Indian Institute of Technology
Hyderabad

Indian Institute of Technology Hyderabad

Biomedical Imaging

CT IMAGING

Abhishek Amit Raje

February 2024

1 *Sinogram of a Shepp Logan Phantom*

- A sinogram is a graphical representation of the intensity values obtained from a set of projections of an object. It is the projected sum of the 1D Fourier transforms taken about the image.
- The sinogram is represented as a 2D plot where the x-axis corresponds to the projection angle θ and the y-axis corresponds to the radial distance ρ. Each point in the sinogram represents the integral of the attenuation along a specific ray at a specific angle.

```python
import numpy as np
import matplotlib.pyplot as plt
from skimage.data import shepp_logan_phantom
from skimage.transform import radon

def generate_sinogram_with_theta(phantom, theta, cmap='viridis'):
    sinogram = radon(phantom, theta=theta, circle=True)

    plt.subplot(121)
    plt.imshow(phantom, cmap='gray')
    plt.title('Shepp-Logan Phantom')

    plt.subplot(122)
    plt.imshow(sinogram, cmap=cmap, aspect='auto', extent=[min(theta), max(theta), 0, sinogram.shape[0]])
    plt.colorbar(label='Intensity')
    plt.title('Sinogram' )

    plt.show()

# Generate Shepp-Logan phantom
phantom = shepp_logan_phantom()

# Specify the sampling angle theta (in degrees) with a sampling rate of 0.05 degrees
theta_values = np.arange(0, 180,0.05)

# Generate sinogram with specified theta values
generate_sinogram_with_theta(phantom, theta_values)
```

Listing 1: Python code for generating sinogram

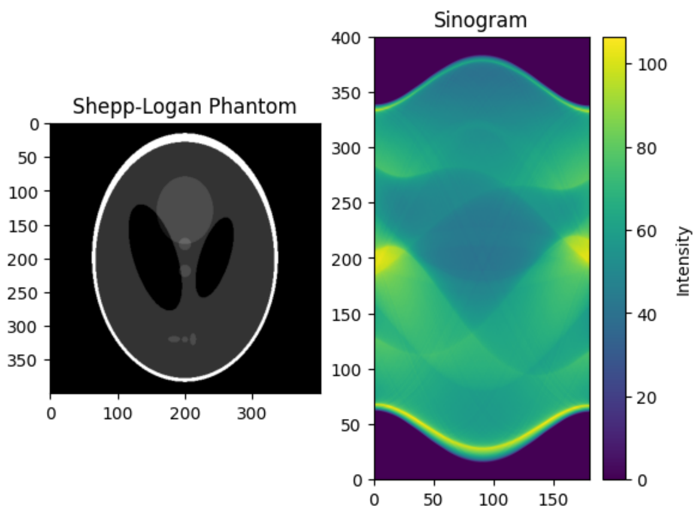


Figure 1: Sinogram at a sampling rate $\theta = 0.5$ degree

2 *Filtered Back Projection*

- Filtered back projection is a reconstruction technique commonly used in computed tomography (CT) and other medical imaging modalities.
- It is employed to convert the acquired projection data (sinogram) back into a two-dimensional image of the internal structure of an object.
- A filter is applied to the sinogram data to enhance specific frequency components and reduce artifacts in the reconstructed image.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.ndimage import convolve1d
from skimage.data import shepp_logan_phantom
from skimage.transform import radon

def generate_sinogram_with_theta(phantom, theta, cmap='viridis'):
    sinogram = radon(phantom, theta=theta, circle=True)

    # Apply Mexican Hat filter
    mexican_hat_kernel = np.array([-1, 2, -1])
    filtered_sinogram = convolve1d(sinogram, mexican_hat_kernel, axis=0, mode='constant', cval=0)

    fig, axes = plt.subplots(1, 3, figsize=(15, 5))  # 1 row, 3 columns
```

```python

    axes[0].imshow(phantom, cmap='gray')
    axes[0].set_title('Shepp-Logan Phantom')

    im1 = axes[1].imshow(sinogram, cmap=cmap, aspect='auto',
     extent=[min(theta), max(theta), 0, sinogram.shape[0]])
    axes[1].set_title('Original Sinogram')
    plt.colorbar(im1, ax=axes[1], label='Intensity')  # Add
    colorbar to the first subplot

    im2 = axes[2].imshow(filtered_sinogram, cmap=cmap,
    aspect='auto', extent=[min(theta), max(theta), 0,
    filtered_sinogram.shape[0]])
    axes[2].set_title('Sinogram with Mexican Hat Filter')
    plt.colorbar(im2, ax=axes[2], label='Intensity')  # Add
    colorbar to the second subplot

    plt.subplots_adjust(wspace=0.5)  # Adjust the width
    space between subplots

    plt.show()

# Generate Shepp-Logan phantom
phantom = shepp_logan_phantom()

# Specify the sampling angle theta (in degrees) with a
    sampling rate of 0.05 degrees
theta_values = np.arange(0, 180, 0.05)

# Generate sinogram with specified theta values and apply
    Mexican Hat filter
generate_sinogram_with_theta(phantom, theta_values)
```

Listing 2: Python code for generating sinogram with Mexican Hat filter

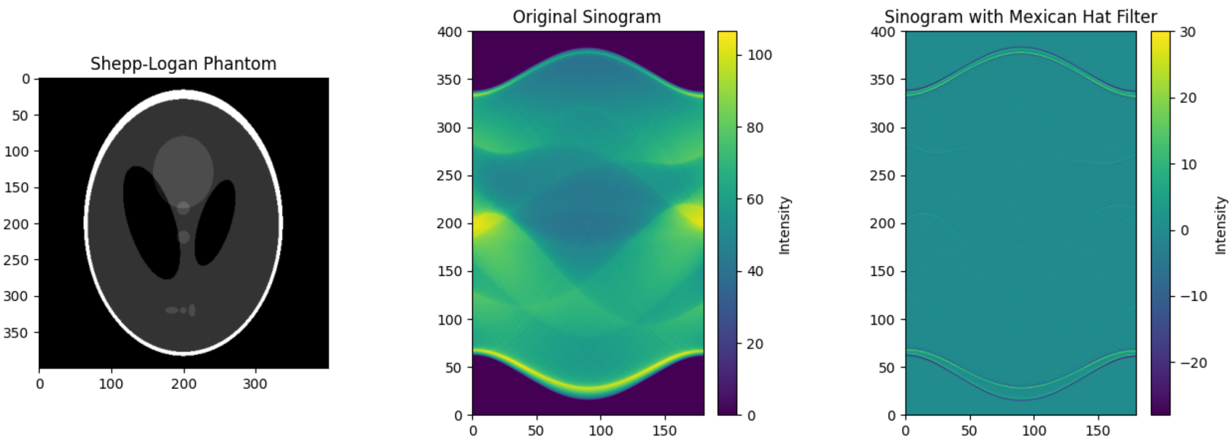


Figure 2: Singogram with Mexican Hat Filter

Comparing Properties

- As Observed the Mexican Hat filter enhances certain frequency components in the sinogram while suppressing others.
- The sinogram with the Mexican Hat filter is has sharper features
- The filtering process reduces noise in the sinogram, leading to a cleaner representation of the object's attenuation characteristics

]

3 *Reconstruction of Image*

```python
import numpy as np
import matplotlib.pyplot as plt
from skimage.data import shepp_logan_phantom
from skimage.transform import radon, iradon

def generate_sinogram_with_theta(phantom, theta, cmap='viridis'):
    sinogram = radon(phantom, theta=theta, circle=True)
    plt.subplot(121)
    plt.imshow(phantom, cmap='gray')
    plt.title('Shepp-Logan Phantom')

    plt.subplot(122)
    plt.imshow(sinogram, cmap=cmap, aspect='auto', extent=[min(theta), max(theta), 0, sinogram.shape[0]])
    plt.colorbar(label='Intensity')
    plt.title('Sinogram' )

    plt.show()

    return sinogram

def reconstruct_image_from_sinogram(sinogram, theta):
    reconstructed_image = iradon(sinogram, theta=theta, circle=True)

    plt.imshow(reconstructed_image, cmap='gray')
    plt.title('Reconstructed Image')
    plt.show()

# Generate Shepp-Logan phantom
phantom = shepp_logan_phantom()

# Specify the sampling angle theta (in degrees) with a sampling rate of 0.05 degrees
theta_values = np.arange(0, 180, 5)

# Generate sinogram with specified theta values
sinogram = generate_sinogram_with_theta(phantom, theta_values)

# Reconstruct the image from the sinogram
reconstruct_image_from_sinogram(sinogram, theta_values)
```

Listing 3: Reconstructing Sinogram using Inverse Radon Transform

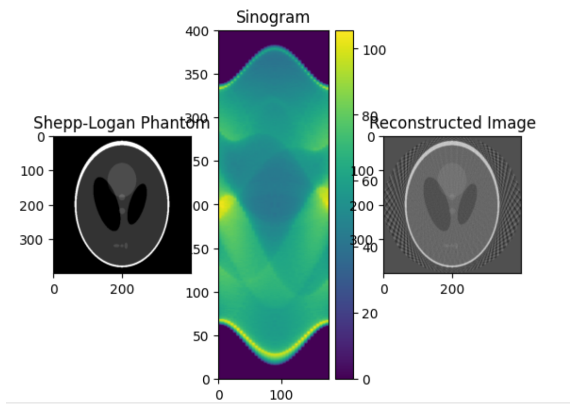


Figure 3: Reconstruction at sampling angle $\theta = 5$ degree

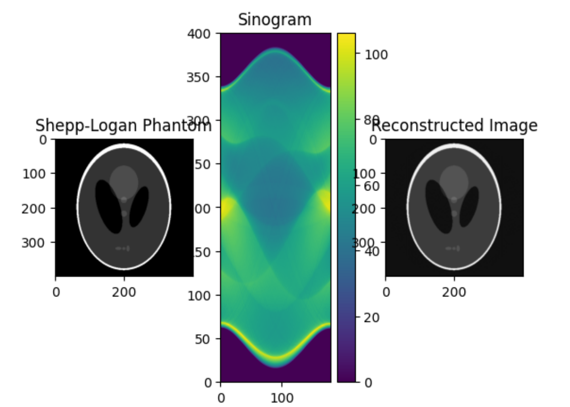


Figure 4: Reconstruction at sampling angle $\theta = 0.5$ degree i.e 360 projections

4 *Sinogram using 360-degree projection*

Effect of Sampling Rate on Image Reconstruction

- Insufficient sampling can lead to aliasing artifacts in the reconstructed image
- When the angle of sampling is too sparse, streaking artifacts may appear in the reconstructed image
- Increasing the number of projections (higher sampling rate) generally improves the resolution and sharpness of the reconstructed image.

5 *Reconstruction Using Central Slice Theorem*

```python
import numpy as np
import matplotlib.pyplot as plt
from skimage.data import shepp_logan_phantom
from skimage.transform import radon, iradon

# Generate Shepp-Logan phantom
phantom = shepp_logan_phantom()

# Perform Radon transform to obtain sinogram
sinogram = radon(phantom)

# Apply Central Slice Theorem by taking the inverse Radon
    transform
reconstructed_image = iradon(sinogram)

# Display the results
plt.figure(figsize=(10, 4))

plt.subplot(131)
plt.imshow(phantom, cmap='gray')
plt.title('Shepp-Logan Phantom')

plt.subplot(132)
plt.imshow(sinogram, cmap='viridis', aspect='auto', extent
    =[0, 180, 0, sinogram.shape[0]])
plt.title('Sinogram')

plt.subplot(133)
plt.imshow(reconstructed_image, cmap='gray')
plt.title('Reconstructed Image')

plt.show()
```

Listing 4: Reconstructing Sinogram using Central Slice Theorem

- the Central Slice Theorem states that the Fourier transform of a 1D projection of an object is equivalent to a central line in the 2D Fourier transform of the object itself.

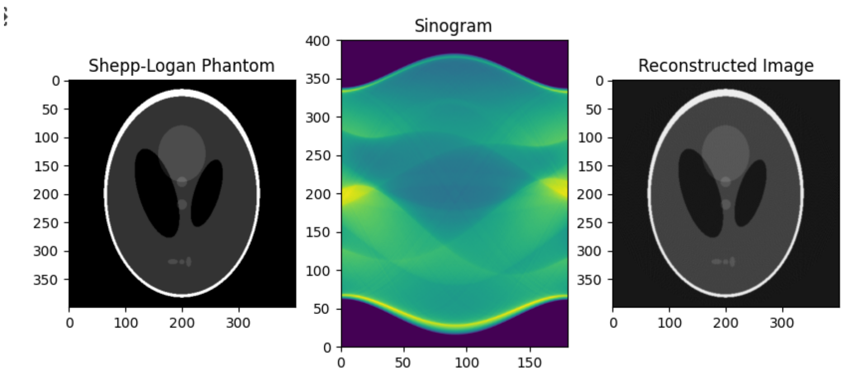


Figure 5: Reconstruction using central Slice Theorem

6 *Reconstruction After Filtering*

```
import numpy as np
import matplotlib.pyplot as plt
from skimage.data import shepp_logan_phantom
from skimage.transform import radon, iradon

def apply_filter(sinogram, filter_type):
    projections = radon(sinogram, circle=True)
    # Apply frequency domain filter
    if filter_type == 'ram-lak':
        filter_func = np.abs(np.fft.fftshift(np.fft.fft(np.
    fft.ifftshift(np.arange(-len(projections)//2, len(
    projections)//2)))))
    elif filter_type == 'shepp-logan':
        filter_func = np.abs(np.fft.fftshift(np.fft.fft(np.
    fft.ifftshift(np.sqrt(np.abs(np.arange(-len(projections)
    //2, len(projections)//2)))))))
    elif filter_type == 'cosine':
        filter_func = np.abs(np.fft.fftshift(np.fft.fft(np.
    fft.ifftshift(np.cos(np.arange(-len(projections)//2, len(
    projections)//2) * np.pi / len(projections))))))
    elif filter_type == 'hamming':
        filter_func = np.abs(np.fft.fftshift(np.fft.fft(np.
    fft.ifftshift(0.54 + 0.46 * np.cos(2 * np.pi * np.arange
    (-len(projections)//2, len(projections)//2) / len(
    projections))))))

    filtered_projections = projections * filter_func
    return filtered_projections, angles

def reconstruct_image(sinogram, angles):
    reconstructed_image = iradon(sinogram, theta=angles,
```

```python
    circle=True)
    return reconstructed_image

# Generate Shepp-Logan phantom
phantom = shepp_logan_phantom()

# Specify the sampling angle theta (in degrees)
theta_values = np.arange(0, 180, 1)

# Generate sinogram with specified theta values
sinogram = radon(phantom, theta=theta_values, circle=True)

# Define filter types
filter_types = ['ram-lak', 'shepp-logan', 'cosine', 'hamming
    ']

# Perform filtered backprojection with different filters
for filter_type in filter_types:
    filtered_sinoam, angles = apply_filter(sinogram,
    filter_type)
    reconstructed_image = reconstruct_image(
    filtered_sinogram, angles)

    # Display results
    plt.figure(figsize=(12, 4))
    plt.subplot(131)
    plt.imshow(phantom, cmap='gray')
    plt.title('Original Image')

    plt.subplot(132)
    plt.imshow(reconstructed_image, cmap='gray')
    plt.title(f'Reconstructed Image ({filter_type.capitalize
    ()} Filter)')

    plt.subplot(133)
    plt.plot(angles, filtered_sinogram.mean(axis=0), label='
    Filtered Sinogram')
    plt.legend()
    plt.title('Filtered Sinogram')

    plt.show()
```

Listing 5: Reconstructing Sinogram After Filtering

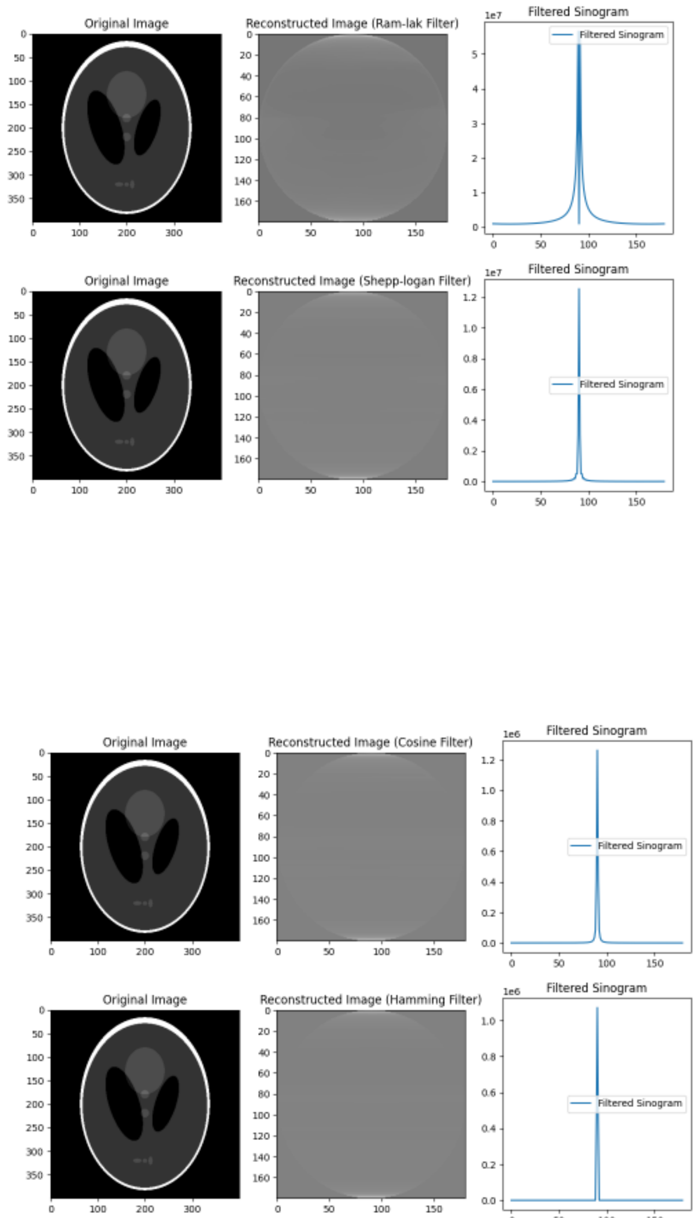


Figure 6: Reconstructing after Filtering

7 *Limitations of projection radiography*

- Projection radiography provides a two-dimensional image of the internal structures, which can lead to overlapping of structures and difficulties in accurately assessing the depth or three-dimensional relationships.
- Traditional projection radiography provides static images, and it is not suitable for real-time imaging
- Various artifacts, such as image distortion or blurring, may occur due to patient movement, equipment malfunctions, or other external factors

X-RAY IMAGING

Abhishek Amit Raje

February 2024

1 *Factor Affecting SNR*

Signal-to-Noise Ratio (SNR) in X-ray imaging is influenced by various factors

- Scatter Radiation: Scatter radiation contributes to noise in the image
- Detector Efficiency: The efficiency of the detector in converting X-ray photons
- Patient Thickness and Density: The composition and thickness of the patient's body influence the attenuation of X-rays

If the SNR doubles then dose becomes four times

2 *Beam Hardening*

- Beam hardening is a phenomenon in X-ray imaging that occurs when a polychromatic X-ray beam, which consists of a range of energy levels, passes through an object or a patient. As the X-ray beam traverses through different materials of varying thickness and composition
- In this process lower-energy X-ray photons are preferentially absorbed, leading to an increase in the average energy of the remaining X-ray beam.

3 *Aliasing,bandwidth limiting,Nyquist condition*

- **Aliasing**:Aliasing occurs when a signal is undersampled, causing high-frequency components to inaccurately fold into lower frequencies
- **Bandwidth limiting**Bandwidth limiting restricts the range of frequencies in a signal, crucial for controlling information transmission and avoiding interference.
- **Nyquist condition**The Nyquist condition sampling theorem, dictates that the sampling rate must be at least twice the maximum frequency in a signal to prevent aliasing. This condition is expressed as $f_s > 2_{max}$

4 *Percentage of X-rays are transmitted through the chest*

We know percentage of X-rays transmitted through the chest, the exponential attenuation formula:

$$I = I_o e^{-ux} \tag{1}$$

where:

- I is the intensity of the transmitted X-rays

- I_o is the initial intensity of the incident X-rays
- u is the linear attenuation coefficient
- x is the thickness of the material.

The linear attenuation coefficient u can be calculated using the half-value layer (HVL) (H) and the formula

$$u = \frac{ln2}{H} \tag{2}$$

Calculate u for muscle and bone

$$\mu_{\text{muscle}} = \frac{0.693}{3.5\,\text{cm}} \approx 0.198\,\text{cm}^{-1}$$

$$\mu_{\text{bone}} = \frac{0.693}{1.8\,\text{cm}} \approx 0.385\,\text{cm}^{-1}$$

$$I_{\text{muscle}} = I_0 \cdot e^{-\mu_{\text{muscle}} \cdot x_{\text{muscle}}} = I_0 \cdot e^{-0.198 \cdot 16}$$

$$I_{\text{bone}} = I_0 \cdot e^{-\mu_{\text{bone}} \cdot x_{\text{bone}}} = I_0 \cdot e^{-0.385 \cdot 4}$$

$$I_{\text{total}} = I_{\text{muscle}} \cdot I_{\text{bone}}$$

Percentage of X-Ray Transmitted

$$e^{-0.198} * e^{-0.385} = e^{-0.583} = 0.558$$

5 *Difference between X-Ray Images*

- Higher Effective Energy (140 keV): X-rays with higher energy penetrate tissues more effectively. This results in lower contrast in the image because the X-rays making it challenging to distinguish
- Lower Effective Energy (50 keV): Lower energy X-rays are more readily absorbed by soft tissues, leading to increased contrast.
- Higher Effective Energy (140 keV): The X-rays with higher energy can penetrate the body more deeply, reaching the detector with less attenuation.
- Lower Effective Energy (50 keV): Lower energy X-rays are more likely to be absorbed by tissues, leading to greater attenuation.

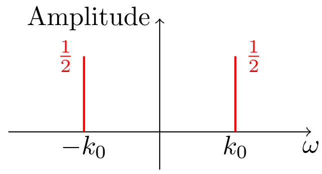
6 *Opinion on getting Xray imaging for heart related diseases.*

Pros	**Cons**
• Diagnostic Tool: X-ray imaging can provide valuable information about the structure of the heart. • Quick and Non-Invasive: X-rays are relatively quick and non-invasive compared to some other imaging modalities. • Widely Available: X-ray facilities are widely available, making it a convenient option for initial screening.	• Radiation Exposure: X-rays involve exposure to ionizing radiation, which carries potential risks. • Limited Soft Tissue Detail: X-rays are better at visualizing bones and dense structures but have limitations in capturing detailed images of soft tissues. • May Require Contrast Agents: In some cases, contrast agents may be used, and individuals may have allergies or adverse reactions to these agents.

7 *Fourier Transforms and Spectra*

a. $\cos^2(k_0 z)$

$$\mathcal{F}\{\cos^2(k_0 z)\} = \frac{1}{2}\left[\delta(\omega - k_0) + \delta(\omega + k_0)\right]$$



b. $\sin^3(k_0 z)$

$$\mathcal{F}\{\sin^3(k_0 z)\} = \frac{3}{4i}\left[\delta(\omega - 3k_0) - \delta(\omega - k_0)\right]$$