Diagnostic Medical Image Processing
Reconstruction – Parallel Beam Reconstruction: Practical Aspects

WS 2015/2016

Andreas Maier, Joachim Hornegger, Markus Kowarschik
Pattern Recognition Lab (CS 5)

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

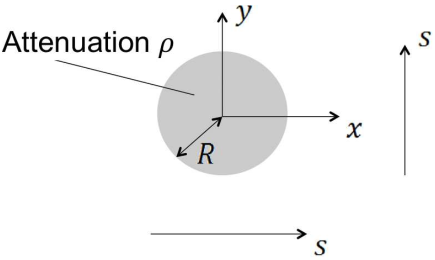
Topics

How to Implement a Parallel Beam Algorithm

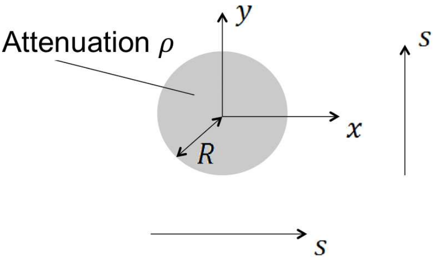
On Noise, Filtering and Window Functions

Sinograms

Example: Homogeneous Cylinder (My First Phantom)



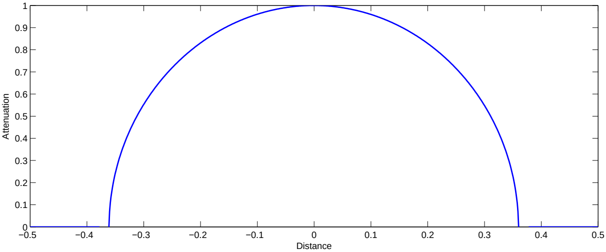
Example: Homogeneous Cylinder (My First Phantom)



- Disc is in the center $\rightarrow$ Projection is the same in all views:

$$p(s) = \begin{cases} 2\rho\sqrt{R^2 - s^2} & s \leq R \\ 0 & s > R \end{cases}$$

Example: Homogeneous Cylinder (2)



Filtered Backprojection – Practical Algorithm

- Apply Filter on the detector row:

$$q(s,\theta) = h(s) * p(s,\theta)$$

$$h(s) = \int_{-\infty}^{\infty} |\omega| e^{2\pi i\omega s} \mathrm{d}\omega$$

- Backproject $q(s,\theta)$:

$$f(x,y) = \int_{0}^{\pi} q(s,\theta)|_{s=x\cos\theta+y\sin\theta} \mathrm{d}\theta$$

Discrete Spatial Form of the Ramp Filter

- Find the inverse Fourier transform of $|\omega|$
- Set cut-off frequency of the ramp filter at $\omega = \frac{1}{2}$

Discrete Spatial Form of the Ramp Filter

- Find the inverse Fourier transform of $|\omega|$
- Set cut-off frequency of the ramp filter at $\omega = \frac{1}{2}$

$$h(s) = \int_{-\frac{1}{2}}^{\frac{1}{2}} |\omega| e^{2\pi i \omega s} \mathrm{d}\omega$$

Discrete Spatial Form of the Ramp Filter

- Find the inverse Fourier transform of $|\omega|$
- Set cut-off frequency of the ramp filter at $\omega = \frac{1}{2}$

$$h(s) = \int_{-\frac{1}{2}}^{\frac{1}{2}} |\omega| e^{2\pi i \omega s} \mathrm{d}\omega$$

$$h(s) = \frac{1}{2} \quad \frac{\sin \pi s}{\pi s} - \frac{1}{4} \left[\frac{\sin\left(\frac{\pi s}{2}\right)}{\frac{\pi s}{2}} \right]^2$$

Discrete Spatial Form of the Ramp Filter (2)

$$h(s) = \frac{1}{2} \quad \frac{\sin \pi s}{\pi s} - \frac{1}{4}\left[\frac{\sin\left(\frac{\pi s}{2}\right)}{\frac{\pi s}{2}}\right]^2$$

Discrete Spatial Form of the Ramp Filter (2)

$$h(s) = \frac{1}{2} \frac{\sin \pi s}{\pi s} - \frac{1}{4}\left[\frac{\sin\left(\frac{\pi s}{2}\right)}{\frac{\pi s}{2}}\right]^2$$

- Convert to discrete form: Let $s = n$ (integer)

$$h(n) = \begin{cases} \frac{1}{4} & n = 0 \\ 0 & n \text{ even} \\ -\frac{1}{n^2\pi^2} & n \text{ odd} \end{cases}$$

Discrete Spatial Form of the Ramp Filter (2)

$$h(s) = \frac{1}{2} \frac{\sin \pi s}{\pi s} - \frac{1}{4} \left[\frac{\sin\left(\frac{\pi s}{2}\right)}{\frac{\pi s}{2}} \right]^2$$

- Convert to discrete form: Let $s = n$ (integer)

$$h(n) = \begin{cases} \frac{1}{4} & n = 0 \\ 0 & n \text{ even} \\ -\frac{1}{n^2 \pi^2} & n \text{ odd} \end{cases}$$

- Also known as the “Ramachandran-Lakshminarayanan” convolver or “Ram-Lak” convolver

Discrete Spatial Form of the Ramp Filter (3)

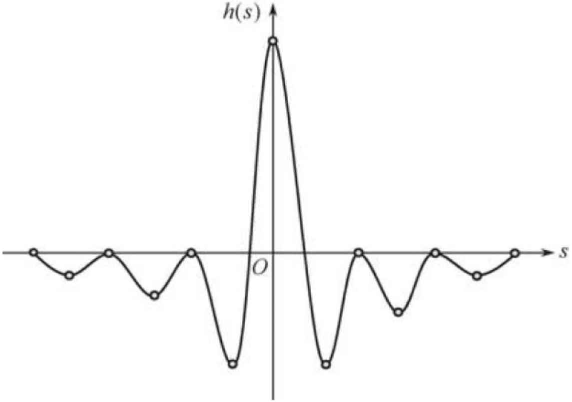


Image: Zeng, 2009

Discrete Spatial vs. Continuous Frequency Version

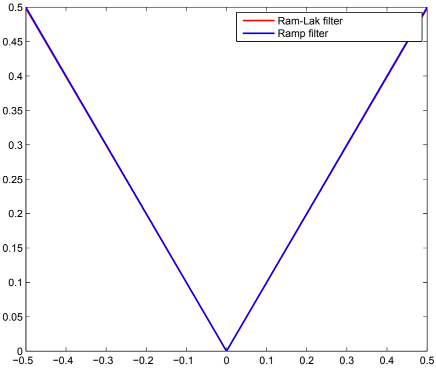
- Continuous frequency representation of the ramp filter:

$$H(\omega) = |\omega|$$

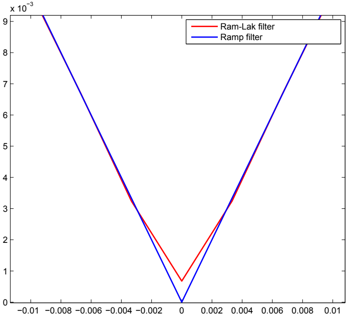
- Discrete spatial form:

$$h(n) = \begin{cases} \frac{1}{4} & n = 0 \\ 0 & n \text{ even} \\ -\frac{1}{n^2\pi^2} & n \text{ odd} \end{cases}$$

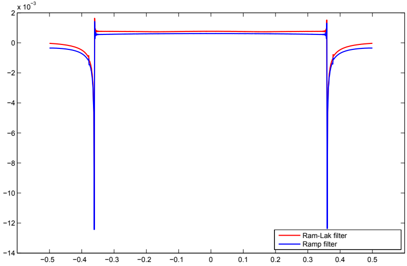
Discrete Spatial vs. Continuous Frequency Version (2)



Discrete Spatial vs. Continuous Frequency Version (3)



Example: Homogeneous Cylinder after Filter



Practical Algorithm - Filtering

- Precompute filter h(s) in spatial domain - $O(N)$

Practical Algorithm - Filtering

- Precompute filter h(s) in spatial domain - $O(N)$
- Transform filter to frequency domain $H(\omega)$ via FFT - $O(N \log N)$

Practical Algorithm - Filtering

- Precompute filter h(s) in spatial domain - $O(N)$
- Transform filter to frequency domain $H(\omega)$ via FFT - $O(N \log N)$
- For each of $\#P$ projections:

Practical Algorithm - Filtering

- Precompute filter h(s) in spatial domain - $O(N)$
- Transform filter to frequency domain $H(\omega)$ via FFT - $O(N \log N)$
- For each of $\#P$ projections:
  - Compute FFT of $p(s, \theta) - O(N \log N)$

Practical Algorithm - Filtering

- Precompute filter h(s) in spatial domain - $O(N)$
- Transform filter to frequency domain $H(\omega)$ via FFT - $O(N \log N)$
- For each of $\#P$ projections:
  - Compute FFT of $p(s, \theta) - O(N \log N)$
  - Apply filter $P(\omega, \theta) \cdot H(\omega) - O(N)$

Practical Algorithm - Filtering

- Precompute filter h(s) in spatial domain - $O(N)$
- Transform filter to frequency domain $H(\omega)$ via FFT - $O(N \log N)$
- For each of $\#P$ projections:
    - Compute FFT of $p(s, \theta) - O(N \log N)$
    - Apply filter $P(\omega, \theta) \cdot H(\omega) - O(N)$
    - Compute filtered projection $q(s)$ via iFFT - $O(N \log N)$

Practical Algorithm - Filtering

- Precompute filter h(s) in spatial domain - $O(N)$
- Transform filter to frequency domain $H(\omega)$ via FFT - $O(N \log N)$
- For each of $\#P$ projections:
  - Compute FFT of $p(s, \theta) - O(N \log N)$
  - Apply filter $P(\omega, \theta) \cdot H(\omega) - O(N)$
  - Compute filtered projection $q(s)$ via iFFT - $O(N \log N)$
- Total complexity:

$$O(N + N \log N + \#P(N + 2N \log N)) = O(\#P\ N \log N)$$

Practical Algorithm - Backprojection

- Initialize $f(x, y) = 0 - O(N^2)$

Practical Algorithm - Backprojection

- Initialize $f(x, y) = 0 - O(N^2)$
- For each of $N \times N$ pixels:

Practical Algorithm - Backprojection

- Initialize $f(x, y) = 0 - O(N^2)$
- For each of $N \times N$ pixels:
  - For each of $\#P$ projections:

Practical Algorithm - Backprojection

- Initialize $f(x, y) = 0 - O(N^2)$
- For each of $N \times N$ pixels:
  - For each of $\#P$ projections:
    - Compute $s = x \cos\theta + y \sin\theta - O(1)$

Practical Algorithm - Backprojection

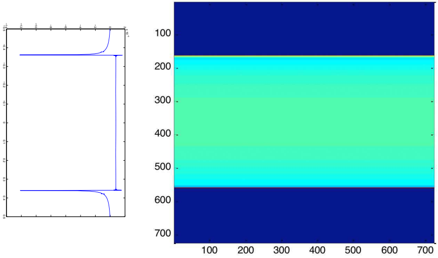
- Initialize $f(x, y) = 0 - O(N^2)$
- For each of $N \times N$ pixels:
  - For each of $\#P$ projections:
    - Compute $s = x \cos\theta + y \sin\theta - O(1)$
    - Update $f(x, y) + = q(s, \theta) - O(1)$

Practical Algorithm - Backprojection

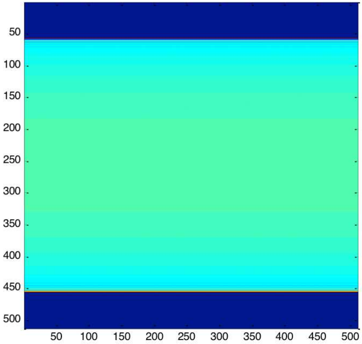
- Initialize $f(x, y) = 0 - O(N^2)$
- For each of $N \times N$ pixels:
  - For each of $\#P$ projections:
    - Compute $s = x\cos\theta + y\sin\theta - O(1)$
    - Update $f(x, y) += q(s, \theta) - O(1)$
- Total complexity:

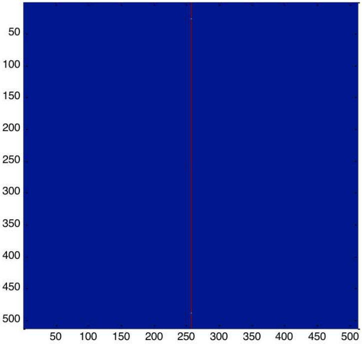
$$O(N^2 + N^2\#P(1 + 1)) = O(N^2\#P)$$

Backprojection Example

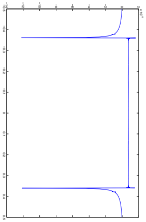


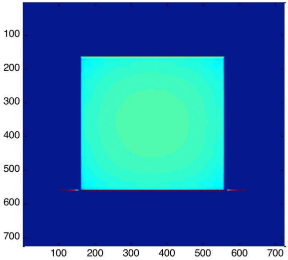
Backprojection and Fourier Slice Theorem

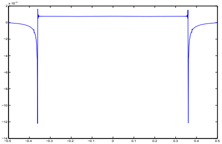




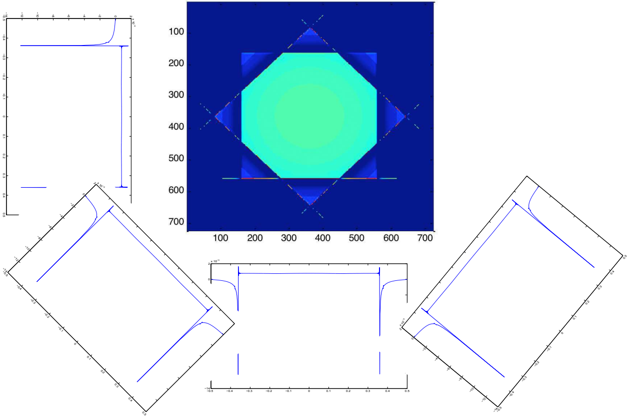
Backprojection Example



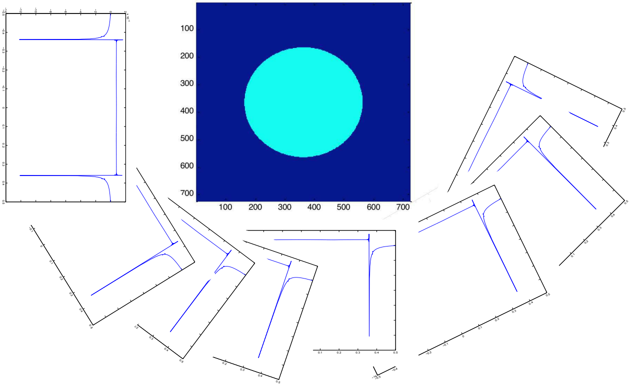




Backprojection Example



Backprojection Example



Filtered Backprojection – Practical Algorithm

- Apply Filter on the detector row:

$$O(\#P\ N \log N)$$

- Backproject:

$$O(\#P\ N^2)$$

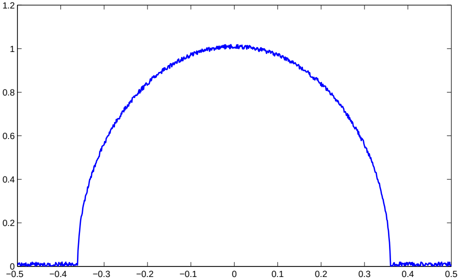
Topics

How to Implement a Parallel Beam Algorithm

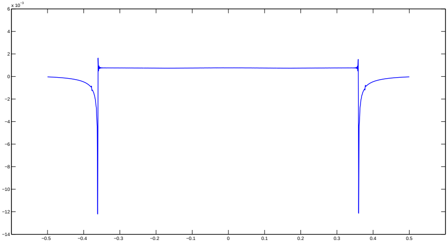
On Noise, Filtering and Window Functions

Sinograms

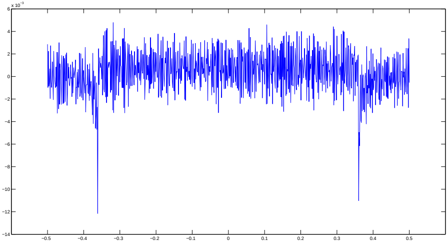
Additive Noise (+2%)



Additive Noise – After Filtering



Additive Noise – After Filtering



Window Functions

- Window functions are used to improve signals
  - High frequencies are reduced
    - $\rightarrow$ Noise reduction
    - $\rightarrow$ Reduces high frequencies caused by cutting
  - Many window functions are known:
    - Cosine Window
    - Shepp-Logan Window
    - ...

Window Functions (2)

- Apply window function in frequency domain:

$$P'(\omega, \theta) = W(\omega) \cdot P(\omega, \theta)$$

Window Functions (2)

- Apply window function in frequency domain:

$$P'(\omega, \theta) = W(\omega) \cdot P(\omega, \theta)$$

- Then apply filter:

$$Q'(\omega, \theta) = H(\omega) \cdot P'(\omega, \theta)$$

Window Functions (2)

- Apply window function in frequency domain:

$$P'(\omega, \theta) = W(\omega) \cdot P(\omega, \theta)$$

- Then apply filter:

$$Q'(\omega, \theta) = H(\omega) \cdot P'(\omega, \theta)$$

$$Q'(\omega, \theta) = H(\omega) \cdot W(\omega) \cdot P(\omega, \theta)$$

Window Functions (2)

- Apply window function in frequency domain:

$$P'(\omega, \theta) = W(\omega) \cdot P(\omega, \theta)$$

- Then apply filter:

$$Q'(\omega, \theta) = H(\omega) \cdot P'(\omega, \theta)$$

$$Q'(\omega, \theta) = H(\omega) \cdot W(\omega) \cdot P(\omega, \theta)$$

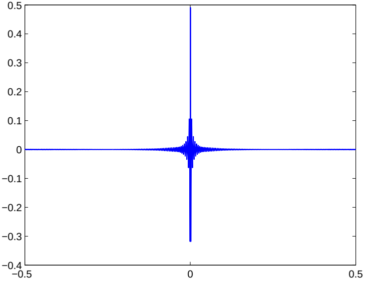
- Rewrite filtering equation to adjusted filter:

$$Q'(\omega, \theta) = H'(\omega) \cdot P(\omega, \theta)$$

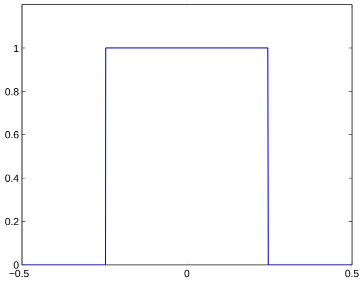
$$H'(\omega) = H(\omega) \cdot W(\omega)$$

Rectangular Window

Spatial Domain

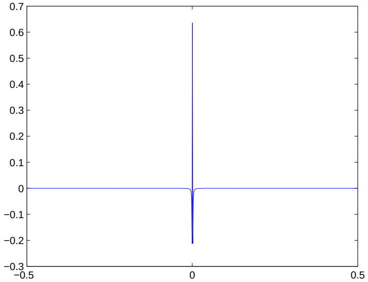


Frequency Domain

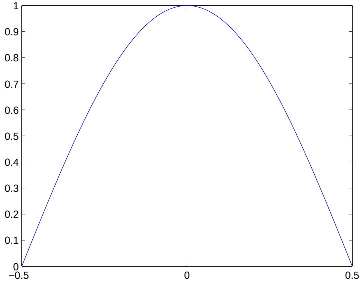


Cosine Window – $\cos(\pi \cdot x)$

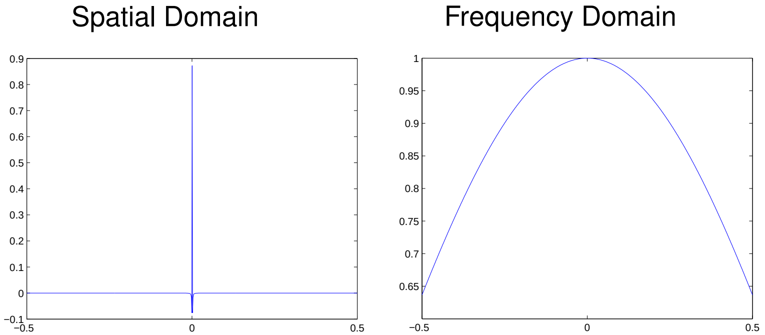
Spatial Domain



Frequency Domain

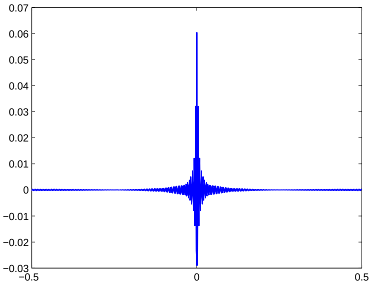


Shepp-Logan Window – $\dfrac{\sin(\pi \cdot x)}{(\pi \cdot x)}$

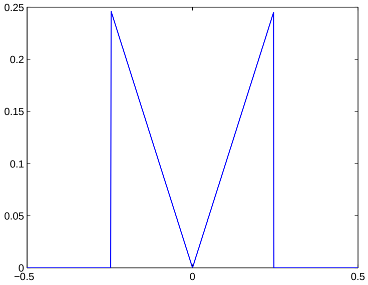


Rectangular Filter

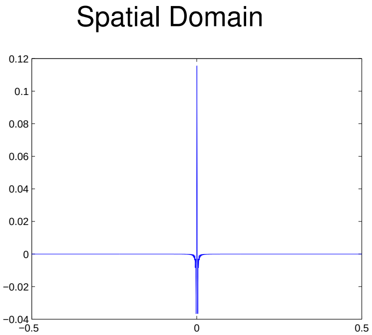
Spatial Domain

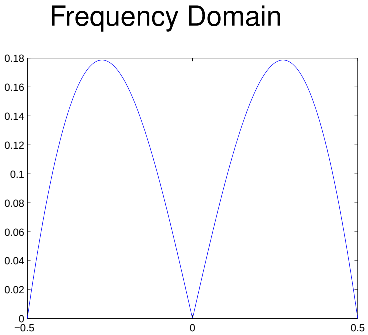


Frequency Domain



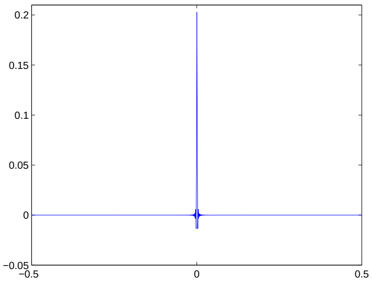
Cosine Filter



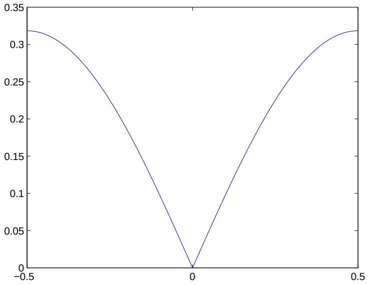


Shepp-Logan Filter

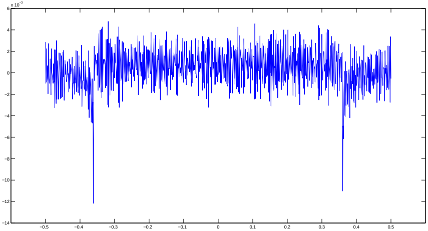
Spatial Domain



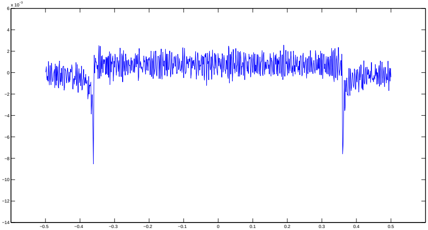
Frequency Domain



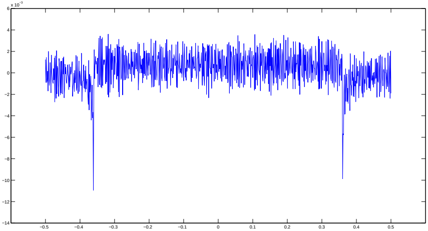
Ramp Filter Result



Shepp-Logan Filter Result



Cosine Filter Result



Noise

- Is amplified by ramp filter
- Has to be taken care of in an appropriate manner
- Is indirectly proportional to the applied dose
- Affects different reconstruction methods differently

Topics

How to Implement a Parallel Beam Algorithm

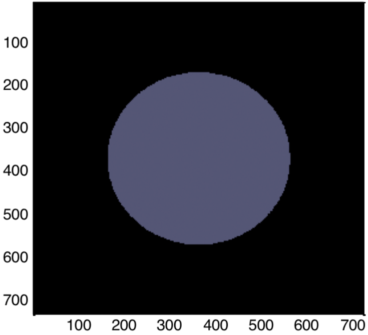
On Noise, Filtering and Window Functions

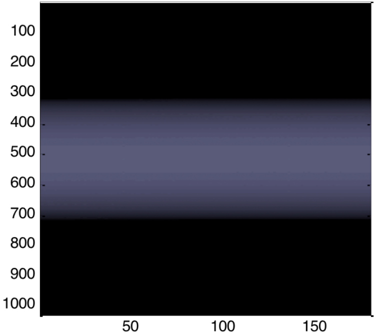
Sinograms

Sinogram

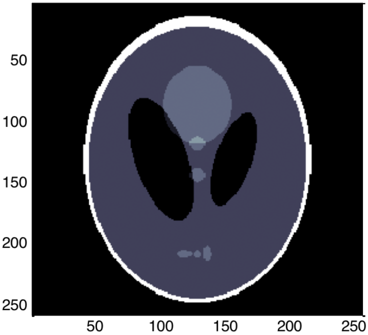
- Method to visualize all projections in one image
- Contains all information to reconstruct one slice
- Also called ‘Fanogram” in fan beam geometry
- Popular method for visualization with narrow detectors

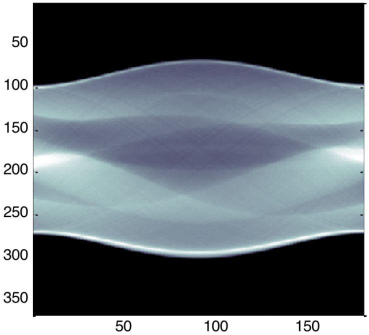
Sinogram (2)


100
200
300
400
500
600
700
100 200 300 400 500 600 700


100
200
300
400
500
600
700
800
900
1000
50 100 150

Sinogram (3)


50
100
150
200
250
50
100
150
200
250


50
100
150
200
250
300
350
50
100
150

Further Readings

- Gengsheng Lawrence “Larry” Zeng. "Medical Image Reconstruction – A Conceptual Tutorial". Springer 2009
- Ronald N. Bracewell. "The Fourier Transform and Its Applications". McGraw-Hill Publishing Company. 1999
- Thorsten M. Buzug. "Computed Tomography: From Photon Statistics to Modern Cone-Beam CT". Springer 2008

Questions?