Karlsruher Institut für Technologie

Institut der Biomedizinische Technik

Biomedical Imaging for Modeling and Simulation

# Einführungspraktikum Medizintechnik
# Medical Imaging Technology

**by:**

**Yekta Ahmadi Simab** (unkho)

**Pratham Gupta** (ufuxl)

**Isabel Melisa Güldali** (ubxhe)

**Lecturer:** Prof. Dr.-Ing. M. Francesca Spadea

**Supervisor:** MSc Edoardo De Rose

**Date: 04.02.2024**

# MTF EXPERIMENT

## Theoretical Background:

### MODULATION TRANSFER FUNCTION (MTF)

The ability of an imaging system to reproduce different levels of detail or contrast in an image. It is a measure of the system's ability to transmit variations in intensity from the object to the image.

- A high MTF indicates that the system can accurately represent fine details in the images.
- A low MTF may result in a loss of detail and contrast.

It is crucial for evaluating the spatial resolution and contrast transfer properties of imaging systems.

### DISCRETE FOURIER TRANSFORM (DFT)

It is a mathematical technique used to analyse the frequency content of a discrete signal. It transforms a sequence of sampled values into its constituent frequencies.

It is used in medical imaging to analyse the frequency components of image data thus helping in tasks such as filtering, compression, and feature extraction of the image.

### FAST FOURIER TRANSFORM (FFT)

It is an algorithm that efficiently computes the DFT of a sequence or signal. It reduces the number of computations required compared to the standard DFT.

FFT is widely used in medical imaging because of the fast and efficient computation of the Fourier transform.

## Matlab Codes:

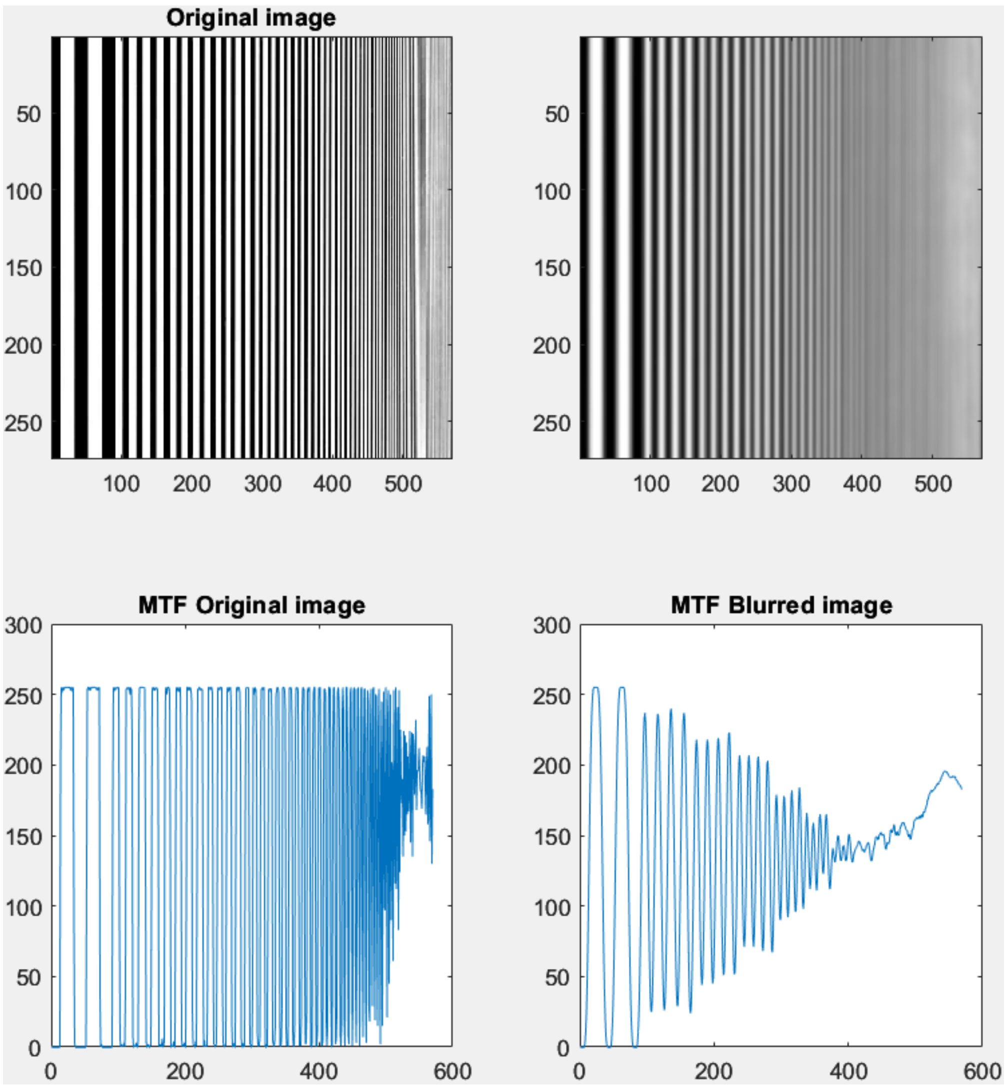
All the codes have been commented and improved.

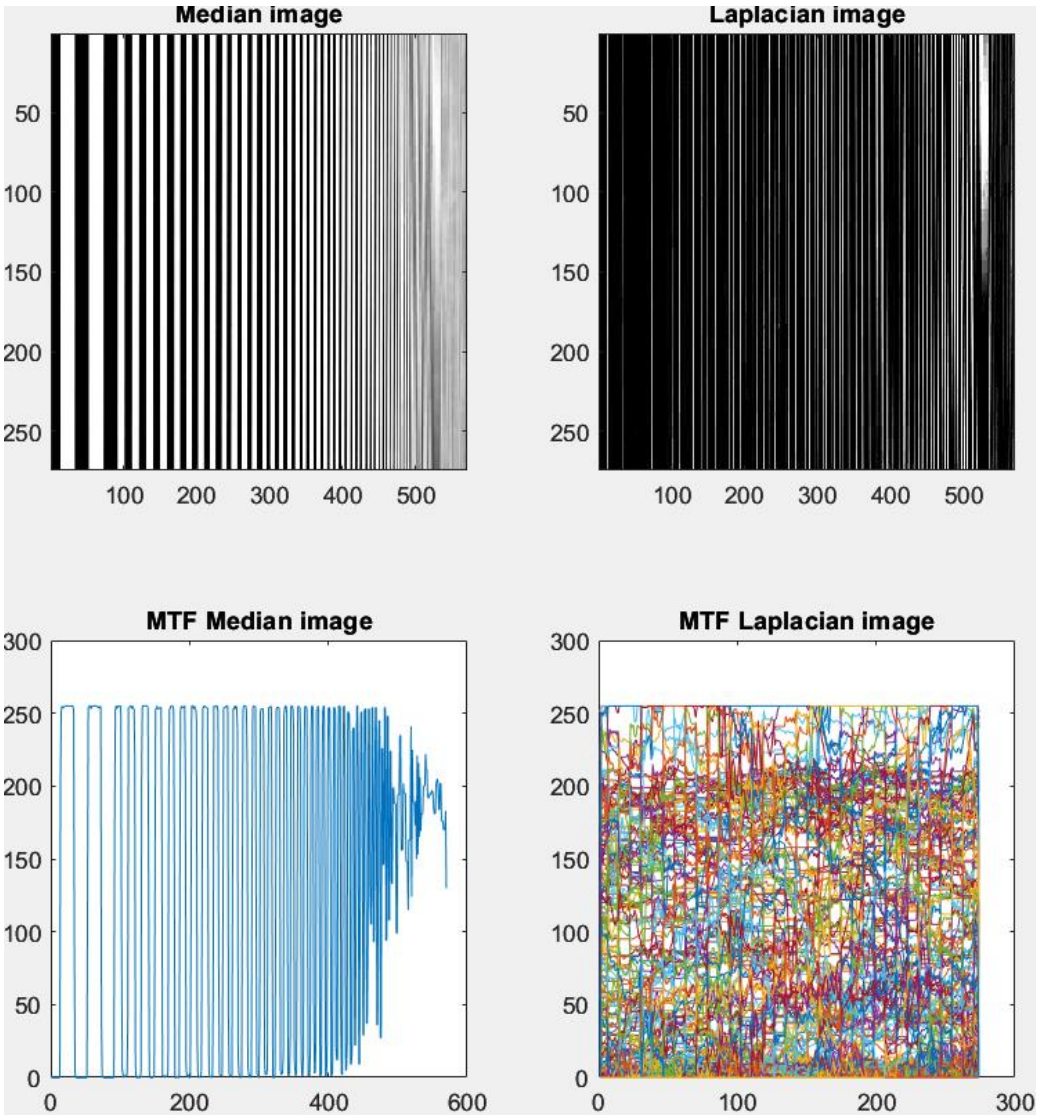
### 1) mtf.m

```matlab
%Read Image
%img = imread("pattern.tif");
img = imread("printedpattern.jpg");
%Display Original Image
figure();
subplot(2,4,1);
imagesc(img);
title('Original image');
colormap("gray");

%Extract and Plot Central Row
[rows, cols, chans] = size(img);
central_row = uint8(rows/2);

mtf_img = img(central_row, :);
subplot(2,4,5);
plot(mtf_img);
title('MTF Original image');

%Apply Gaussian Blur to Original Image:
img_blurred = imgaussfilt(img, 3);
subplot(2,4,2);
imagesc(img_blurred);
title('Blurred image');
colormap("gray");
imagesc(img_blurred);

%Extract and Plot Central Row of Blurred Image
mtf_img_blurred = img_blurred(central_row, :);
subplot(2,4,6);
plot(mtf_img_blurred);
title('MTF Blurred image');

%Apply Median Filter to Original Image
img_median = medfilt2(img,[5,5]);
subplot(2,4,3);
imagesc(img_median);
title('Median image');
colormap("gray");

%Extract and Plot Central Row of Median Filtered Image
mtf_img_median = img_median(central_row, :);
subplot(2,4,7);
plot(mtf_img_median);
title('MTF Median image');

h = fspecial('laplacian');
img_laplacian = imfilter(img, h);
subplot(2,4,4);
imagesc(img_laplacian);
title('Laplacian image');
colormap("gray");

mtf_img_laplacian = img_laplacian(central_row, :);
subplot(2,4,8);
plot(img_laplacian);
title('MTF Laplacian image');
```

## Observations:

- **Filters used in the code:**
  - **Gaussian Filter:** Gaussian blurring is a common image processing technique used to reduce noise and detail in an image while preserving its overall structure.
  - **Median Filter:** The median filter is a non-linear filter that replaces each pixel value with the median value of the pixels in its neighbourhood.
  - **Laplacian Filter:** Laplacian filter enhances regions of rapid intensity change, which often correspond to edges in the image. The result of applying a Laplacian filter is an image that highlights areas where the intensity changes abruptly.
- **Results:**
  - **Original Version:**

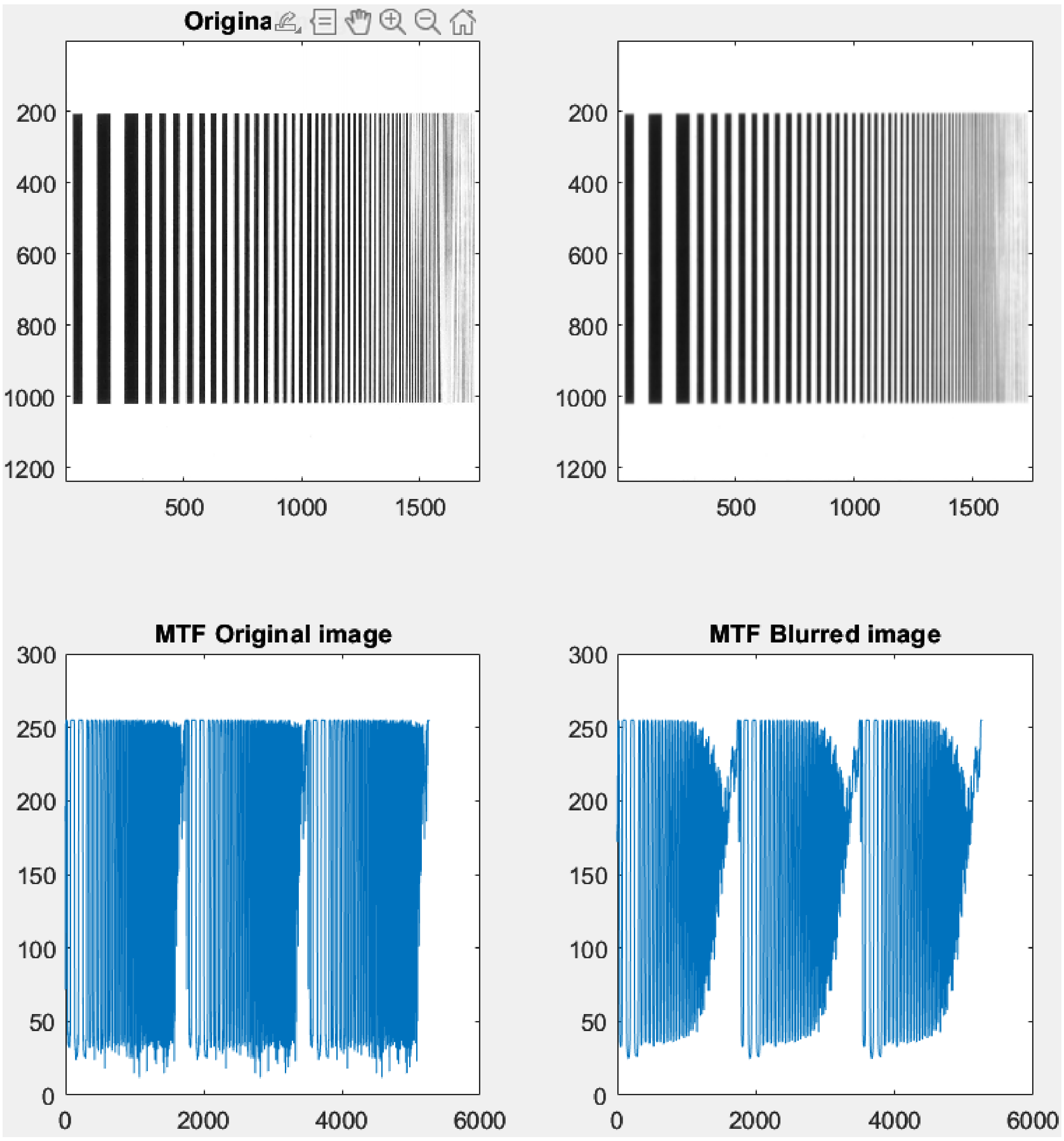




In the examination of various filters applied to the original image, distinct MTF outputs have been obtained:

1. In **Gaussian Filter**: High frequencies have been suppressed, resulting in reduced noise and reduced detail in the transformed image.
2. In **Median Filter**: The transformation is not significantly different, given the image's symmetry. However, at high values on the x-axis, a loss in MTF is noticeable.
3. In **Laplacian Filter:** The filtered image is observed only up to 300 due to computer limitations, indicating a constrained output range.

- **Scanned Version:**



When dealing with a lower pixelated scanned version of the same image, we observe distinct outcomes:

1. The Gaussian Blurred image exhibits some resemblance to the original image since the original image undergoes blurring during the printing and subsequent scanning process.
2. Other filters fail to yield expected results due to the low resolution of the provided image.

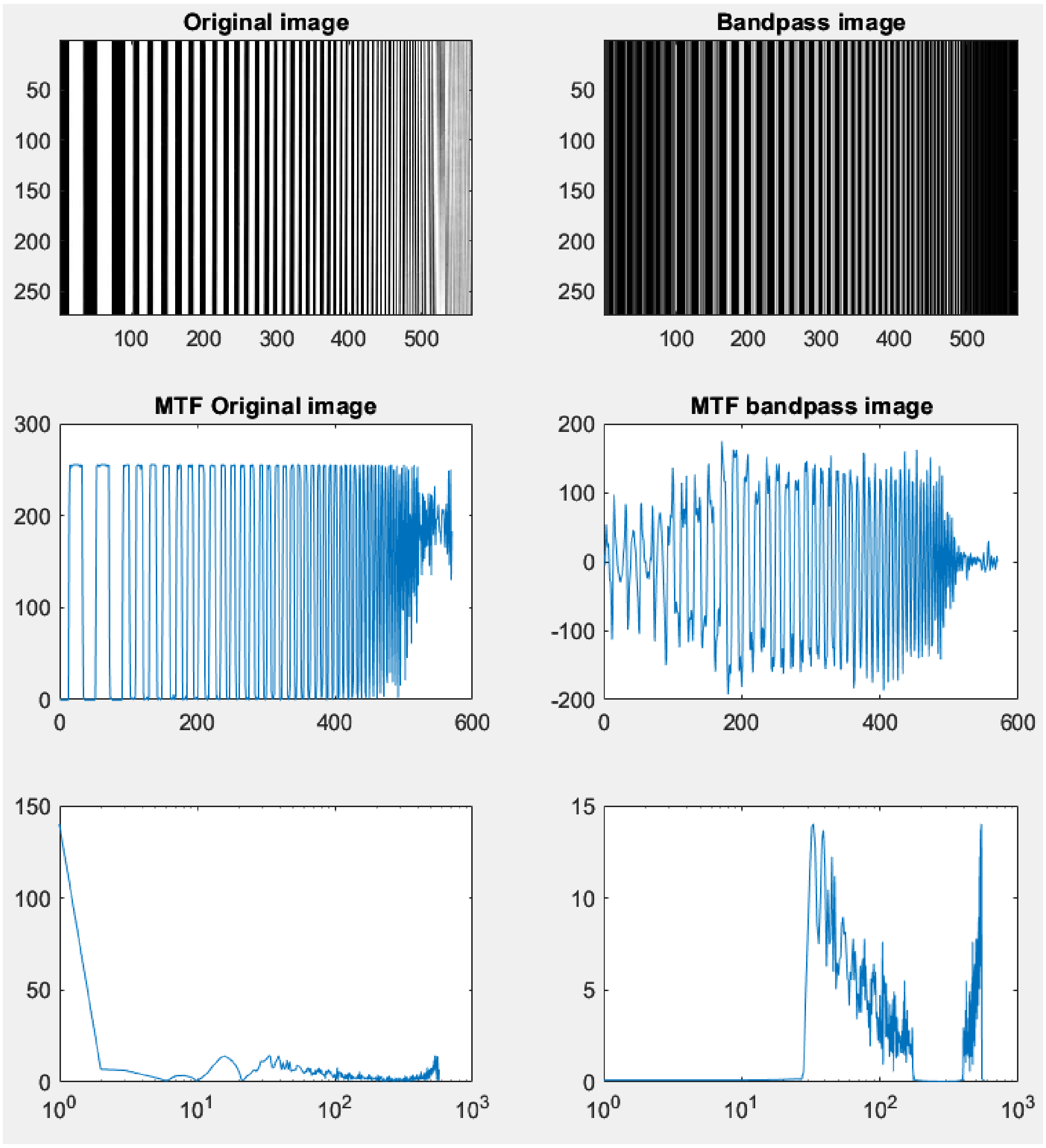
## 2) Mtf1DFT.m

```
%Read Image
%img = imread("pattern.tif");
img = imread("printedpattern.jpg");

% Display Original Image
figure();
subplot(3,4,1);
imagesc(img);
title('Original image');
colormap("gray");
[rows, cols, chans] = size(img);
central_row = uint8(rows/2);

%Extract and Plot Central Row
mtf_img = img(central_row, :);
subplot(3,4,5);
plot(mtf_img);
title('MTF Original image');

%Compute and Plot FFT of Central Row
FFT_img = fft(double(mtf_img)); %1d fft
FFT_img = abs(FFT_img)/(length (FFT_img));
subplot(3,4,9);
plot (FFT_img);
semilogx(FFT_img);

%Apply Bandpass Filter to Central Row
mtf_band_pass = bandpass(double(mtf_img), [0.1,0.6]);
%mtf_band_pass = bandpass(double(mtf_img), [0.05, 0.4]);
subplot(3,4,6);
plot(mtf_band_pass);
title('MTF bandpass image');

%Create Bandpass Filtered Image
img_band_pass = repmat(mtf_band_pass,rows,1);
subplot(3,4,2);
imagesc(uint8(img_band_pass));
title('Bandpass image');
colormap("gray");

%Compute and Plot FFT of Bandpass Filtered Image
FFT_img = fft(mtf_band_pass);
FFT_img = abs(FFT_img)/(length (FFT_img));
subplot(3,4,10);
plot (FFT_img);
semilogx(FFT_img);

%Apply Lowpass Filter to Central Row
mtf_low_pass = lowpass(double(mtf_img),0.05);
%mtf_low_pass = lowpass(double(mtf_img),0.05);
subplot(3,4,7);
plot(mtf_low_pass);
title('MTF lowpass image');

%Create Lowpass Filtered Image
img_low_pass = repmat(mtf_low_pass,rows,1);
subplot(3,4,3);
imagesc(uint8(img_low_pass));
title('Lowpass image');
colormap("gray");

%Compute and Plot FFT of Lowpass Filtered Image
FFT_img = fft(mtf_low_pass);
```

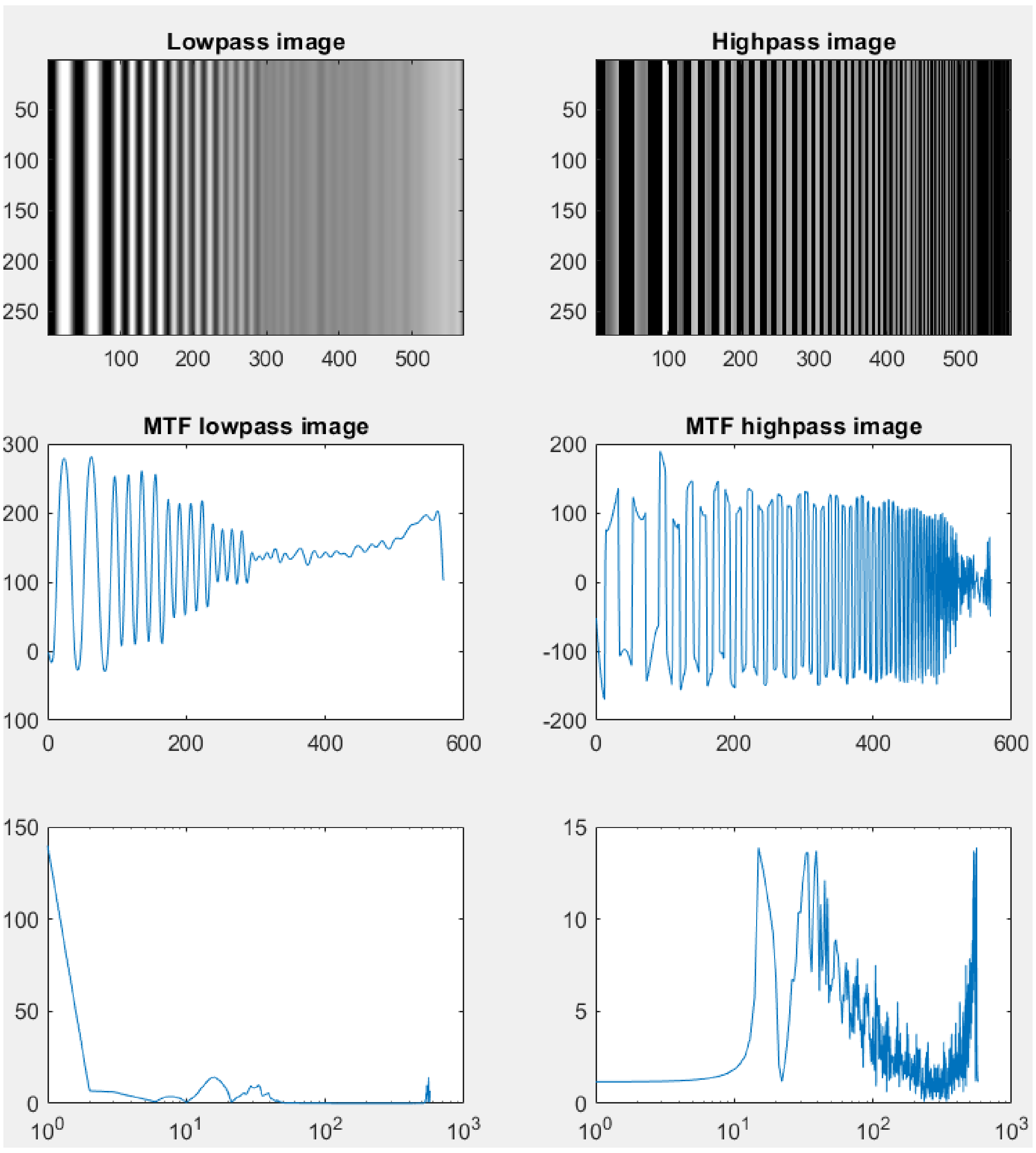
```
%Apply Highpass Filter to Central Row
mtf_high_pass = highpass(double(mtf_img),0.6);
%mtf_high_pass = highpass(double(mtf_img),0.9);
subplot(3,4,8);
plot(mtf_high_pass);
title('MTF highpass image');

%Create Highpass Filtered Image
img_high_pass = repmat(mtf_high_pass,rows,1);
subplot(3,4,4);
imagesc(uint8(img_high_pass));
title('Highpass image');
colormap("gray");

%Compute and Plot FFT of Highpass Filtered Image
FFT_img = fft(mtf_high_pass);
FFT_img = abs(FFT_img)/(length (FFT_img));
subplot(3,4,12);
plot (FFT_img);
semilogx(FFT_img);
```

## Observations:

- **Filters used in the code:**
  - **Bandpass Filter:** Allows a specific range or "band" of frequencies to pass through, blocking others. Used to isolate a particular frequency range in a signal.
  - **High pass Filter:** Permits high-frequency signals to pass, attenuating lower frequencies. Used to emphasize or isolate high-frequency components.
  - **Lowpass Filter:** Allows low-frequency signals to pass, attenuating higher frequencies. Used for smoothing or noise reduction.
- **Results:**
  - **Original Version:**

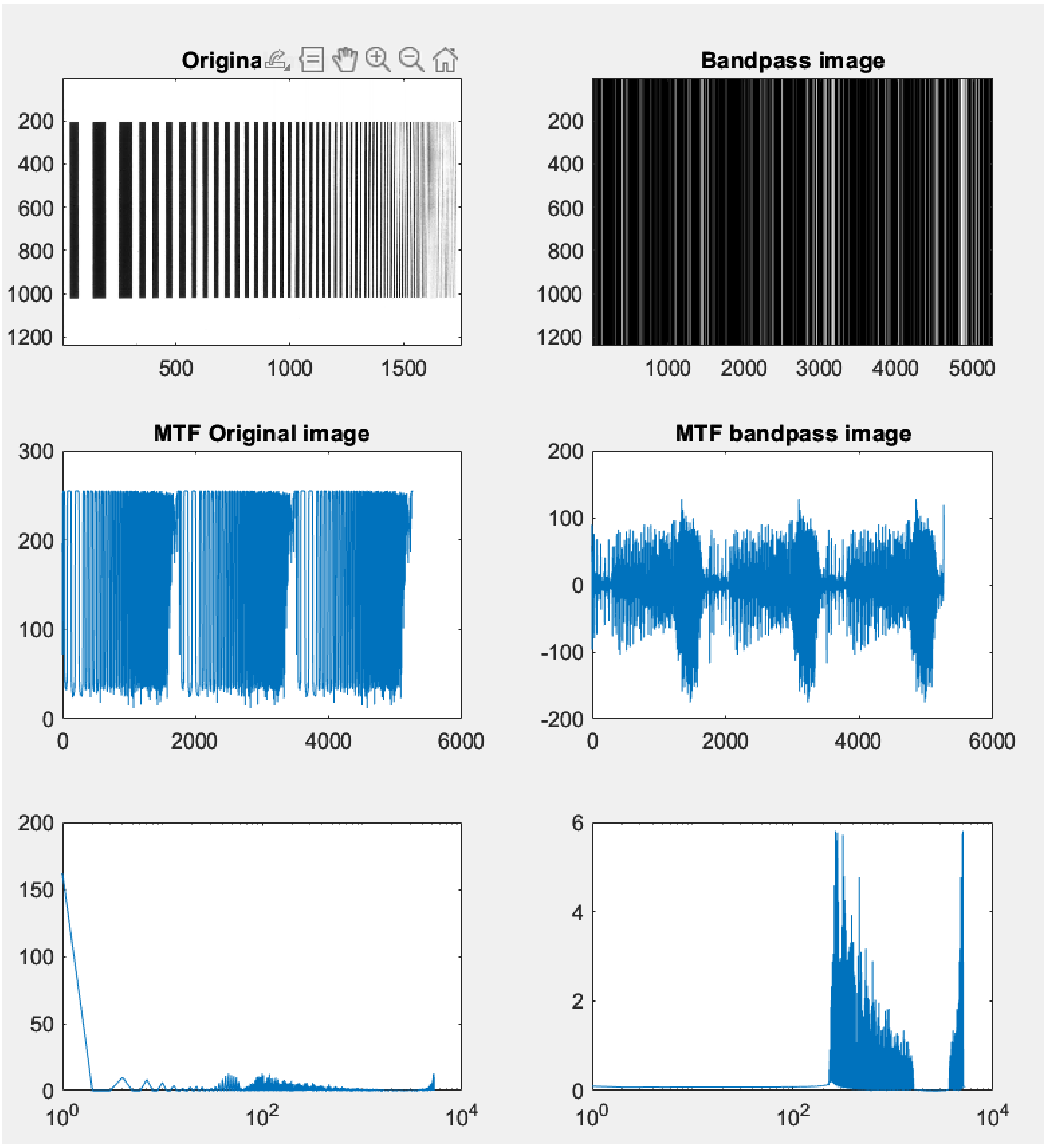


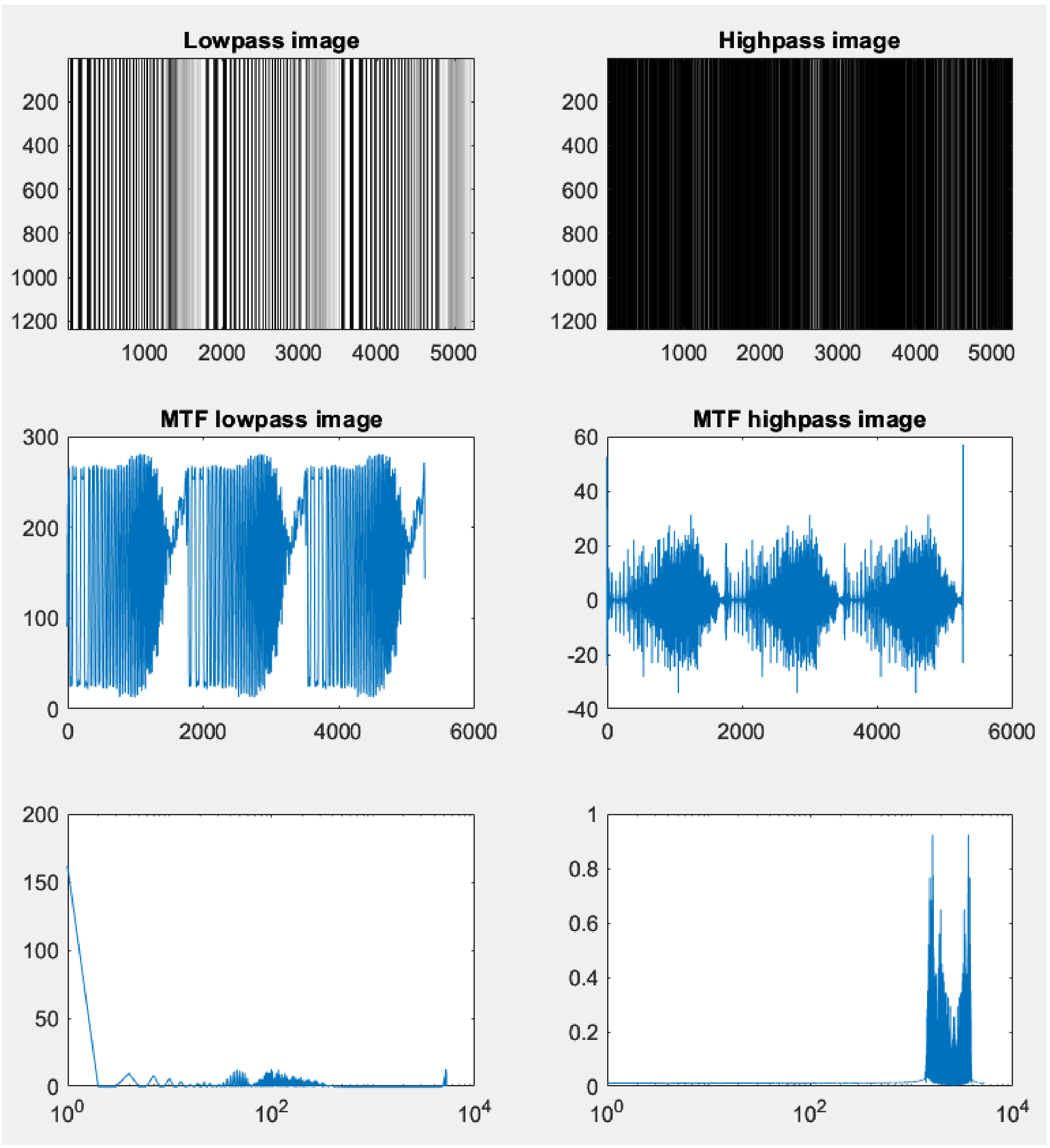
○



- Bandpass filter restricts frequencies to the range [0.1, 0.6].
- Lowpass filter allows frequencies below 0.05.
- Highpass filter allows frequencies above 0.6.

- **Scanned Version:**




Lowpass image
Highpass image
MTF lowpass image
MTF highpass image
200
400
600
800
1000
1200
1000
2000
3000
4000
5000
300
100
0
6000
60
40
20
-20
-40
150
50
1
0.8
0.6
0.4
0.2
$10^0$
$10^2$
$10^4$

## 3) FFTimage.m

```
img = imread("pattern.tif");
%img = imread("printedpattern.jpg");

figure();
subplot(3,3,1);
imagesc(img);
title('Original image');
colormap("gray");

% 2d fft
img_fft = fft2(img);
% Centre the Fourier spectrum and with abs take the amplitude
amplitude = abs(fftshift(img_fft));
subplot(3,3,2);

%plot with log to visualize better the FFT image contrast
imagesc(log(amplitude+1));
title('FFT image');
colormap("gray");

% Create a Gaussian low-pass filter
gaussian_filter = fspecial ('gaussian', size (img),20);
% Filter the Fourier spectrum
filtered_img_FFT = fftshift(img_fft).*gaussian_filter;

% Returns the filtered spectrum to its original position
filtered_img_FFT = ifftshift(filtered_img_FFT);

img_filtered = ifft2(filtered_img_FFT);

% Extract the real part of the filtered image
img_filtered = real(img_filtered);

% sobel_filter = fspecial('sobel');
% img_filtered = imfilter(img, sobel_filter, 'replicate');

subplot(3,3,3);
imagesc(img_filtered);
title('Filtered image');
colormap("gray");
```

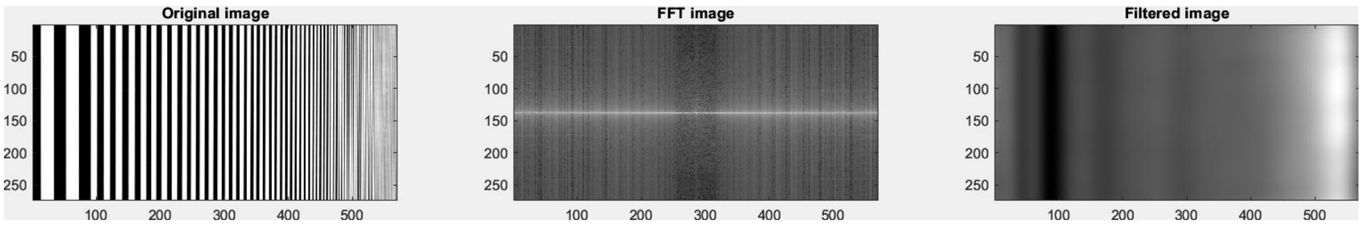
## Observations:

- **Filters used in the code:**
  - **Gaussian Filter:** It is commonly used for blurring and smoothing. A larger sigma value will result in a wider Gaussian filter, leading to more significant blurring. Here we have used three different values of Sigma i.e. 5, 10, 20.
  - **Sobel Filter:** The Sobel filter is an edge detection filter using convolution with two kernels (horizontal and vertical) to highlight image gradients and edges in computer vision.

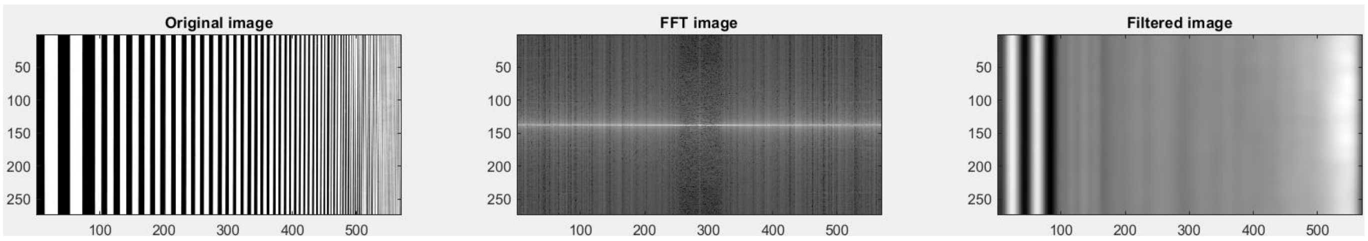
- **Results:**
  - **Original Version:**

**Gaussian Filter**

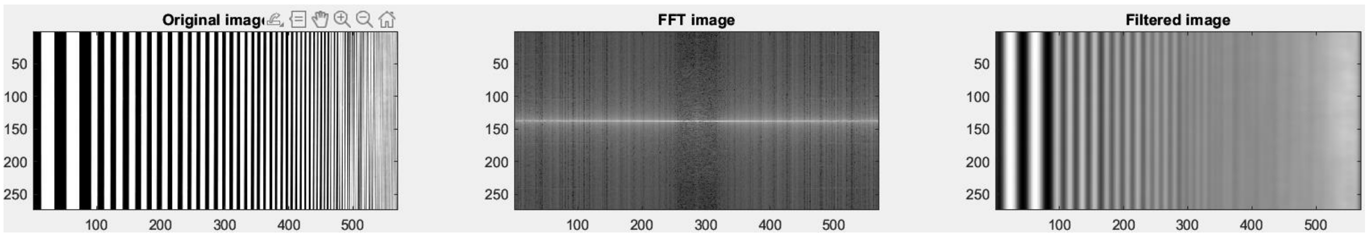
For sigma = 5



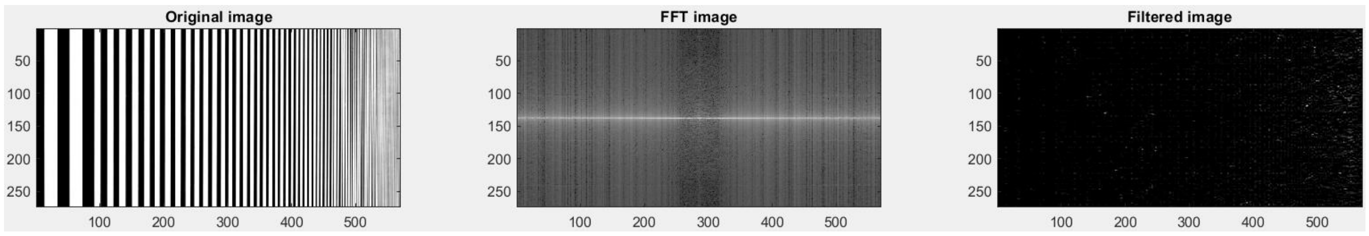
For sigma = 10



For sigma = 20



**Sobel Filter**

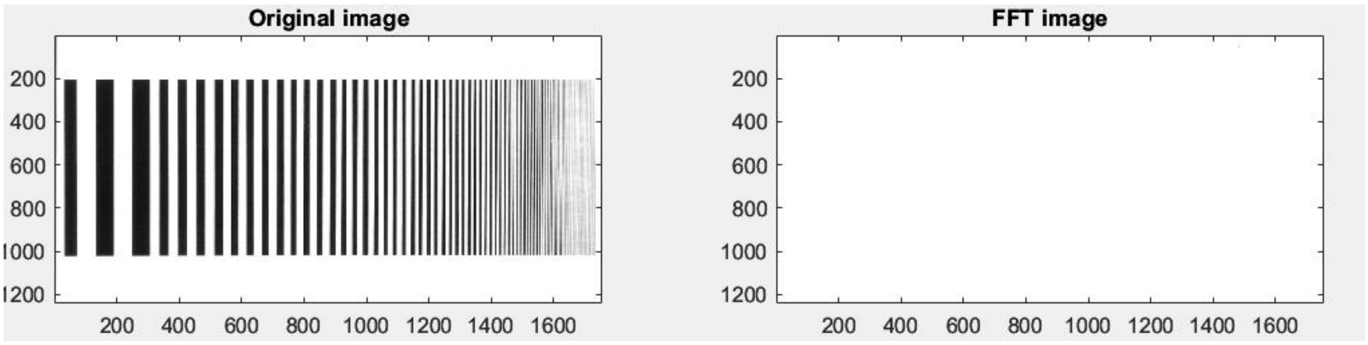


Here, we can see that the FFT image of the original image is easily distinguishable. A larger sigma value has heightened the blurring effect on the image.

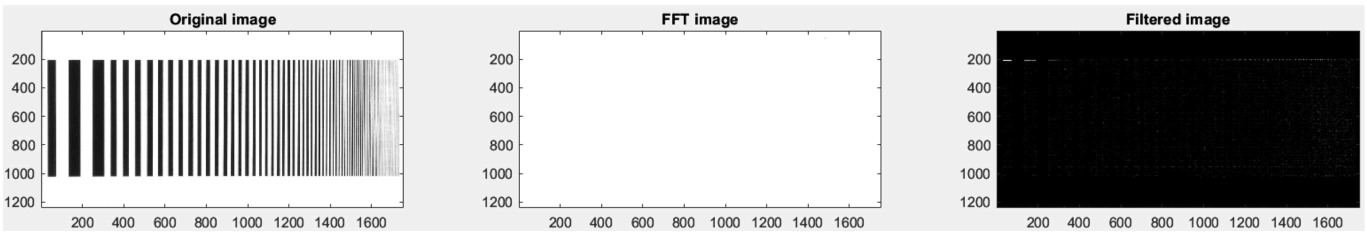
- **Scanned Version:**

**Gaussian Filter**

- For sigma = 5



**Sobel Filter**



Here in our FFT image, the white colour signifies that the intensity of various frequencies is evenly distributed. This even distribution creates a uniform appearance in the frequency domain.

# CT Experiment

## Theoretical Background:

### COMPUTED TOMOGRAPHY

A medical imaging technique that uses X-ray technology to create detailed cross-sectional images of the body. CT Scan is mainly used to differentiate between tissues of varying densities, such as bone, muscle, and organs, thanks to the differential absorption of X-rays by different materials.

How do CT Scan work?

- A rotating X-ray machine captures multiple cross-sectional images or "slices" of the body from various angles.
- These images are then processed by a computer to create detailed, three-dimensional representations of the internal structures. This part of the process is known as reconstruction.

### SINOGRAM

A sinogram is a graphical representation of the raw data collected during the scanning process. It is a visual representation of the measured X-ray attenuation values as a function of the rotation angle of the X-ray source around the patient.

### SIMPLE BACK PROJECTION

It involves tracing rays through pixels in acquired X-ray projections, accumulating values along these rays, and combining results from different angles to create a reconstructed image. Normally produces blurry images, for that reason we used several other filters to obtain sharper image.

### 2D FOURIER TRANSFORM

The 2D Fourier Transform is a mathematical operation that converts a 2D image from its spatial domain (pixel values) to its frequency domain (amplitude and phase of spatial frequencies). Filtering in the frequency domain involves modifying the amplitude or phase of specific frequency components in order to reduce noise, obtain sharper or smoother images. Thus making it more making it clearer and more interpretable.

### CT PHANTOM

Phantom mimic the attenuation properties of human tissues. They contain structures with known density and contrast, allowing for calibration and quality assurance of CT scanners.

## Matlab Code:

We've exclusively utilized "tomo_rec.m" for our work, making modifications solely to this document. The remaining documents remain unaltered.

## tomo_rec.m

```matlab
% Import an image to reconstruct
% You can choose: 1) slice of a tooth in tif format; 2) define a phantom to
% test image reconstruction; 3) import a thorax's slice in dicom format
% you can resize the image

%img = phantom(526);
%img = dicomread('CT.dcm');
img =imread("tooth_cl.tif");

%Add noise if you want
img_double = imnoise(img,'salt & pepper', 1);

% set min of image to zero and transform to double
img_double = double(img - min(img(:)));


%transformimg to 16-bit
img_16 = mat2gray(img_double);
img_16 = uint16(img_16*65535);


figure();
subplot(3,2,1);
imagesc(img_double);
title('Original image');
colormap("gray");

% set some parameters
%freq = 1; % [1/degree] how many projections in 1 degree
freq = 0.5;
%freq = 1.5;
theta_max = 180; % half reconstruction, you can change to full reconstruction
thetas = 0:1/freq:theta_max-1/freq; % compute the number of projections

sinogram = myRadon(img_double, thetas); %generate the sinogram of the ideal image
subplot(3,2,2)
imagesc(sinogram)
title('Sinogram');
colormap("gray");

% simple backprojection (schlegel & bille 9.1.2)
subplot(3,2,3);
simpleBackprojection = myBackprojection(sinogram,thetas);
simpleBackprojection16 = mat2gray(simpleBackprojection);
simpleBackprojection16 = uint16(simpleBackprojection16*65535);
imagesc(simpleBackprojection16);
title('Image reconstructed with simple backprojection');
colormap("gray")
```

```
% reconstruction with filtered back projection in spatial domain (schlegel
% & bille 9.3.1).
subplot(3,2,4);

reconstruction1DFTSpatialDomain_ramp = myFilteredBackprojectionSpatialDomain(sinogram, thetas, 'ramLak'); % apply a ramp filer
reconstruction1DFTSpatialDomain_ramp = mat2gray(reconstruction1DFTSpatialDomain_ramp);
reconstruction1DFTSpatialDomain_ramp = uint16(reconstruction1DFTSpatialDomain_ramp*65535);
imagesc(reconstruction1DFTSpatialDomain_ramp);
title('Image reconstructed Ramp filtered back projection in spatial domain');
colormap("gray");
subplot(3,2,5);
reconstruction1DFTSpatialDomain_shepp = myFilteredBackprojectionSpatialDomain(sinogram, thetas, 'sheppLogan');% apply a sheppLogan filer
reconstruction1DFTSpatialDomain_shepp = mat2gray(reconstruction1DFTSpatialDomain_shepp);
reconstruction1DFTSpatialDomain_shepp = uint16(reconstruction1DFTSpatialDomain_shepp*65535);
imagesc(reconstruction1DFTSpatialDomain_shepp);
title('Image reconstructed with Shepp-Logan filtered back projection in spatial domain');
colormap("gray");

% reconstruction with 2D fourier transformation (schlegel & bille 9.2.1)
% note that this function uses the backprojection of the sinogram as input
% and _not_ the sinogram
subplot(3,2,6);
reconstrution2DFT = myFilteredBackprojection2DFT(simpleBackprojection);
reconstrution2DFT = mat2gray(reconstrution2DFT);
reconstrution2DFT = uint16(reconstrution2DFT*65535);
imagesc(reconstrution2DFT);
title('Image reconstructed with filtered 2D fourier transform');
colormap("gray");

%Calculate SNR and MSE
%For simple back-projection
[peaksnr_bp, snr_bp] = psnr(img_16, simpleBackprojection16);
mse_bp = immse(img_16, simpleBackprojection16);
fprintf('MSE_bp = %f\n',mse_bp);
fprintf('snr_bp = %f\n',snr_bp);

%For filtered back-projection with ramp filter
[peaksnr_fbp_ramp, snr_fbp_ramp] = psnr(img_16, reconstruction1DFTSpatialDomain_ramp);
mse_fbp_ramp = immse(img_16, reconstruction1DFTSpatialDomain_ramp);
fprintf('MSE_fbp_ramlak = %f\n',mse_fbp_ramp);
fprintf('snr_fbp_ramlak = %f\n',snr_fbp_ramp);

%For filtered back-projection with shepp-logan filter
[peaksnr_fbp_shepp, snr_fbp_shepp] = psnr(img_16, reconstruction1DFTSpatialDomain_shepp);
mse_fbp_shepp = immse(img_16, reconstruction1DFTSpatialDomain_shepp);
fprintf('MSE_fbp_shepp = %f\n',mse_fbp_shepp);
fprintf('snr_fbp_shepp = %f\n',snr_fbp_shepp);

%for 2D fourier transformation

[peaksnr_2dft, snr_2dft] = psnr(img_16, reconstrution2DFT);
mse_2dft = immse(img_16, reconstrution2DFT);
fprintf('MSE_fbp_2dfbp = %f\n',mse_2dft);
fprintf('snr_fbp_2dfbp = %f\n',snr_2dft);
```

## Observations:

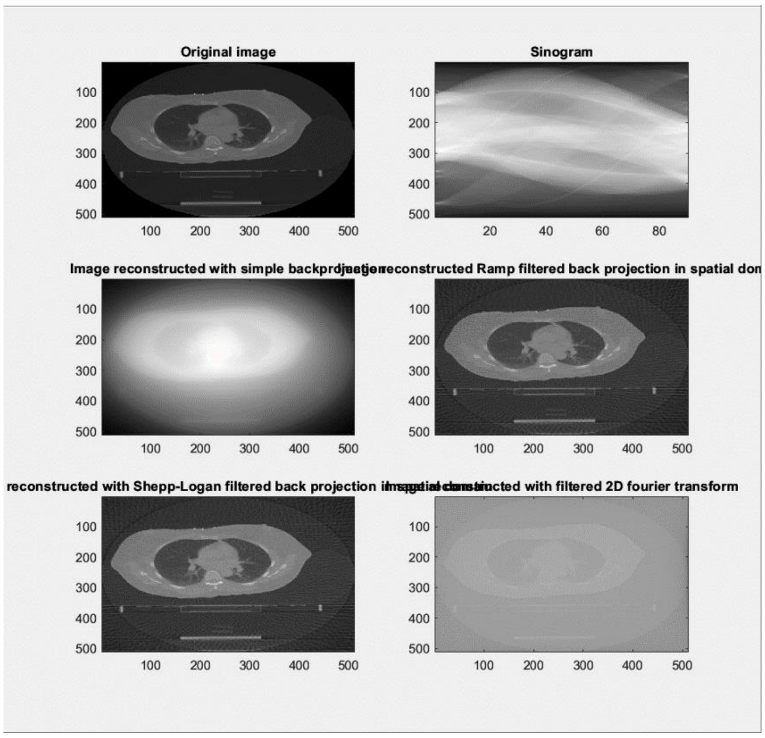
Using this code, we examined three distinct images: CT images of a Phantom, Thorax, and Tooth. We varied the number of projections per 1 degree (frequency) and computed their Signal-to-Noise Ratio (SNR) and Mean Squared Error (MSE) across different filters.

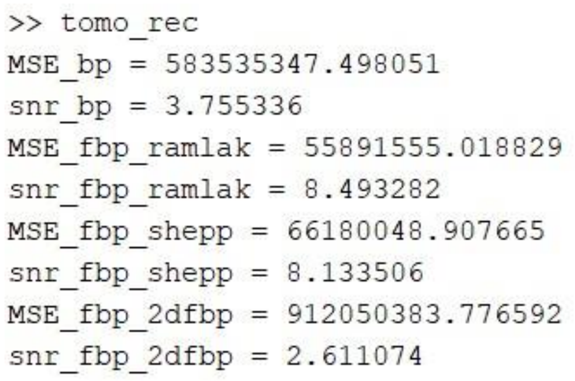
- ➢ SNR (Signal-to-Noise Ratio) is a measure that compares the strength of a desired signal to the level of background noise. A higher SNR indicates better signal quality, while a lower SNR suggests a weaker signal relative to the noise. It is often expressed in decibels (dB) for convenience.
- ➢ The Mean Squared Error is a metric that measures the average squared difference between corresponding pixel intensities of the original and reconstructed images. A lower MSE indicates better similarity between the two images.

- **Filters used in the code:**
  - Ramlak Filter/ Ramp Filter: It is applied in the frequency domain to enhance high-frequency components, improving the sharpness of reconstructed images by compensating for inherent blurring during the image acquisition process.
  - Shepp-Logan Filter: It aims to reduce blurring and enhance image details by employing a smooth frequency response. The filter is designed to improve the quality of reconstructed images, especially when balancing artifact reduction and detail preservation is crucial.
- **Results:**

**THOORAX DIACOM CT RECONSTRUCTION WITH SNR VAUES**

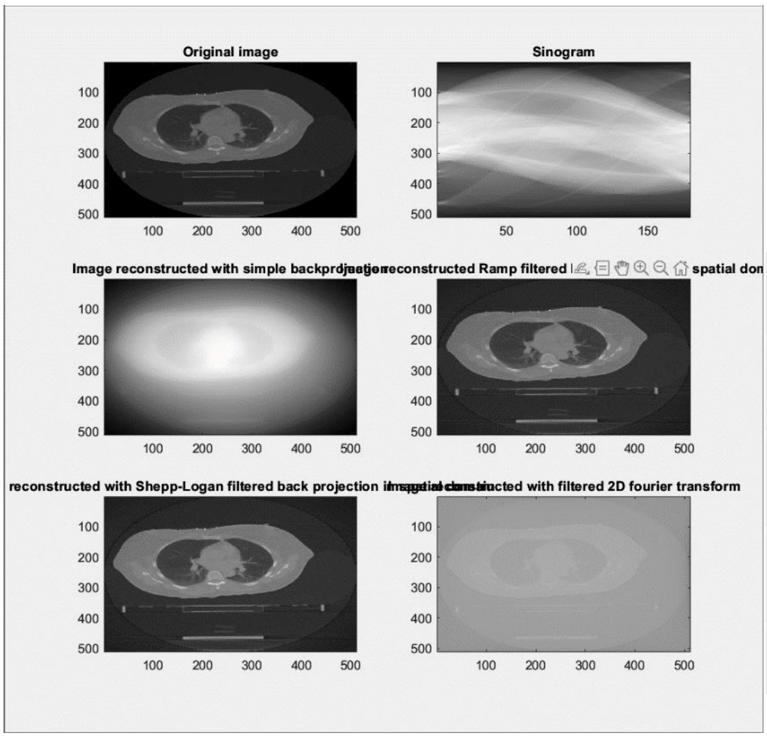
**FREQ= 0.5**

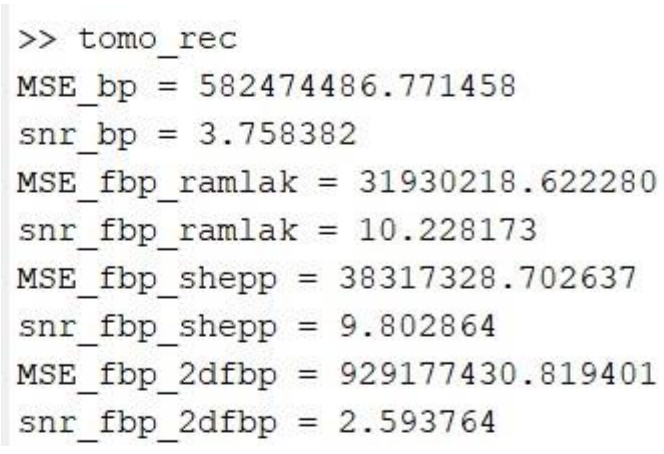

Original image
Sinogram
Image reconstructed with simple backprojection
Image reconstructed Ramp filtered back projection in spatial domain
reconstructed with Shepp-Logan filtered back projection in spatial domain
Image reconstructed with filtered 2D fourier transform



```
>> tomo_rec
MSE_bp = 583535347.498051
snr_bp = 3.755336
MSE_fbp_ramlak = 55891555.018829
snr_fbp_ramlak = 8.493282
MSE_fbp_shepp = 66180048.907665
snr_fbp_shepp = 8.133506
MSE_fbp_2dfbp = 912050383.776592
snr_fbp_2dfbp = 2.611074
```

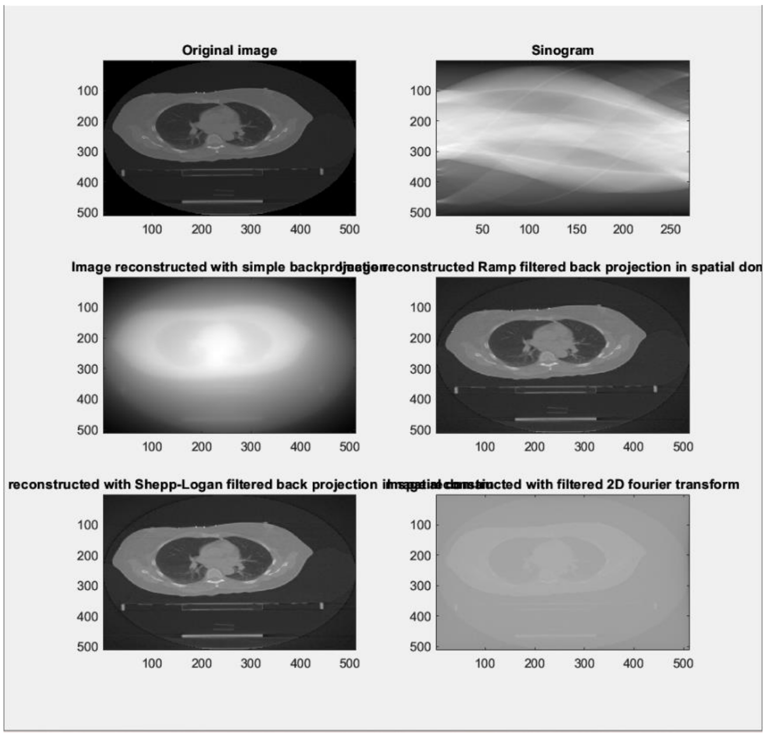
**FREQ= 1.0**

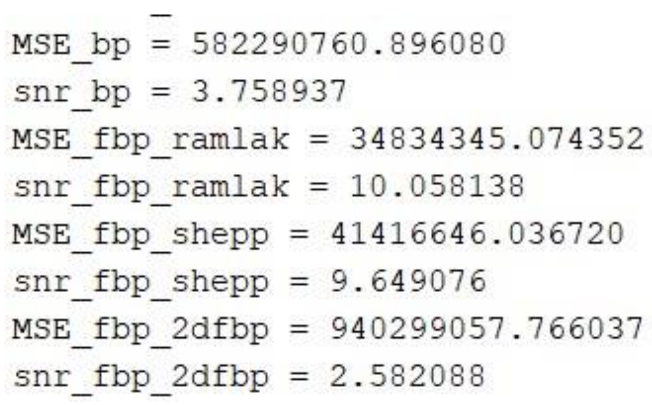

Original image
Sinogram
Image reconstructed with simple backprojection
Image reconstructed Ramp filtered back projection in spatial domain
reconstructed with Shepp-Logan filtered back projection in spatial domain
Image reconstructed with filtered 2D fourier transform



```
>> tomo_rec
MSE_bp = 582474486.771458
snr_bp = 3.758382
MSE_fbp_ramlak = 31930218.622280
snr_fbp_ramlak = 10.228173
MSE_fbp_shepp = 38317328.702637
snr_fbp_shepp = 9.802864
MSE_fbp_2dfbp = 929177430.819401
snr_fbp_2dfbp = 2.593764
```

**FREQ= 1.5**

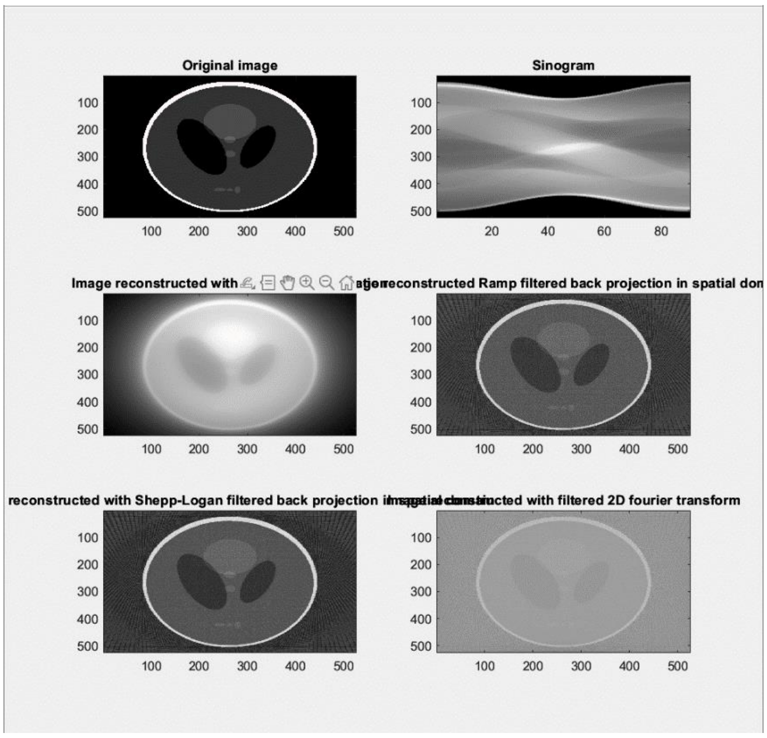

Original image
Sinogram
Image reconstructed with simple backprojection
Image reconstructed Ramp filtered back projection in spatial domain
reconstructed with Shepp-Logan filtered back projection in spatial domain
Image reconstructed with filtered 2D fourier transform

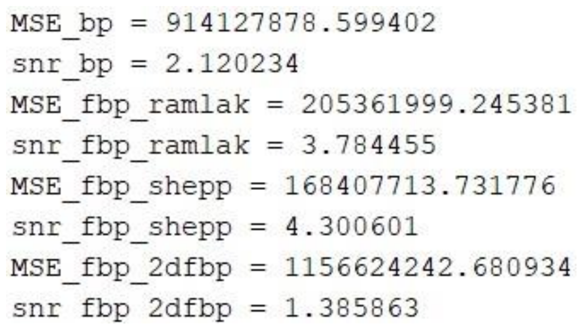


```
MSE_bp = 582290760.896080
snr_bp = 3.758937
MSE_fbp_ramlak = 34834345.074352
snr_fbp_ramlak = 10.058138
MSE_fbp_shepp = 41416646.036720
snr_fbp_shepp = 9.649076
MSE_fbp_2dfbp = 940299057.766037
snr_fbp_2dfbp = 2.582088
```

## PHANTOM CT RECONSTRUCTION

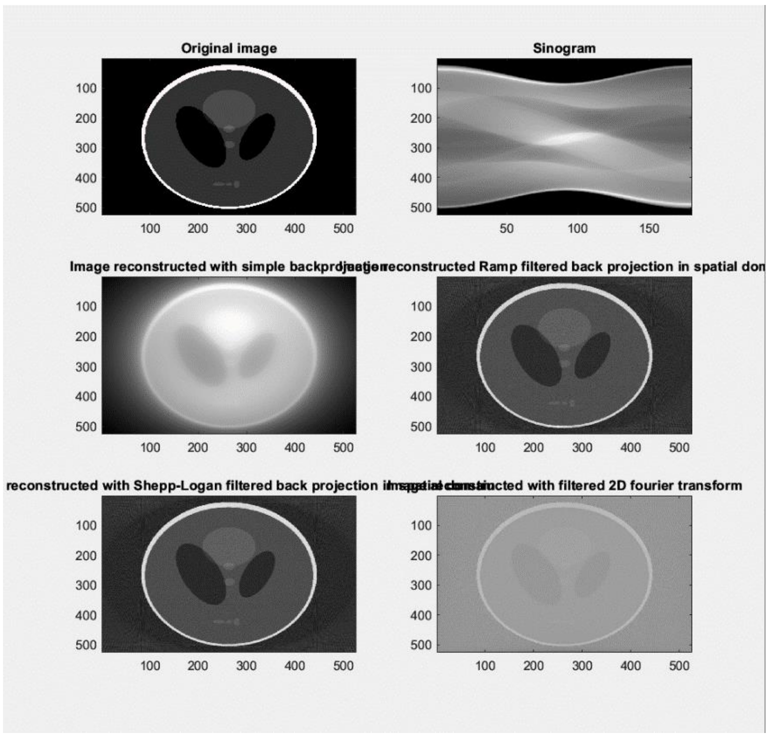
### FREQ= 0.5

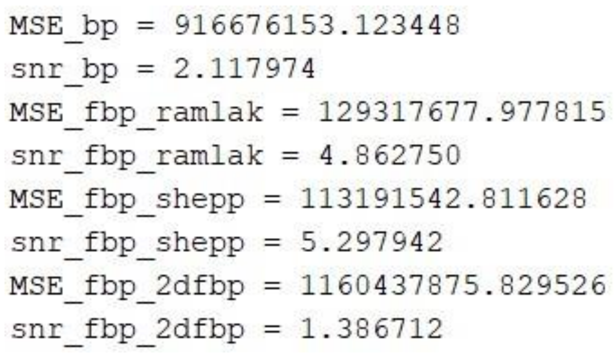

Original image
Sinogram
Image reconstructed with
reconstructed Ramp filtered back projection in spatial dom
reconstructed with Shepp-Logan filtered back projection in spatial domain
constructed with filtered 2D fourier transform



```
MSE_bp = 914127878.599402
snr_bp = 2.120234
MSE_fbp_ramlak = 205361999.245381
snr_fbp_ramlak = 3.784455
MSE_fbp_shepp = 168407713.731776
snr_fbp_shepp = 4.300601
MSE_fbp_2dfbp = 1156624242.680934
snr fbp 2dfbp = 1.385863
```

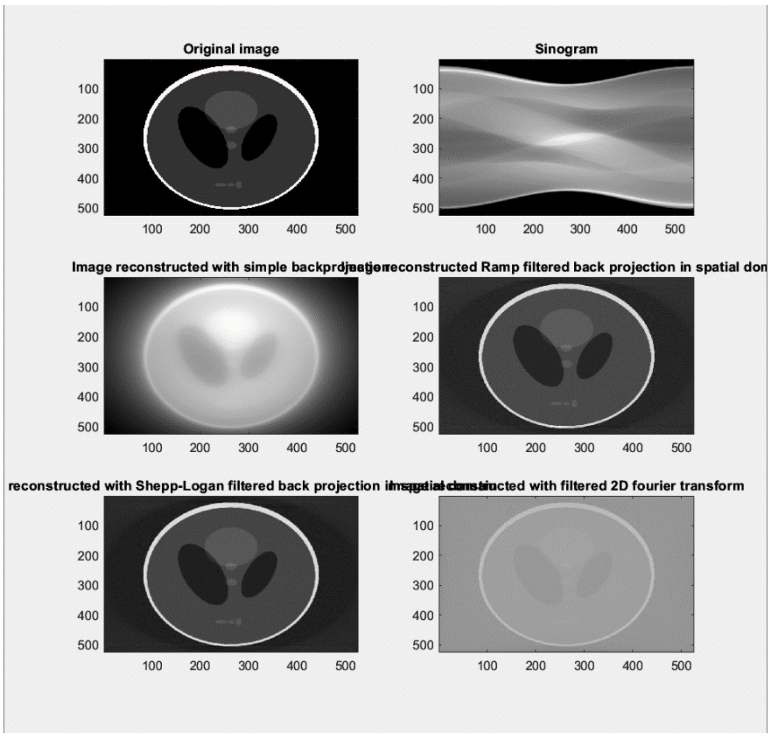
### FREQ= 1.0

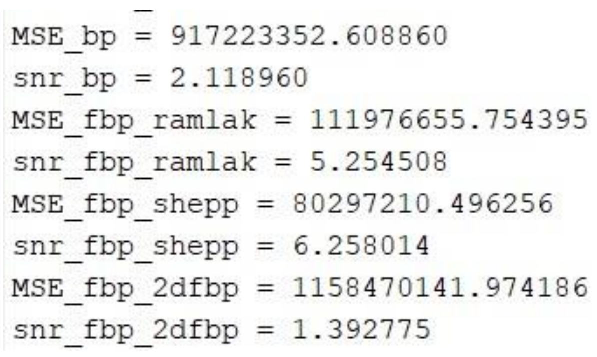

Original image
Sinogram
Image reconstructed with simple backprojection
reconstructed Ramp filtered back projection in spatial dom
reconstructed with Shepp-Logan filtered back projection in spatial domain
constructed with filtered 2D fourier transform



```
MSE_bp = 916676153.123448
snr_bp = 2.117974
MSE_fbp_ramlak = 129317677.977815
snr_fbp_ramlak = 4.862750
MSE_fbp_shepp = 113191542.811628
snr_fbp_shepp = 5.297942
MSE_fbp_2dfbp = 1160437875.829526
snr_fbp_2dfbp = 1.386712
```

**FREQ= 1.5**

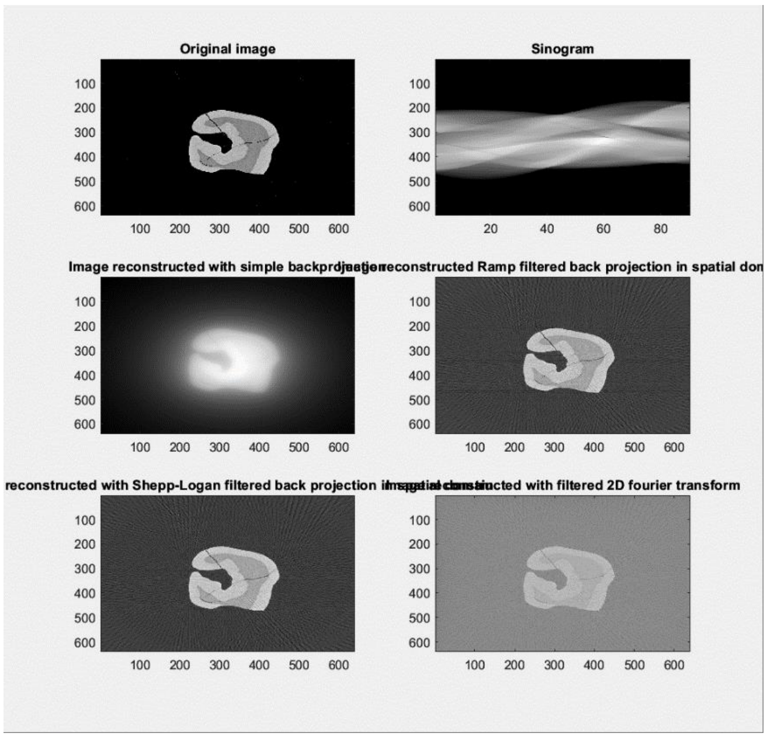

Original image
Sinogram
Image reconstructed with simple backprojection
Image reconstructed Ramp filtered back projection in spatial dom
reconstructed with Shepp-Logan filtered back projection in spatial dom
Image reconstructed with filtered 2D fourier transform

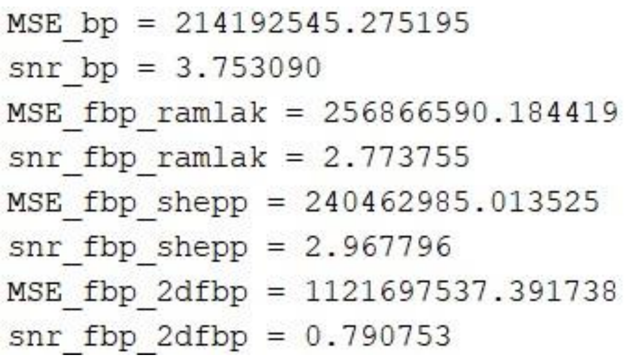


```
MSE_bp = 917223352.608860
snr_bp = 2.118960
MSE_fbp_ramlak = 111976655.754395
snr_fbp_ramlak = 5.254508
MSE_fbp_shepp = 80297210.496256
snr_fbp_shepp = 6.258014
MSE_fbp_2dfbp = 1158470141.974186
snr_fbp_2dfbp = 1.392775
```

**TOOTH CLIFF CT RECONSTRUCTION**

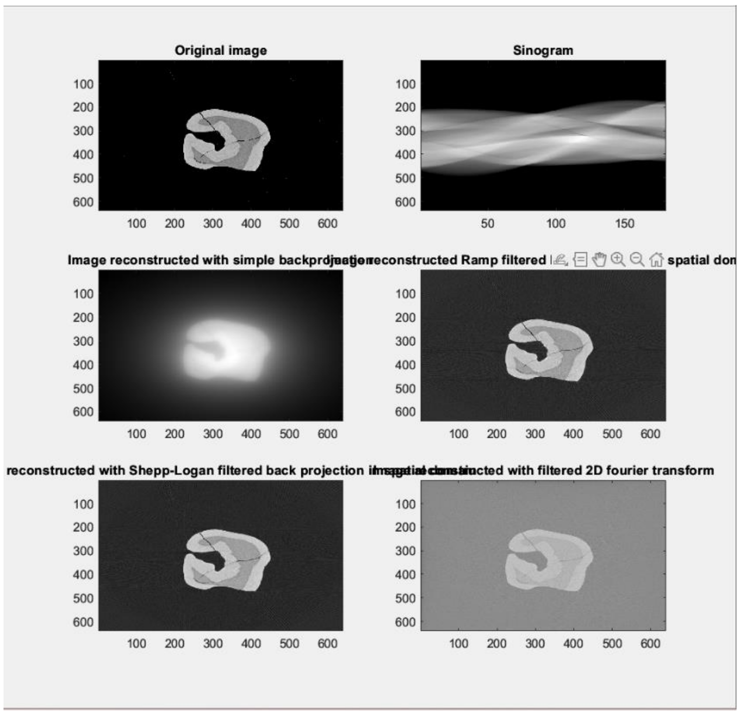
**FREQ= 0.5**


Original image
Sinogram
Image reconstructed with simple backprojection
Image reconstructed Ramp filtered back projection in spatial dom
reconstructed with Shepp-Logan filtered back projection in spatial dom
Image reconstructed with filtered 2D fourier transform



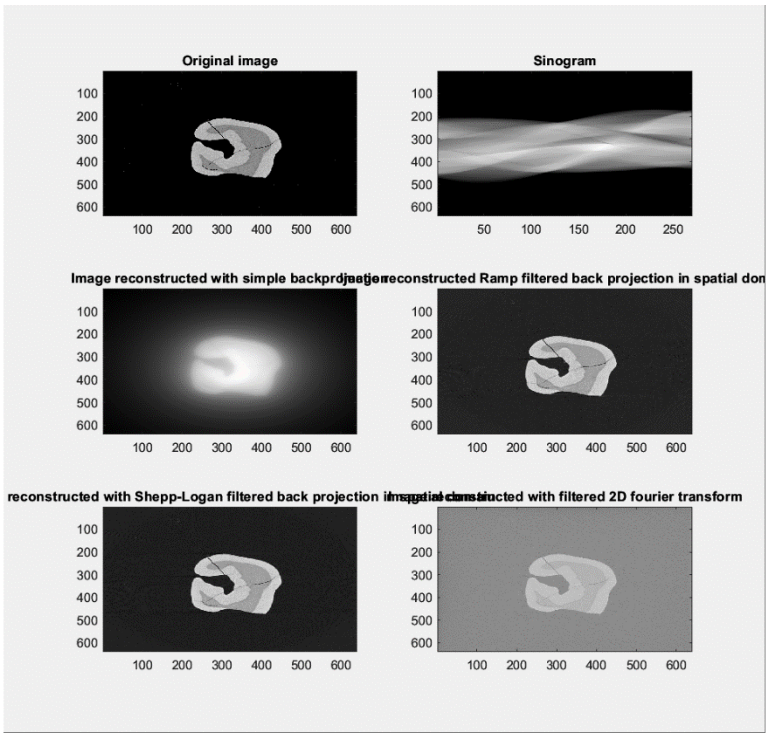
```
MSE_bp = 214192545.275195
snr_bp = 3.753090
MSE_fbp_ramlak = 256866590.184419
snr_fbp_ramlak = 2.773755
MSE_fbp_shepp = 240462985.013525
snr_fbp_shepp = 2.967796
MSE_fbp_2dfbp = 1121697537.391738
snr_fbp_2dfbp = 0.790753
```

**FREQ= 1.0**


Original image
Sinogram
Image reconstructed with simple backprojection
Image reconstructed Ramp filtered
spatial dom
reconstructed with Shepp-Logan filtered back projection in spatial domain
Image reconstructed with filtered 2D fourier transform
100
200
300
400
500
600
50
150

```
MSE_bp = 214427224.364163
snr_bp = 3.754938
MSE_fbp_ramlak = 126790678.439365
snr_fbp_ramlak = 4.460915
MSE_fbp_shepp = 111840978.775540
snr_fbp_shepp = 4.891697
MSE_fbp_2dfbp = 1156719319.304600
snr_fbp_2dfbp = 0.812824
```

**FREQ= 1.5**


Original image
Sinogram
Image reconstructed with simple backprojection
Image reconstructed Ramp filtered back projection in spatial dom
reconstructed with Shepp-Logan filtered back projection in spatial domain
Image reconstructed with filtered 2D fourier transform
100
200
300
400
500
600
50
150
250

```
MSE_bp = 215231467.995654
snr_bp = 3.749987
MSE_fbp_ramlak = 85581720.161831
snr_fbp_ramlak = 5.771690
MSE_fbp_shepp = 67200813.774492
snr_fbp_shepp = 6.601743
MSE_fbp_2dfbp = 1153390219.391509
snr_fbp_2dfbp = 0.824847
```

Looking at pictures from different filters, we notice that images are clearer at higher frequencies and less clear at lower ones. Higher frequencies also give us better SNR values, while lower frequencies have lower SNR values. When we increase the frequency, the MSE value also goes up. So, we need to pick a frequency that gives us sharp images (high SNR) but doesn't raise the MSE too much. Keep in mind, though, that having more projections takes more time and increases the dose for the patient, which is not good.

## References:

**[1]** DD Brüllmann, B d'Hoedt, The modulation transfer function and signal-to-noise ratio of different digital filters: a technical approach, Dentomaxillofacial Radiology, Volume 40, Issue 4, 1 May 2011, Pages 222–229, https://doi.org/10.1259/dmfr/33029984

**[2]** https://en.wikipedia.org/wiki/CT_scan#Mechanism

**[3]** https://www.clear.rice.edu/elec431/projects96/DSP/filters.html