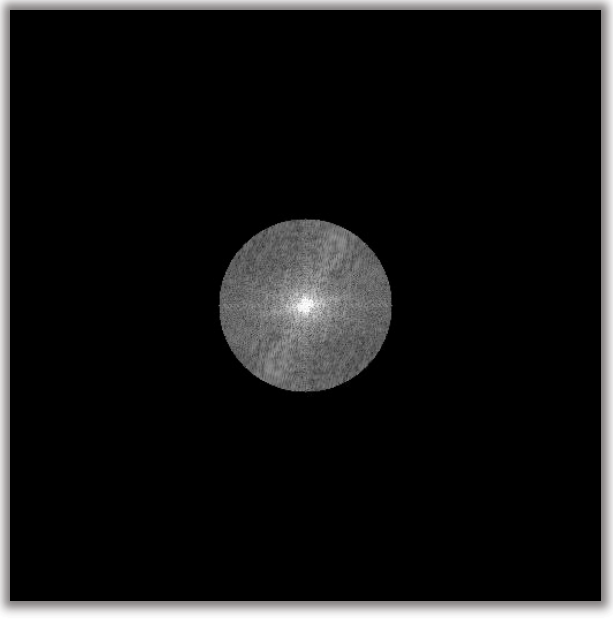


optique-ingenieur.org

Filtering – Low-pass Filter

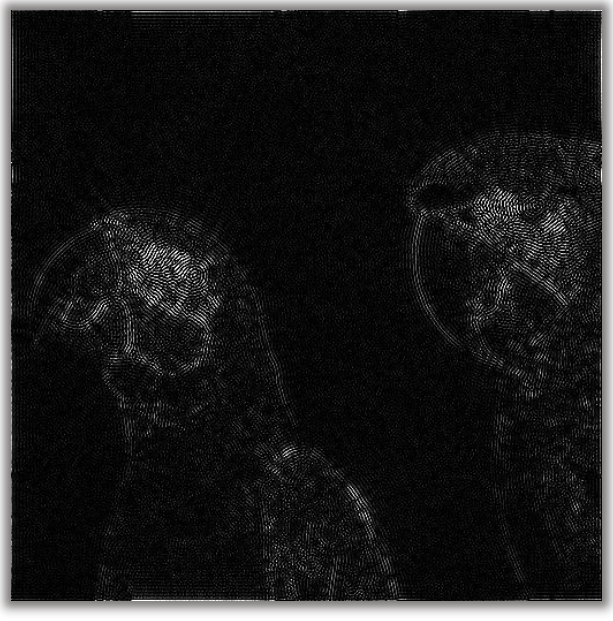
- hard frequency filters often introduce ringing

←

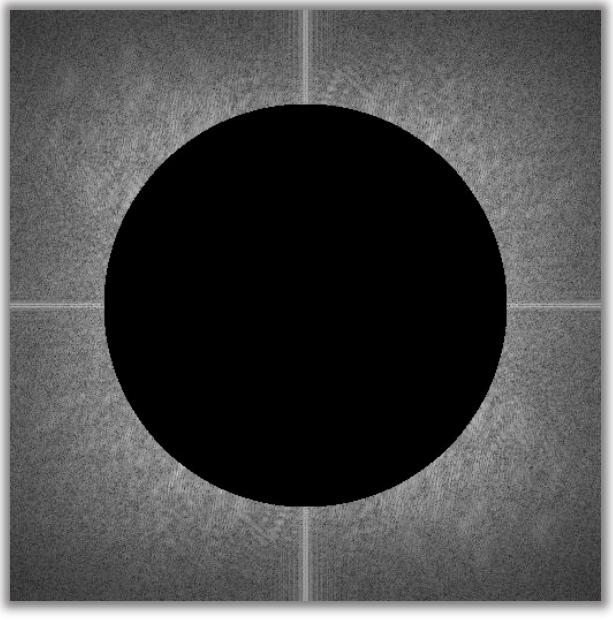


Filtering – High-pass Filter

- sharpening (possibly with ringing, but don't see any here)



←



Filtering – Unsharp Masking

- sharpening (without ringing): unsharp masking, e.g. in Photoshop

$$b = x * (\delta - c_{lowpass\_gauss}) = x - x * c_{lowpass\_gauss}$$

or

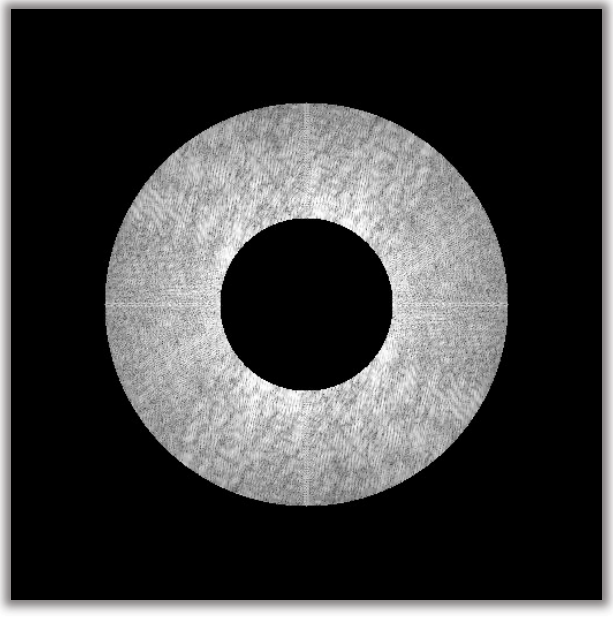
$$b = x * (\delta + c_{highpass}) = x + x * c_{highpass}$$

Filtering – Unsharp Masking

- sharpening (without ringing): unsharp masking, e.g. in Photoshop

Filtering – Band-pass Filter

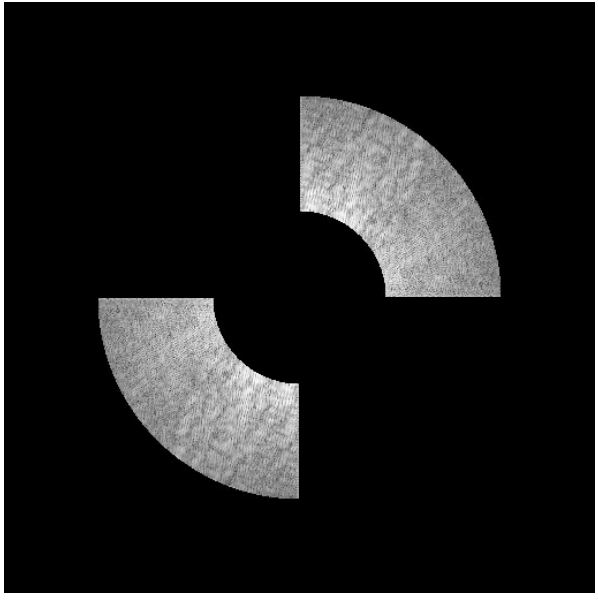
←



Filtering – Oriented Band-pass Filter

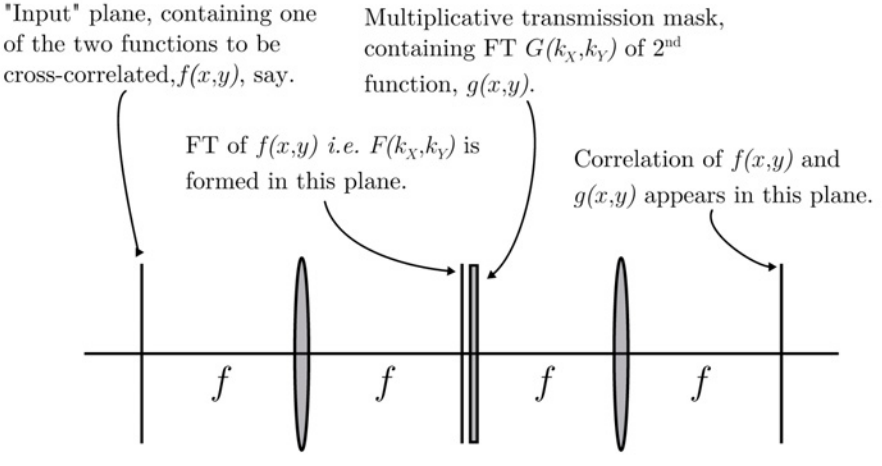
- edges with specific orientation (e.g., hat) are gone!

←



Optical Filtering with Fourier Optics

- can do all of this optically (with coherent light)!
- Fourier optics – more in week 7

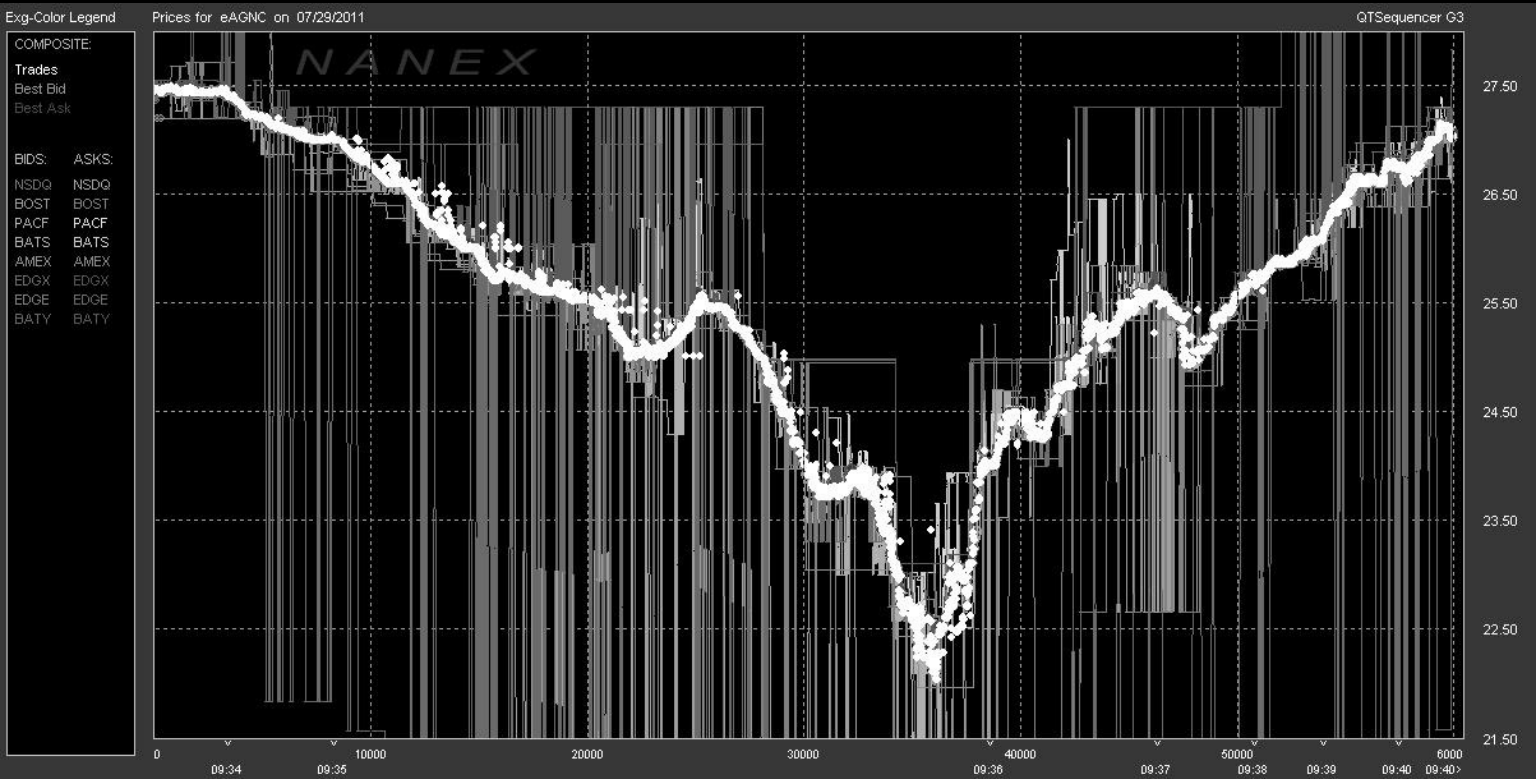


http://en.wikipedia.org/wiki/Fourier_optics

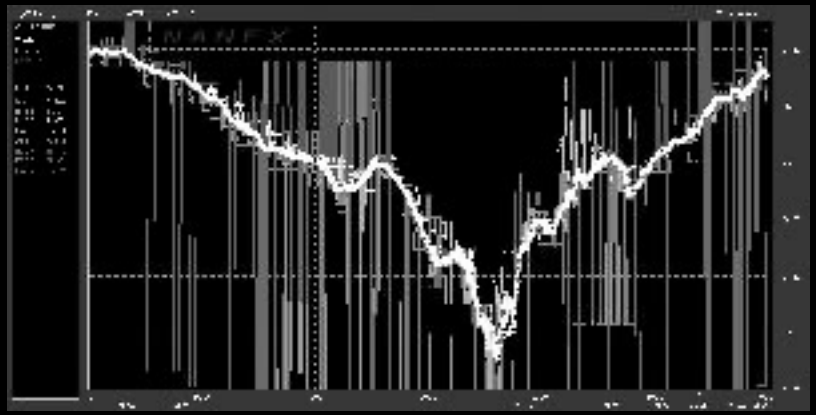
Image Downsampling (& Upsampling)

- best demonstrated with “high-frequency” image
- that’s just resampling, right?

original image: I

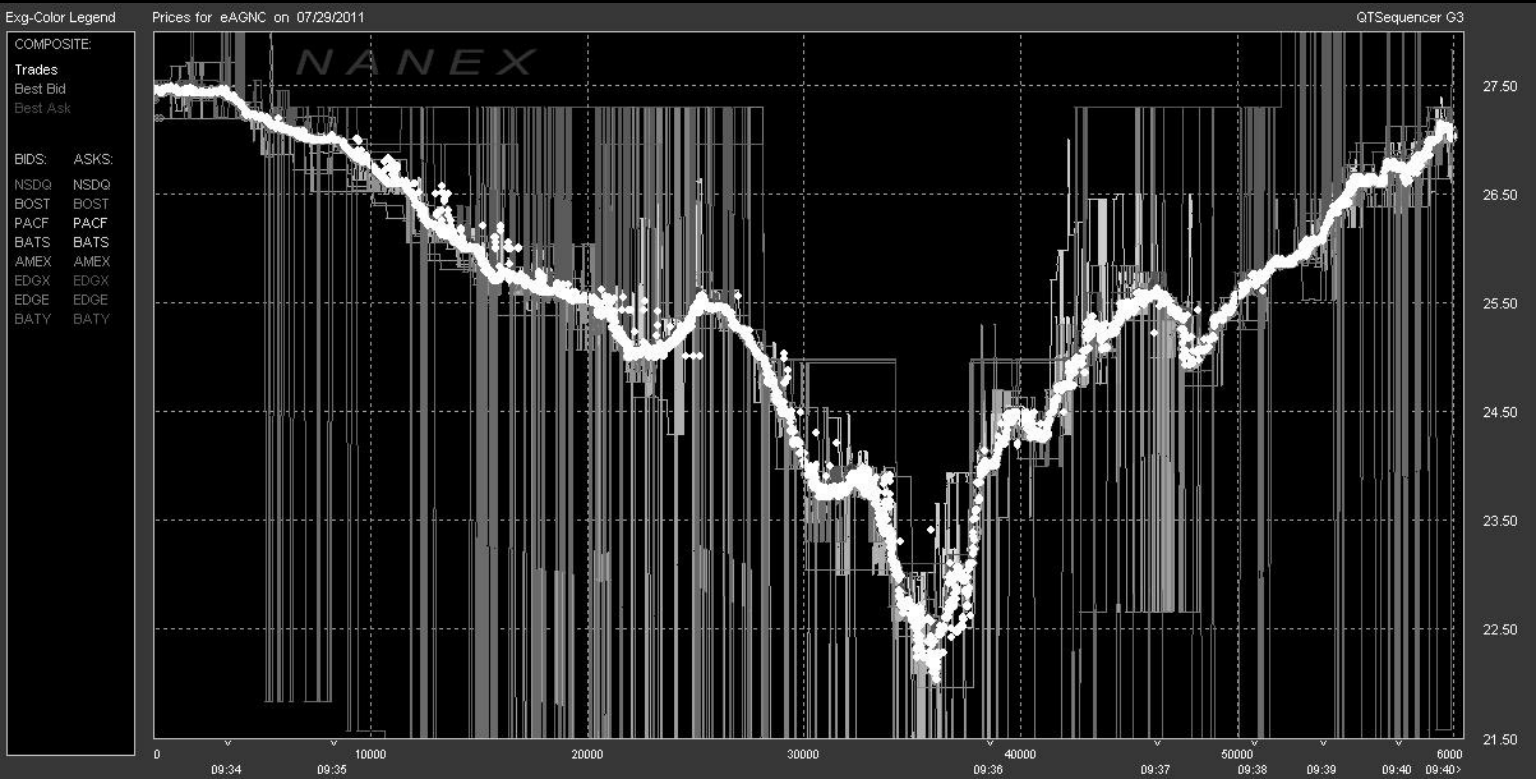


re-sample image: I(1:4:end,1:4:end) in Matlab
something is wrong - aliasing!



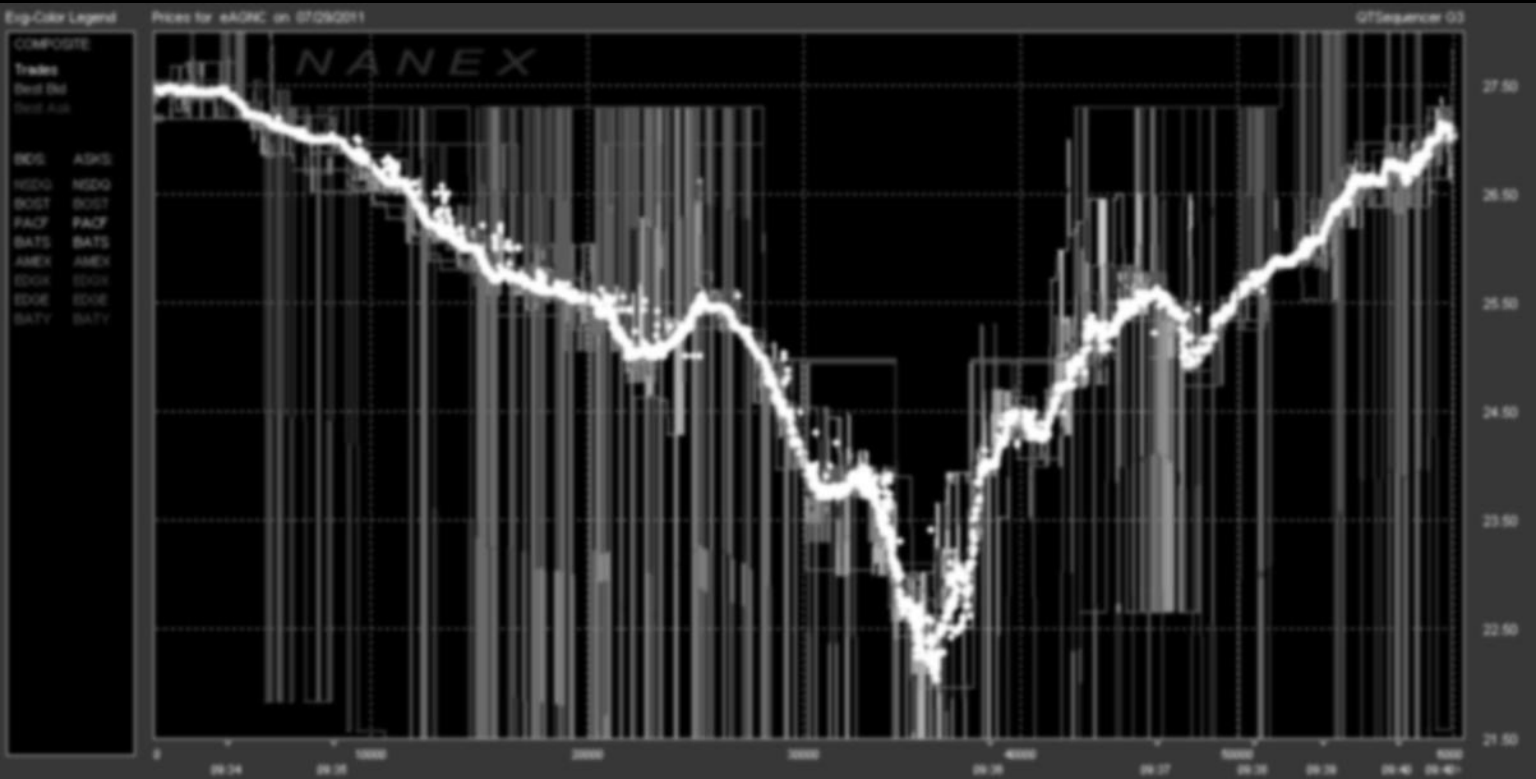
pocketfullofliberty.com/high-frequency-trading

need to low-pass filter image first!



pocketfullofliberty.com/high-frequency-trading

need to low-pass filter image first!



pocketfullofliberty.com/high-frequency-trading

first: filter out high frequencies (“anti-aliasing”)
then: then re-sample image: I(1:4:end,1:4:end)

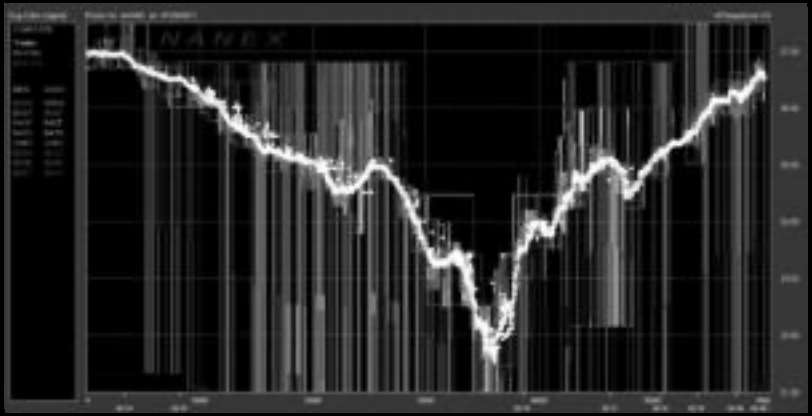


Image Downsampling (& Upsampling)

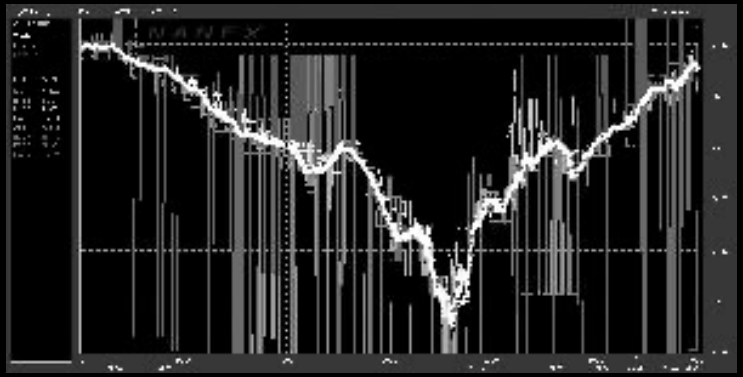
- "anti-aliasing" → before re-sampling, apply appropriate filter!

- how much filtering? Shannon-Nyquist sampling theorem:

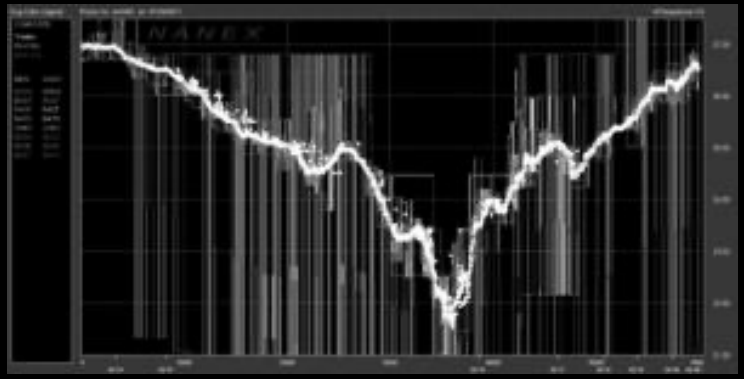
$$\boxed{f_s \geq 2 f_{\max}}$$

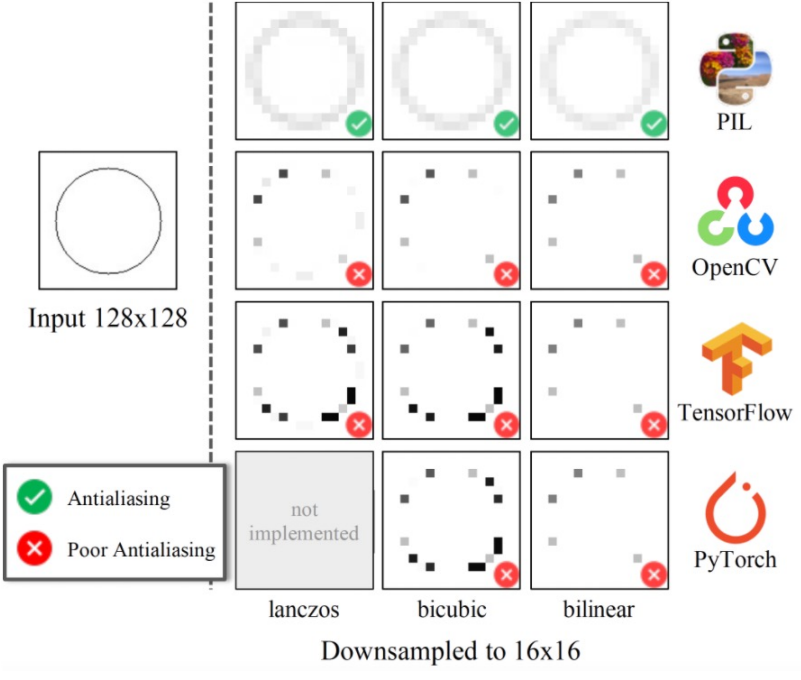
pocketfullofliberty.com/high-frequency-trading

no anti-aliasing



with anti-aliasing

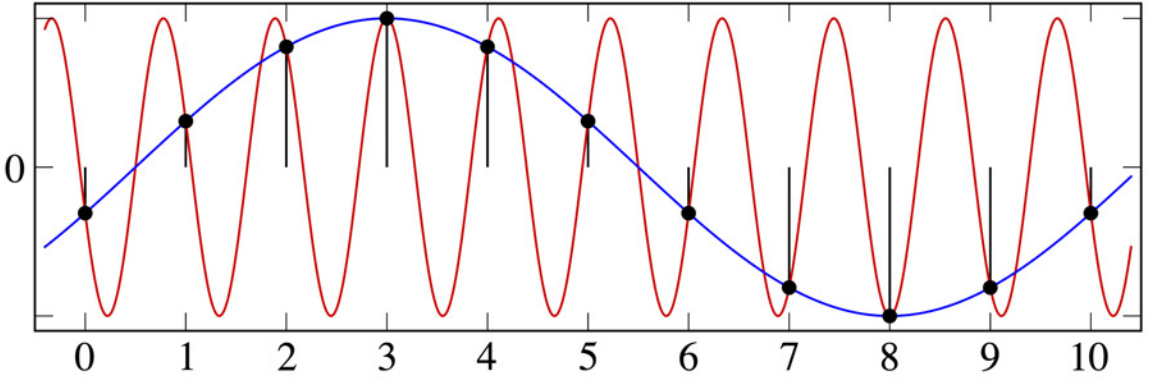




Parmar et al. 2021

Examples of Aliasing: Temporal Aliasing

- wagon wheel effect (temporal aliasing)
- sampling frequency was lower than $2f_{\max}$



wikipedia

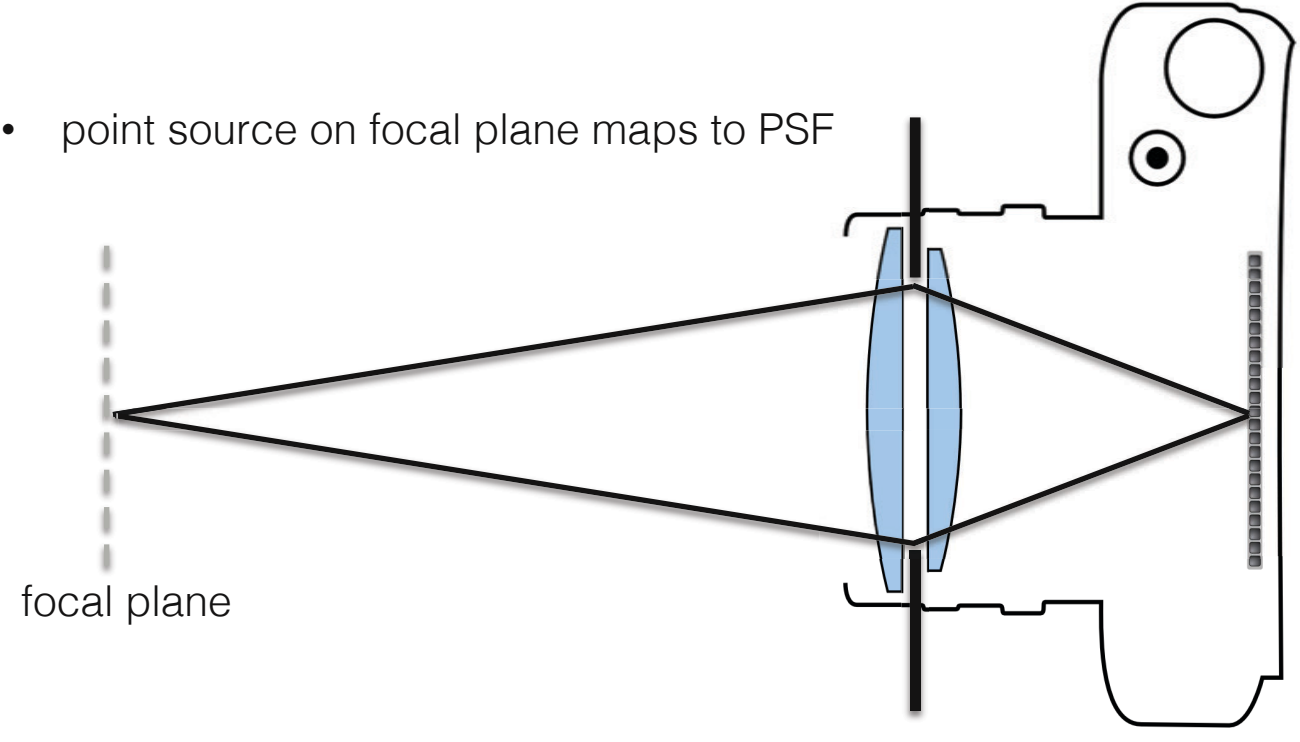
Examples of Aliasing: Temporal Aliasing

- wagon wheel effect

youtube.com/watch?v=jHS9JGkEOmA

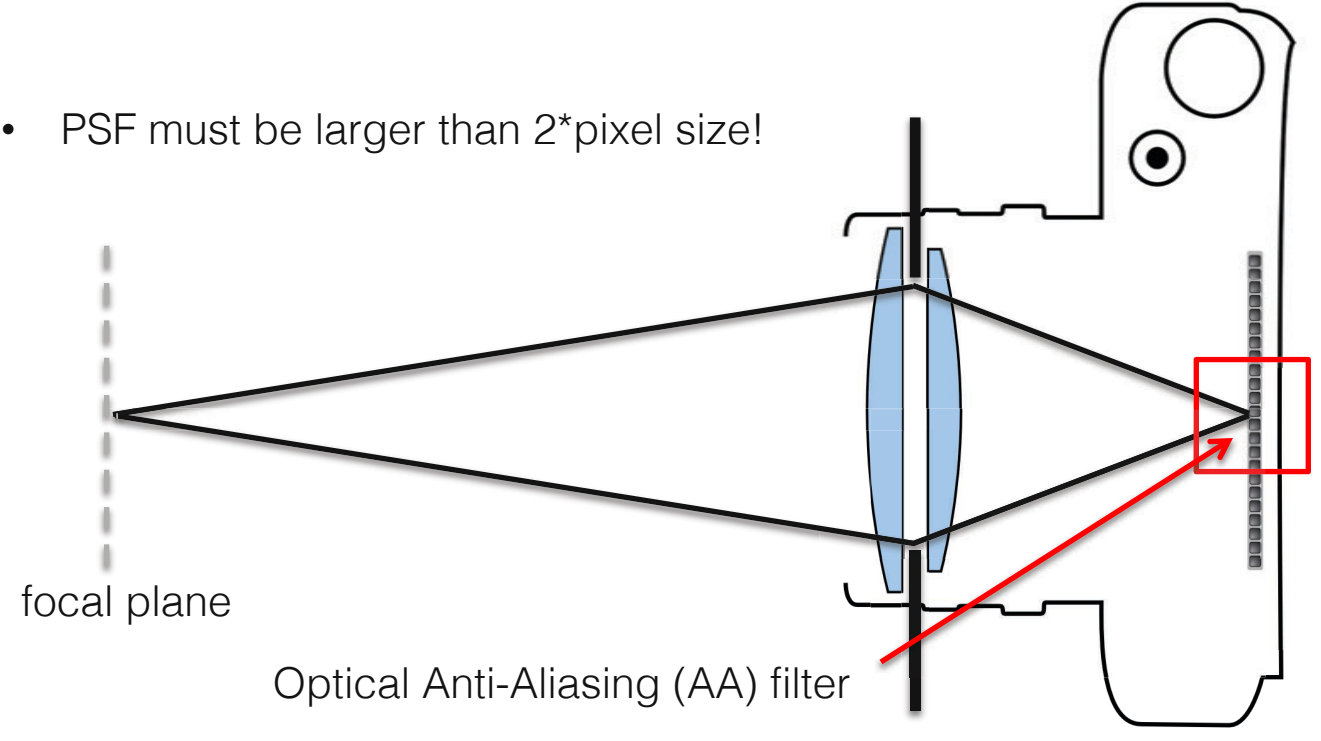
Examples of Aliasing: Sampling on Sensor

- point source on focal plane maps to PSF



Examples of Aliasing: Sampling on Sensor

- PSF must be larger than 2*pixel size!



Other Forms of Aliasing

- photography – optical AA filter removed (“hot rodding” camera)

John Shafer

mosaicengineering.com

Other Forms of Aliasing

- photography – optical AA filter removed (“hot rodding” camera)

without AA filter

with AA filter (standard)

maxmax.com

Other Forms of Aliasing

- photography – optical AA filter removed (“hot rodding” camera)

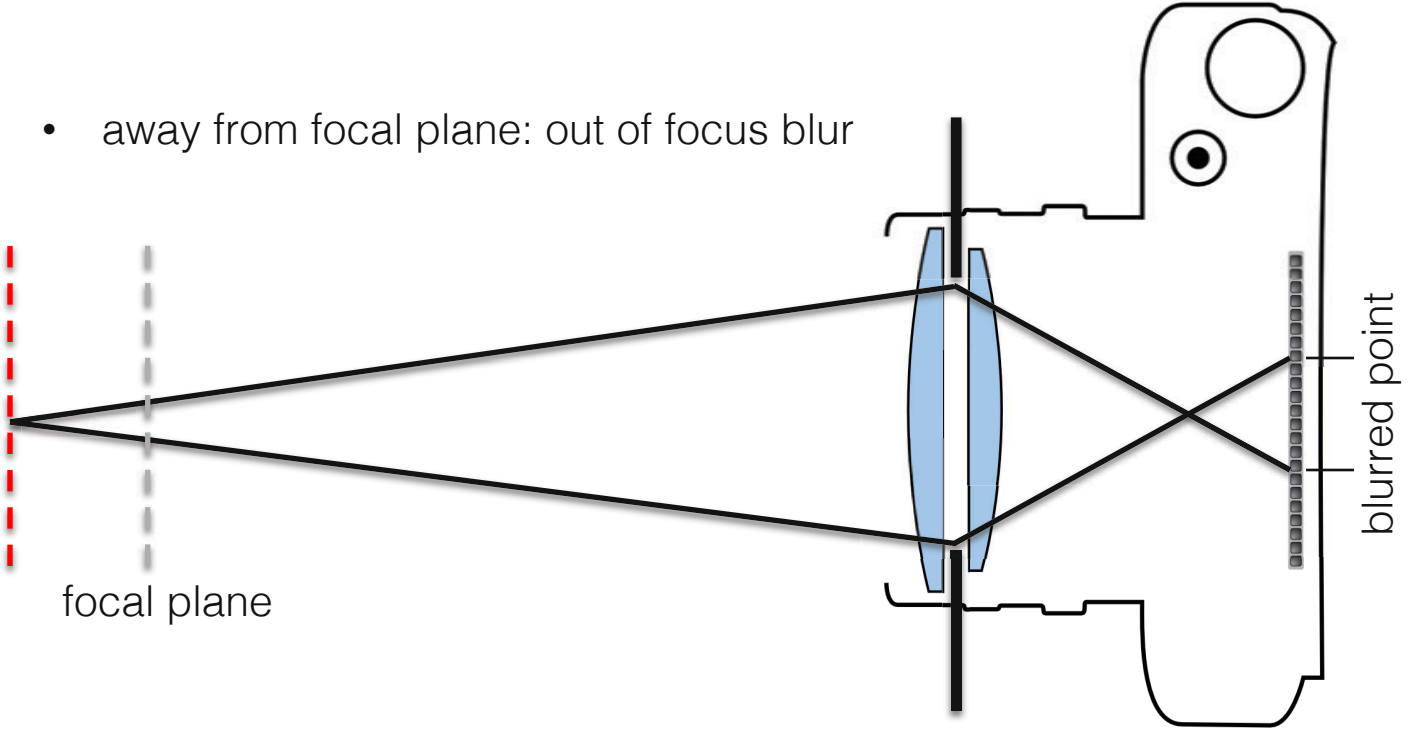
without AA filter

with AA filter (standard)

maxmax.com

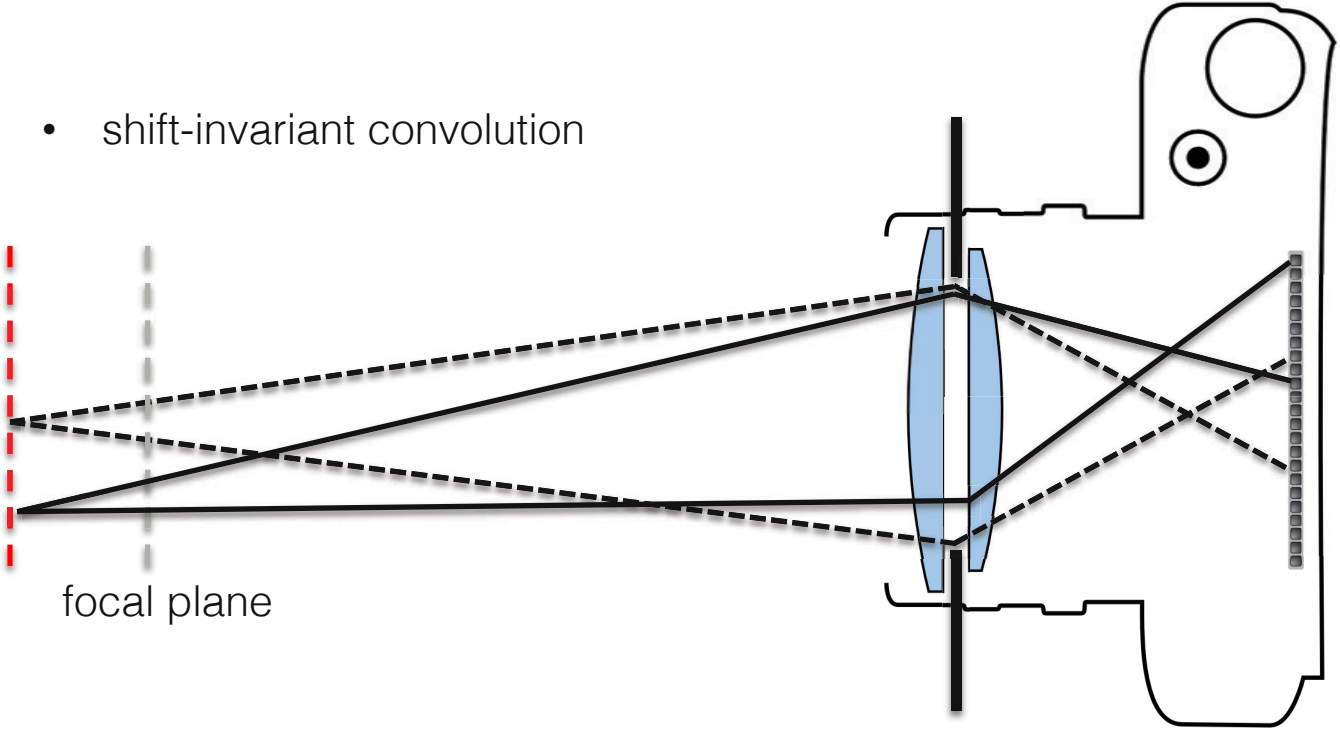
Lens as Optical Low-pass Filter

- away from focal plane: out of focus blur



Lens as Optical Low-pass Filter

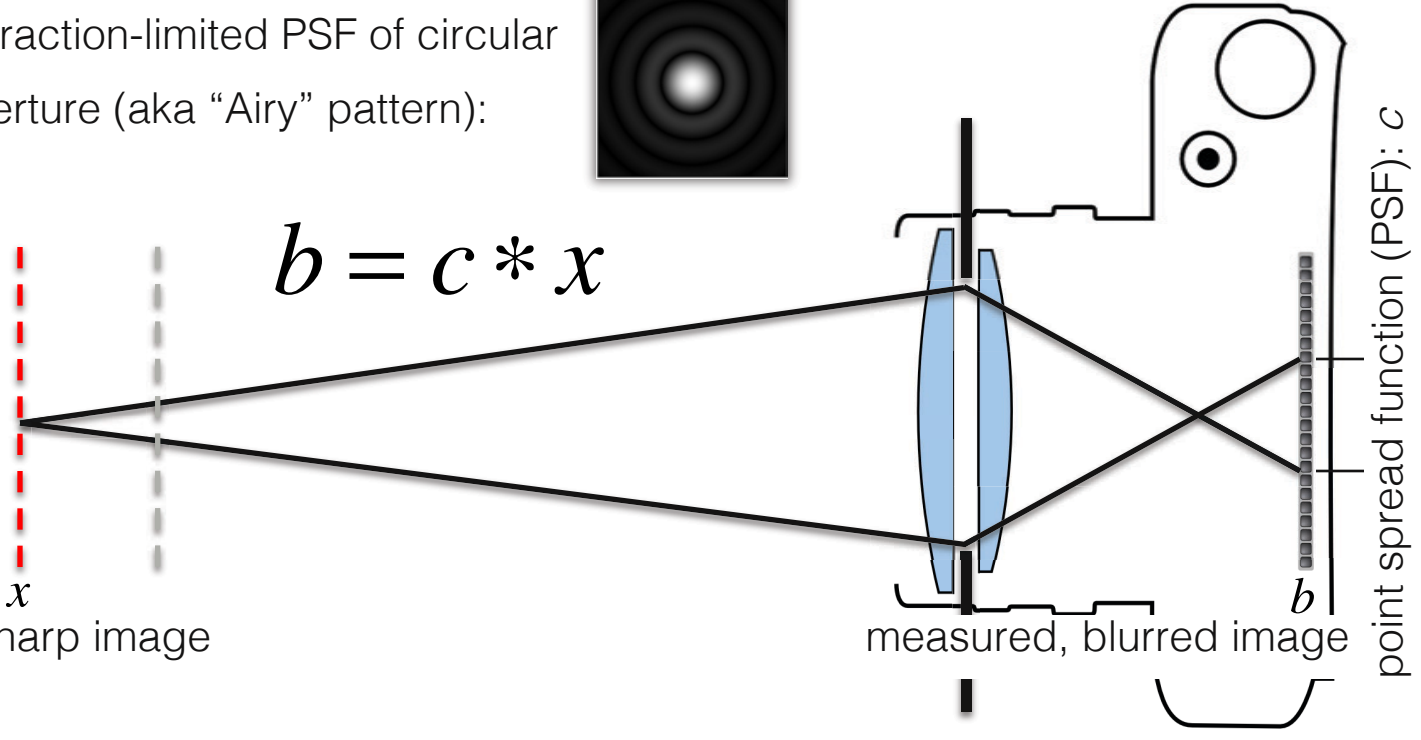
- shift-invariant convolution



Lens as Optical Low-pass Filter

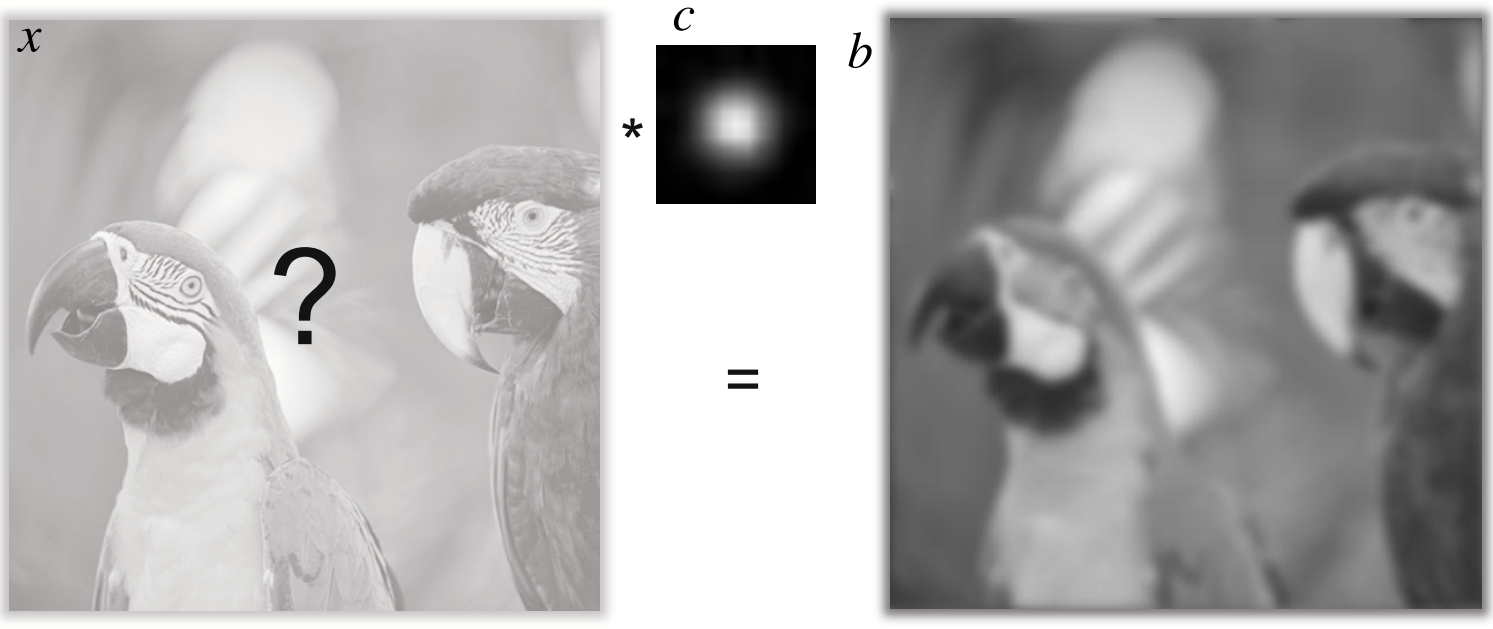
diffraction-limited PSF of circular aperture (aka “Airy” pattern):

$$b = c * x$$



Deconvolution

- given measurements *b* and convolution kernel *c*, what is *x*?



Deconvolution with Inverse Filtering

- naive solution: apply inverse kernel

$$\tilde{x} = c^{-1} * b = F^{-1}\left\{\frac{F\{b\}}{F\{c\}}\right\}$$

x

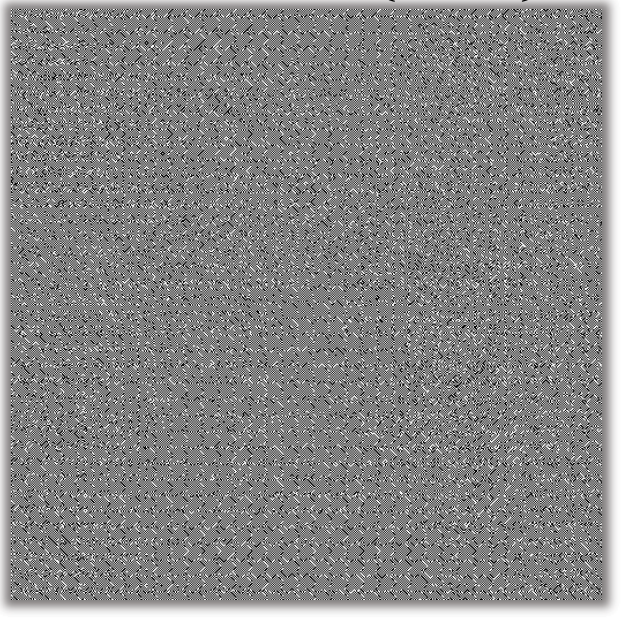
$\tilde{x}$

Deconvolution with Inverse Filtering & Noise

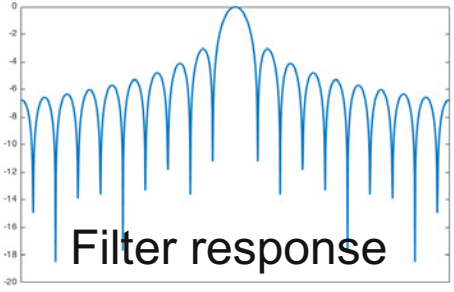
- naive solution: apply inverse kernel $\tilde{x} = c^{-1} * b = F^{-1}\left\{\frac{F\{b\}}{F\{c\}}\right\}$

- Gaussian noise, $\sigma = 0.05$

$\tilde{x}$



Deconvolution with Inverse Filtering & Noise



- results: terrible!
- why? this is an ill-posed problem (division by (close to) zero in frequency domain) → noise is drastically amplified!
- need to include regularization or image prior(s)

Deconvolution with Wiener Filtering

- apply inverse kernel and don't divide by 0

$$\tilde{x} = F^{-1}\left\{\boxed{\frac{\left|F\{c\}\right|^2}{\left|F\{c\}\right|^2 + 1/SNR}} \cdot \frac{F\{b\}}{F\{c\}}\right\}$$

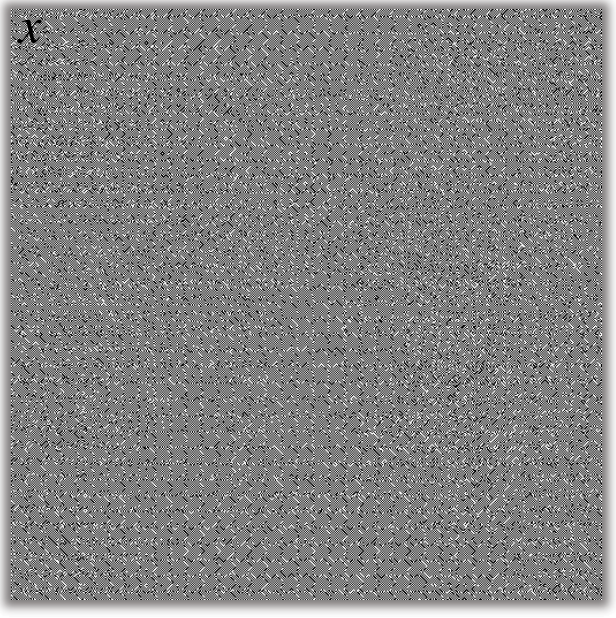
amplitude-dependent

damping factor!

$$SNR = \frac{mean\ signal \approx 0.5}{noise\ std = \sigma}$$

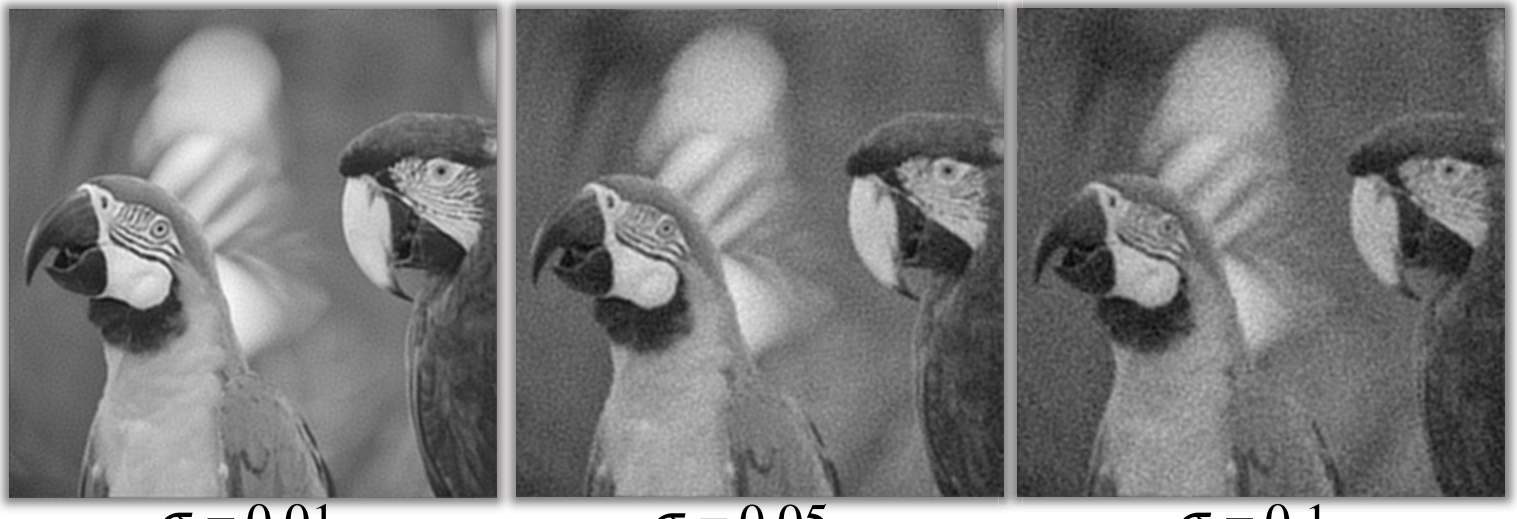
Deconvolution with Wiener Filtering

Naïve inverse filter



Wiener

Deconvolution with Wiener Filtering



$\sigma = 0.01$ $\sigma = 0.05$ $\sigma = 0.1$

Deconvolution with Wiener Filtering

- results: not too bad, but noisy

- need more advance image priors to solve this ill-posed inverse problem robustly → more in week 6

Sampling & Deconvolution – Summary

- Shannon-Nyquist theorem: always sample signal at a sampling rate >= 2*highest frequency of signal!
- if Shannon-Nyquist is violated, aliasing occurs
- aliasing cannot be corrected digitally in post-processing (see optical anti-aliasing filter)
- PSF is usually a low-pass filter, so deconvolution is an ill-posed inverse problem ☹

Matrices and Linear Systems – Review

- basic linear algebra, review if necessary!
- stanford.edu/ee263 – lecture slides and recorded lectures online
- brief review now

Matrices and Linear Systems – Review

- most computational imaging problems are linear
- geometric optics approximation of light is linear in intensity
- not necessarily true for wave-based models (e.g. interference, phase retrieval, …) – week 7

Matrices and Linear Systems – Review

- most computational imaging problems are linear

$$b = Ax$$

blurry, noisy, or otherwise corrupted measurements

unknown image

matrix modeling image formation, usually known

Matrices and Linear Systems – Review

- common problem: given b, what can I hope to recover?
- answer: analyze matrix via condition number, rank, SVD → please review this concepts

$$b = Ax$$

blurry, noisy, or otherwise corrupted measurements

unknown image

matrix modeling image formation, usually known

Matrices and Linear Systems – Review

- other common problem: given b, what is x?
- answer: invert matrix?

$$b = Ax$$

$$x_{est} \stackrel{?}{=} A^{-1}b$$

Matrices and Linear Systems – Review

- other common problem: given b, what is x?
- answer: invert matrix – generally not!

$$b = Ax$$

$$\cancel{x_{est} \overset{?}{=} A^{-1}b}$$

Linear Systems

- <u>problem 1</u>: matrix inverse only defined for square, full-rank matrices – most imaging problems are NOT!

- <u>problem 2</u>: most imaging problems deal with really big matrices – couldn't compute inverse, even if there was one!

- solution: iterative (convex) optimization

Linear Systems

- case 1: over-determined system = more measurements than unknowns

$$A \in \mathbb{R}^{m \times n}, m > n$$

- case 2: under-determined system = fewer measurements than unknowns

$$A \in \mathbb{R}^{m \times n}, m < n$$

→ more in week 6!

Linear Systems

- case 1: over-determined system = more measurements than unknowns

$$A \in \mathbb{R}^{m \times n}, m > n$$

- formulate least-squared error objective function:

$$\underset{x}{\text{minimize}} \; \frac{1}{2}\|b - Ax\|_2^2 \qquad \|r\|_2^2 = \sum_i r_i^2, \quad r = b - Ax$$

ℓ_2 norm

residual

Linear Systems

- least squares solution: gradient of objective = 0

- gradient:

$$\nabla_x \frac{1}{2}\left\|b - Ax\right\|_2^2 = \nabla_x \frac{1}{2}\left(b^T b - 2b^T Ax + x^T A^T Ax\right) = A^T Ax - A^T b$$

- equate to zero – normal equations:

$$A^T Ax = A^T b$$

Linear Systems

- least squares solution: gradient of objective = 0

- gradient:

$$\nabla_x \frac{1}{2}\|b - Ax\|_2^2 = \nabla_x \frac{1}{2}\left(b^T b - 2b^T Ax + x^T A^T Ax\right) = A^T Ax - A^T b$$

- equate to zero – normal equations:

$$A^T Ax = A^T b$$

$$\boxed{A^T(Ax - b) = 0}$$

The residual is “normal” to the columns of A

Linear Systems

- closed-form solution to normal equations:

$$A^T A x = A^T b \quad \longrightarrow \quad x_{est} = \left(A^T A\right)^{-1} A^T b$$

- rarely applicable, because again A is big and usually not full rank

- regularized solution $\quad x_{est} = \left(A^T A + \lambda I\right)^{-1} A^T b$

  (always full rank, but still too big to directly invert)

Linear Systems – Gradient Descent

- solve with iterative method, easiest one: gradient descent

$$\left(\underbrace{A^T A + \lambda I}_{\tilde{A}}\right) x = \underbrace{A^T b}_{\tilde{b}}$$

- use the negative gradient of objective as descent direction at iteration *k*, with step length α

$$x^{(k+1)} = x^{(k)} - \alpha \nabla_x = x^{(k)} - \alpha \tilde{A}^T \left(\tilde{A} x^{(k)} - \tilde{b}\right)$$

Linear Systems – Gradient Descent

- use the negative gradient of objective as descent direction at iteration *k, with* step length α

$$x^{(k+1)} = x^{(k)} - \nabla_x = x^{(k)} - \alpha A^T\left(Ax^{(k)} - b\right)$$

- for large-scale problems, implement as function handles!

Linear Systems – Gradient Descent

- back to convolution example:

$$x^{(k+1)} = x^{(k)} - \nabla_x = x^{(k)} - \alpha A^T\left(Ax^{(k)} - b\right)$$
$$= x^{(k)} - \alpha\left(c^* * \left(c * x^{(k)} - b\right)\right)$$

- efficient implementation using convolution theorem:

$$x^{(k+1)} = x^{(k)} - \alpha\, F^{-1}\left\{F\{c\}^* \cdot \left(F\{c\} \cdot F\left\{x^{(k)}\right\} - F\{b\}\right)\right\}$$

Linear Systems – Stochastic Gradient Descent

$$b = Ax$$

- What if our measurements are too large to store in memory?
- Can happen for linear models—very common for nonlinear models (neural networks)!
- Will see more on this in week 5

Linear Systems – Stochastic Gradient Descent

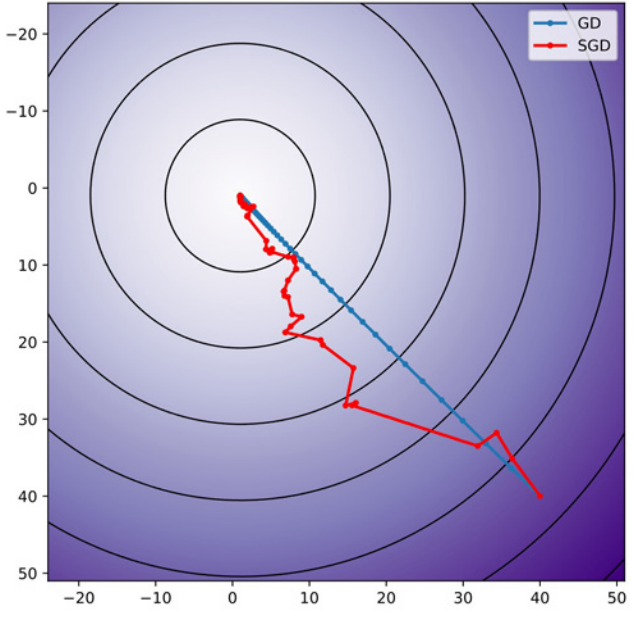
$$b = Ax$$

- Solution?
  - Stochastic optimization by sampling entries/rows from b and A at each iteration

$$\tilde{b} = \tilde{A}x$$

$$x^{(k+1)} = x^{(k)} - \alpha \tilde{A}^{(k)T}(\tilde{A}^{(k)}x^{(k)} - \tilde{b}^{(k)})$$

Linear Systems – Stochastic Gradient Descent



Tradeoffs

GD is expensive

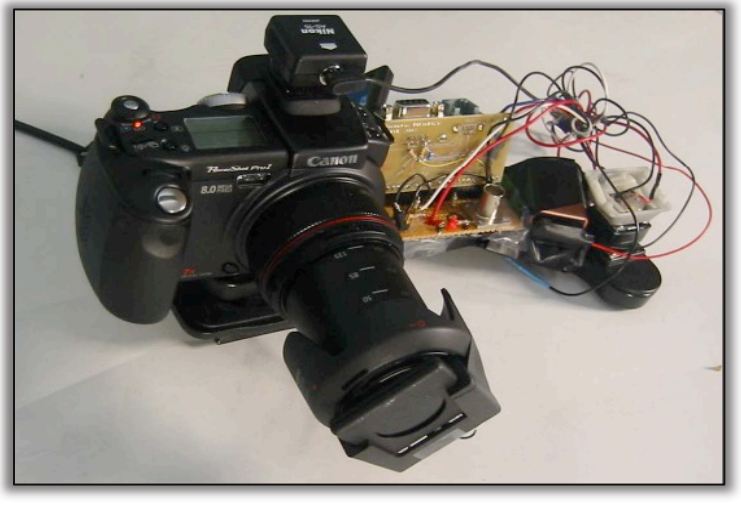
- but better convergence

SGD is more efficient

- works well far from minima
- but struggles close to minima
- can be good for non-convex problems!

Next: Computational Photography

HDR Imaging &
Tone Mapping



Coded Apertures

References and Further Reading

- Boreman, “Modulation Transfer Function in Optical and ElectroOptical Systems”, SPIE Publications, 2001
- http://www.imagemagick.org/Usage/fourier/
- wikipedia