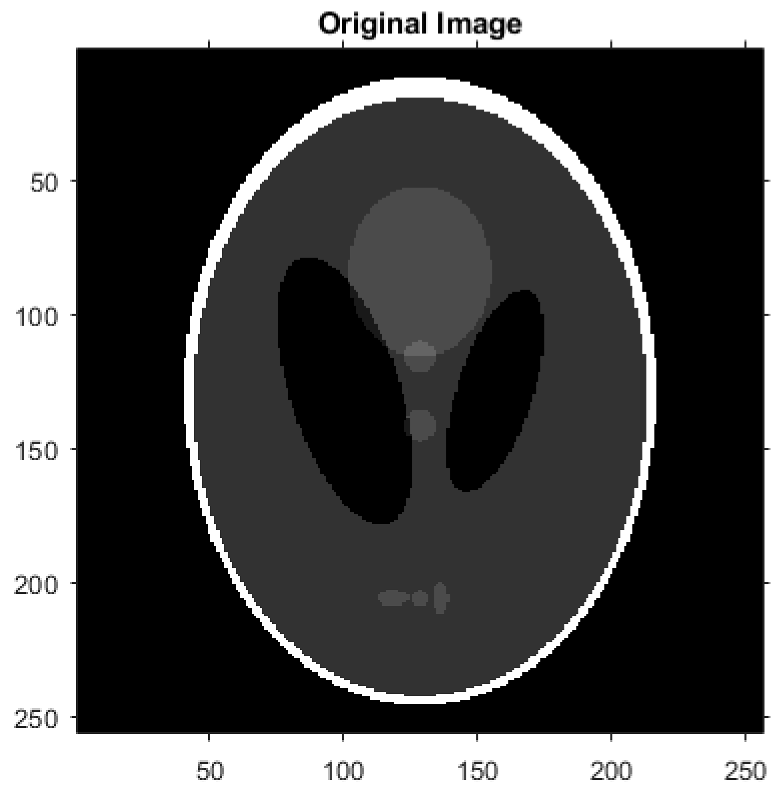
## Table of Contents

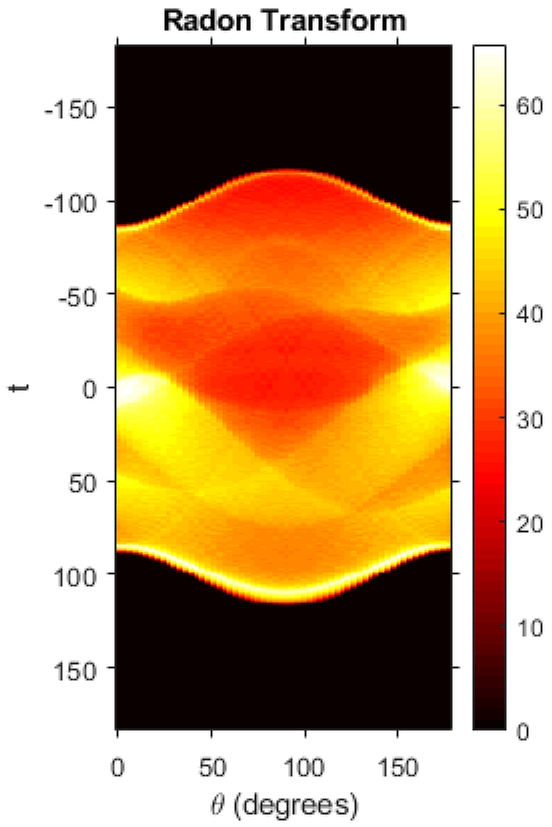
# Original Image

```
clc; close all; clear;
iptsetpref('ImshowAxesVisible','on');
iptsetpref('ImshowInitialMagnification','fit');
img = imread("../../data/SheppLogan256.png");
img = double(img)/255;
n = size(img, 1);
theta = 0:3:177;
figure;
imshow(img);
title('Original Image');
```



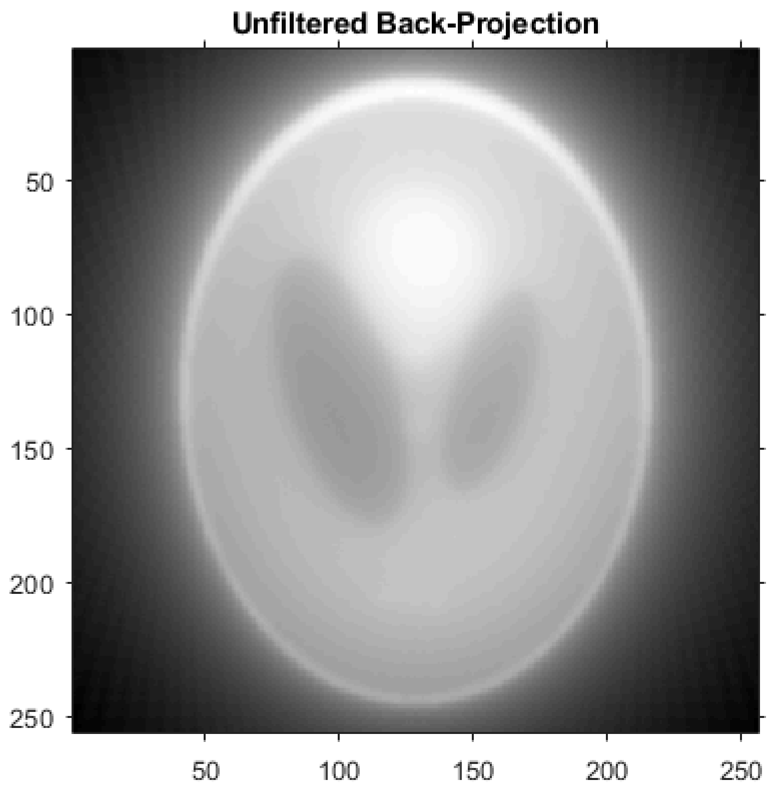
## Radon Transform

```
[rad, xp] = radon(img, theta);
figure;
imshow(rad,[],'Xdata',theta,'Ydata',xp);
xlabel('\theta (degrees)');
ylabel('t');
title('Radon Transform');
colormap(gca,hot), colorbar;
```



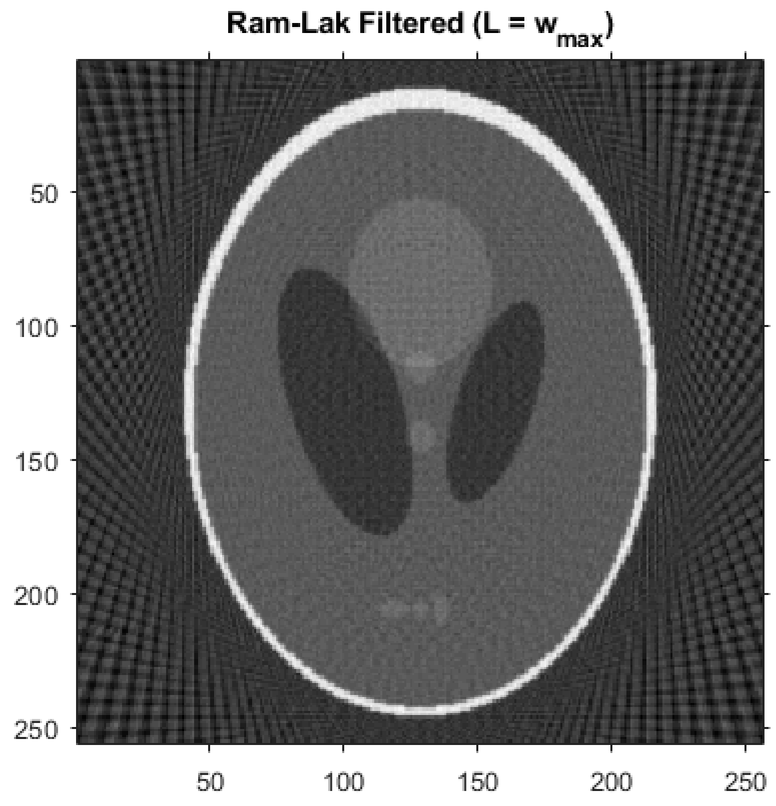
## Unfiltered Back-Projection

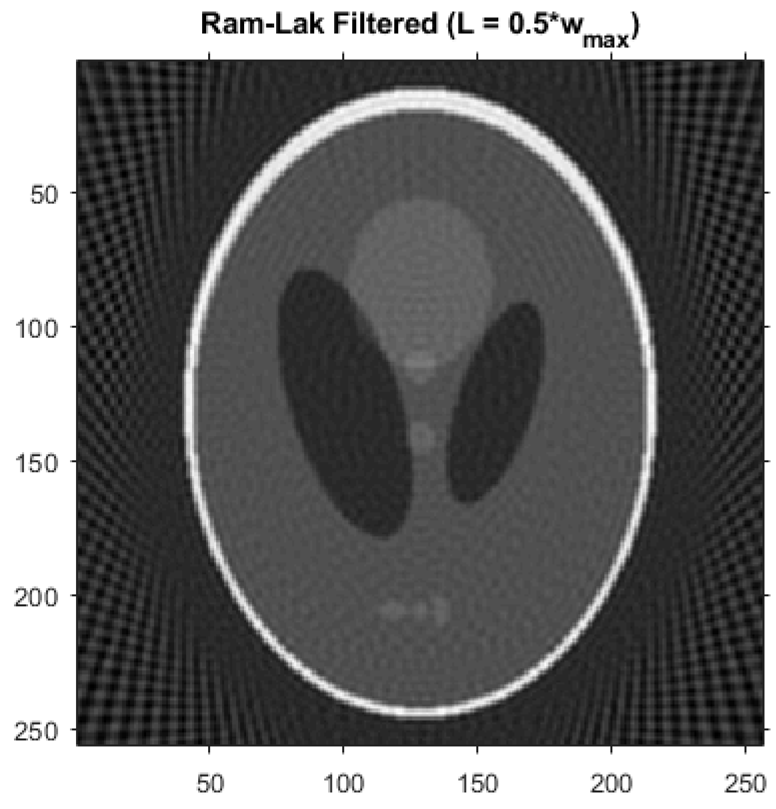
```
unfiltered_img = rescale(iradon(rad,theta,'linear','none',1,n));
figure;
imshow(unfiltered_img);
title('Unfiltered Back-Projection');
```



## Filtered Back-Projection (Ram-Lak Filter)

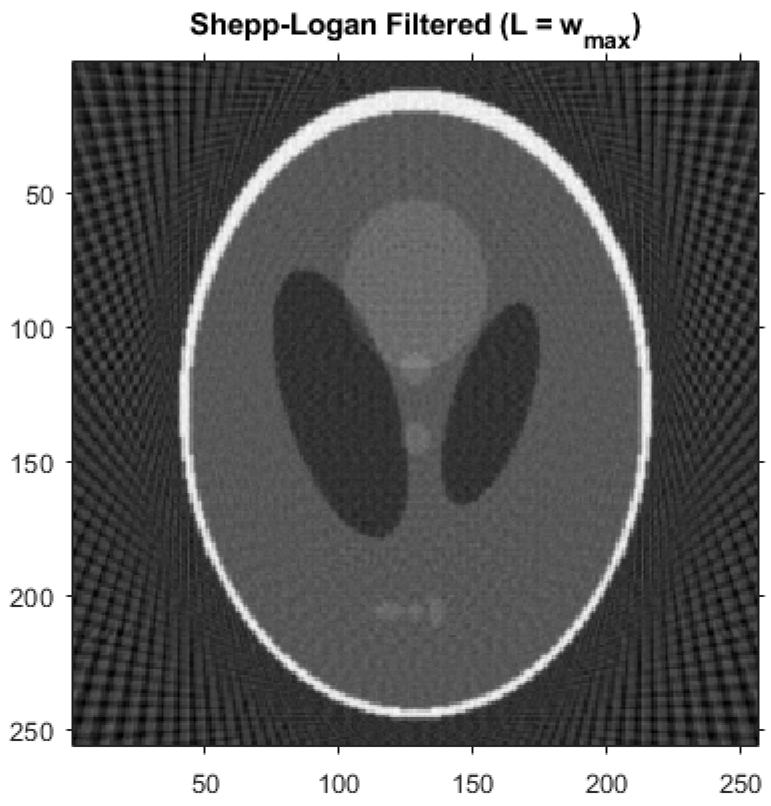
```
filt_rad_full = myFilter(rad,1,'Ram-Lak');
filt_rad_half = myFilter(rad,0.5,'Ram-Lak');

img_full = rescale(iradon(filt_rad_full,theta,'linear','none',1,n));
img_half = rescale(iradon(filt_rad_half,theta,'linear','none',1,n));

figure;
imshow(img_full);
title('Ram-Lak Filtered (L = w_{max})')
figure;
imshow(img_half);
title('Ram-Lak Filtered (L = 0.5*w_{max})')
```

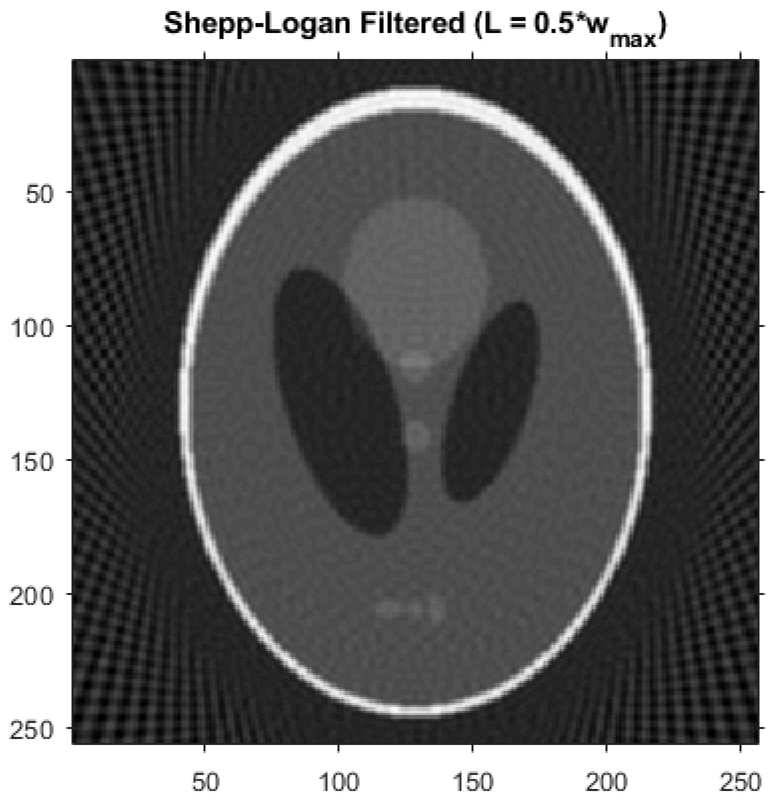

Ram-Lak Filtered (L = $w_{max}$)
50
100
150
200
250
50
100
150
200
250


Ram-Lak Filtered (L = 0.5*$w_{max}$)
50
100
150
200
250
50
100
150
200
250

## Filtered Back-Projection (Shepp-Logan Filter)

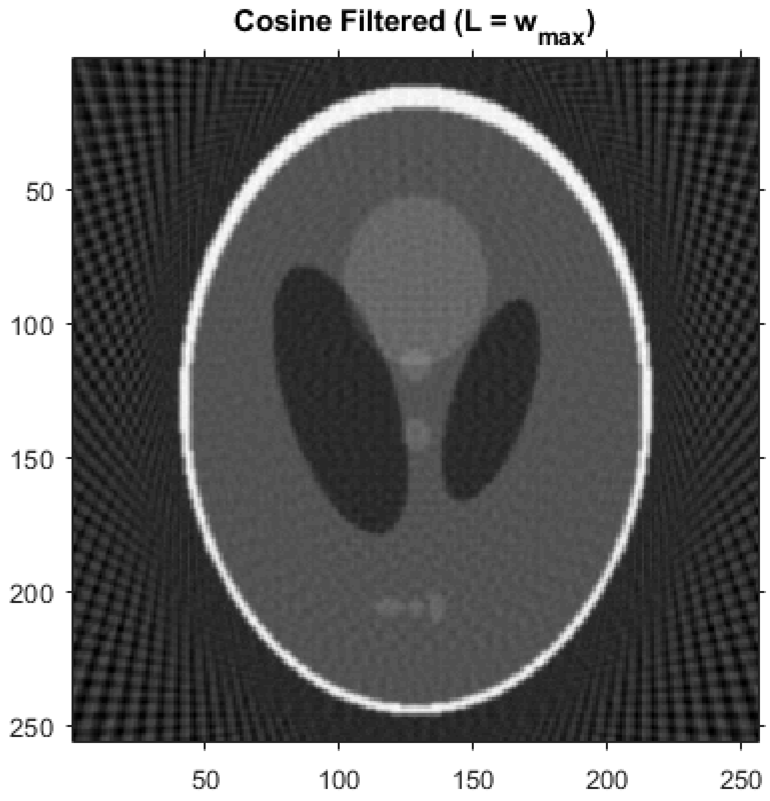
```
filt_rad_full = myFilter(rad,1,'Shepp-Logan');
filt_rad_half = myFilter(rad,0.5,'Shepp-Logan');

img_full = rescale(iradon(filt_rad_full,theta,'linear','none',1,n));
img_half = rescale(iradon(filt_rad_half,theta,'linear','none',1,n));

figure;
imshow(img_full);
title('Shepp-Logan Filtered (L = w_{max})')
figure;
imshow(img_half);
title('Shepp-Logan Filtered (L = 0.5*w_{max})')
```

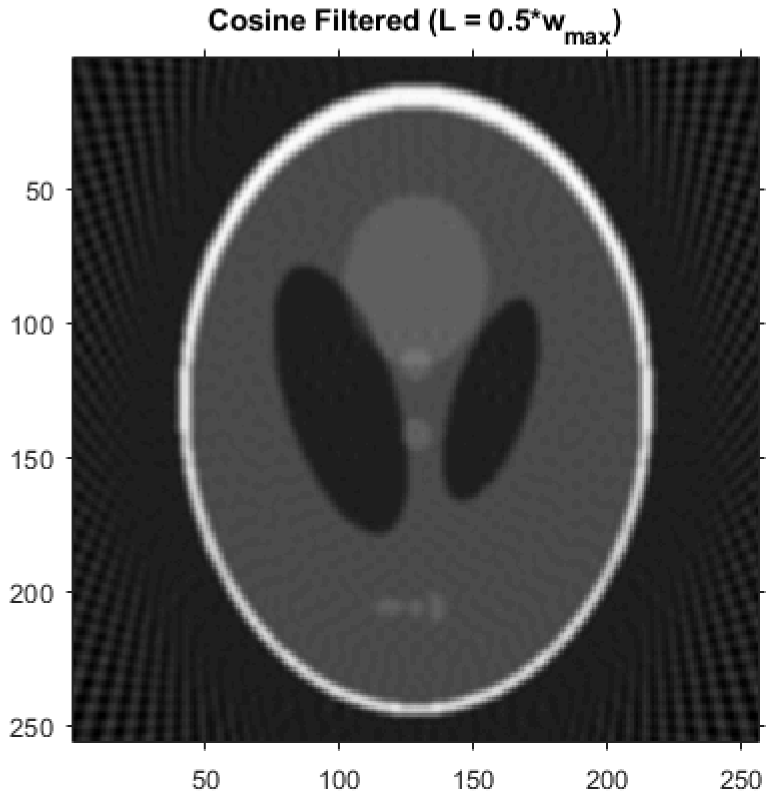




## Filtered Back-Projection (Low-Pass Cosine Filter)

```
filt_rad_full = myFilter(rad,1,'Cosine');
filt_rad_half = myFilter(rad,0.5,'Cosine');

img_full = rescale(iradon(filt_rad_full,theta,'linear','none',1,n));
img_half = rescale(iradon(filt_rad_half,theta,'linear','none',1,n));

figure;
imshow(img_full);
title('Cosine Filtered (L = w_{max})')
figure;
imshow(img_half);
title('Cosine Filtered (L = 0.5*w_{max})')
```


Cosine Filtered (L = w$_{max}$)
50
100
150
200
250
50
100
150
200
250


Cosine Filtered (L = 0.5*w$_{max}$)
50
100
150
200
250
50
100
150
200
250

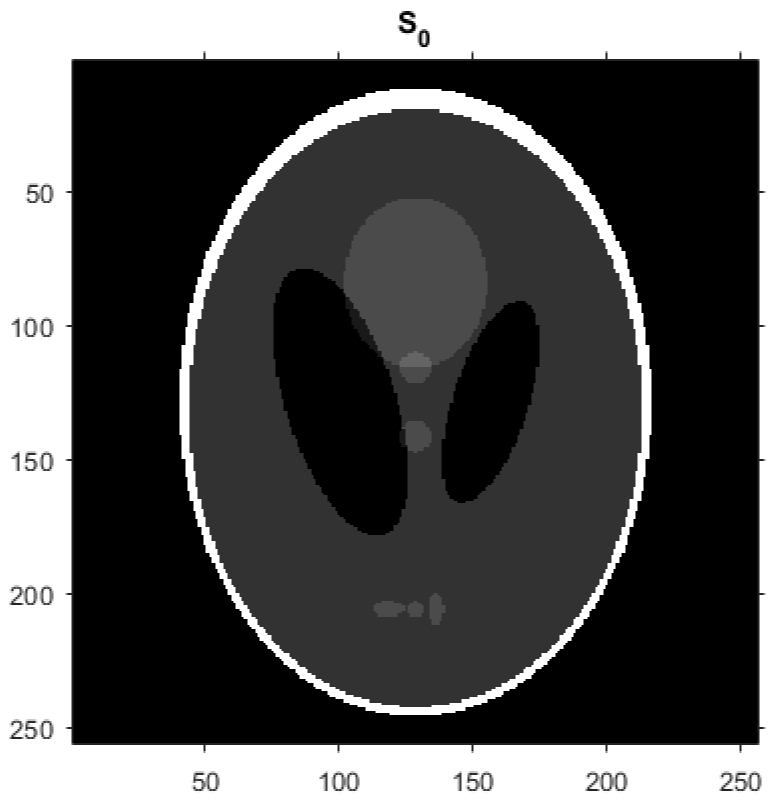
## Observations for the Filtered Reconstructed Images

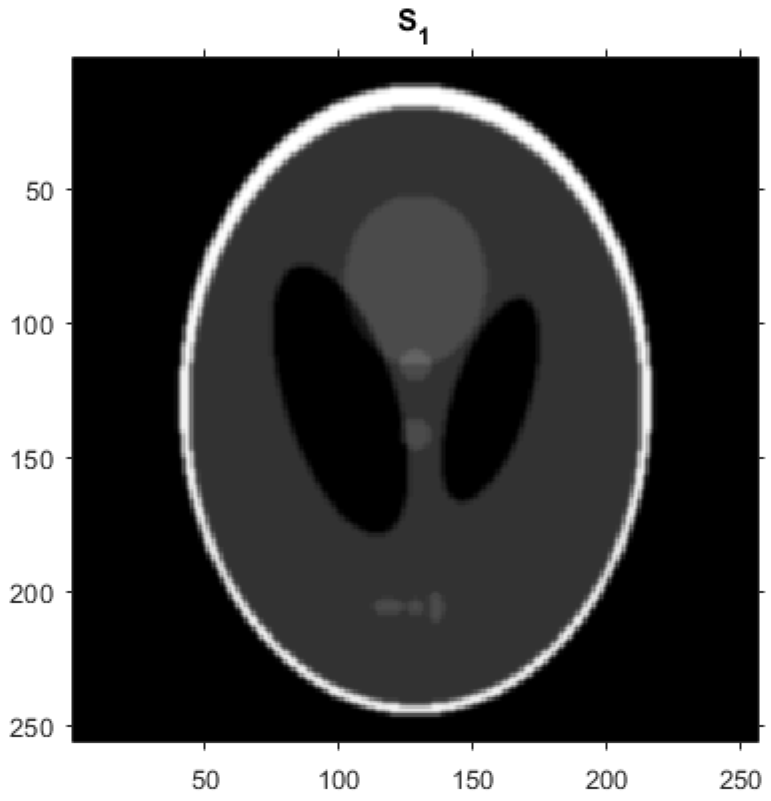
For all the filters, in L=w_max and L=w_max/2, the L=w_max filter has sharper edges, reason being that sharpness features are represented by higher frequencies which could not be captured in the L=w_max/2 filter. Apart from this, the larger patches of black and gray colors, i.e the smooth features, are smooth for the L=w_max/2 filter as the higher frequencies cause noise which haven't been captured by this filter and cause noise in the L=w_max filter.

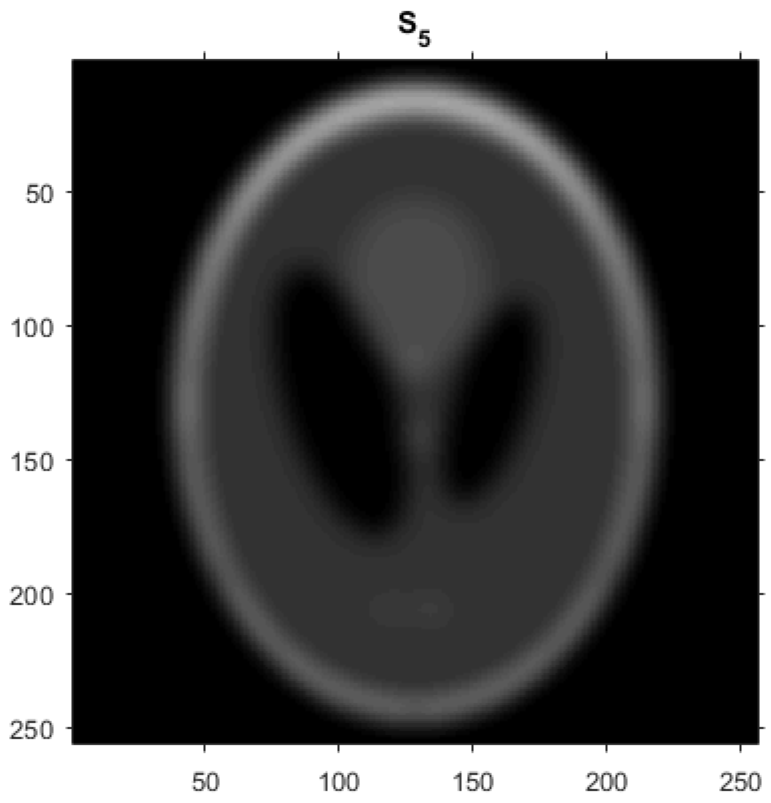
Among the Ram-Lak, Shepp-Logan and Cosine filters, Ram-Lak filter gives sharp edges as it gives higher weight to the high frequencies which correspond to sharp features and dampens lower frequencies but at the same time it is a bit noisy due to high frequencies. Shepp-Logan gives sharp edges, but is also smoother than Ram-Lak, as it gives higher weight to the lower frequencies which correspond to smooth features. Cosine filter gives the smoothest images as it dampens the high frequency values as well thereby reducing the noise even more.

## Generate and display blurred images

```
img0 = img;
img1 = imgaussfilt(img,1);
img5 = imgaussfilt(img,5);

figure;
imshow(img0);
title('S_{0}');
figure;
imshow(img1);
title('S_{1}');
figure;
imshow(img5);
title('S_{5}');
```


$S_0$
50
100
150
200
250
50
100
150
200
250


$S_1$
50
100
150
200
250
50
100
150
200
250



## Ram-Lak filter reconstruction of blurred images

Here we notice the RRMSE values of the reconstructed images for the various blurred images. The highest RRMSE is for (S0,R0) and lowest for (S5,R5). In S0, there are smooth features(the black spaces and gray spaces) and sharp features(the edges and they haven't been blurred), both. So the Ram-Lak filter dampens the low frequency values that correspond to the smooth features and amplifies moderately high frequencies corresponding to sharp features. But here we can see a trade-off between the sharp and smooth features. A higher L will give more weightage to the sharp features than the smooth ones. Since S0 has both, sharp and smooth features, the RRMSE is high as the smooth features get underrepresented. On the other hand, S5 is already blurred by the Gaussian convolution and doesn't have as sharp features as S0. So for blurred images like S5, having L=w_max also doesn't hurt the smooth features as essentially there are only smooth features in the blurred image(sharp features are blurred). So for S5, we mainly have smooth features and Ram-Lak preserves them, so the RRMSE is less in this case. This justifies the order of RRMSE values obtained. Higher the blur in the original image, lower will be the RRMSE of the reconstructed image.

```
[rad0, ~] = radon(img0, theta);
[rad1, ~] = radon(img1, theta);
[rad5, ~] = radon(img5, theta);

filt_rad0 = myFilter(rad0,1,'Ram-Lak');
filt_rad1 = myFilter(rad1,1,'Ram-Lak');
filt_rad5 = myFilter(rad5,1,'Ram-Lak');

back_img0 = iradon(filt_rad0,theta,'linear','none',1,n);
```

```
back_img1 = iradon(filt_rad1,theta,'linear','none',1,n);
back_img5 = iradon(filt_rad5,theta,'linear','none',1,n);

error0 = rrmse(img,back_img0);
error1 = rrmse(img1,back_img1);
error5 = rrmse(img5,back_img5);

fprintf("RRMSE(S0,R0) : %.4f\n", error0);
fprintf("RRMSE(S1,R1) : %.4f\n", error1);
fprintf("RRMSE(S5,R5) : %.4f\n", error5);
```

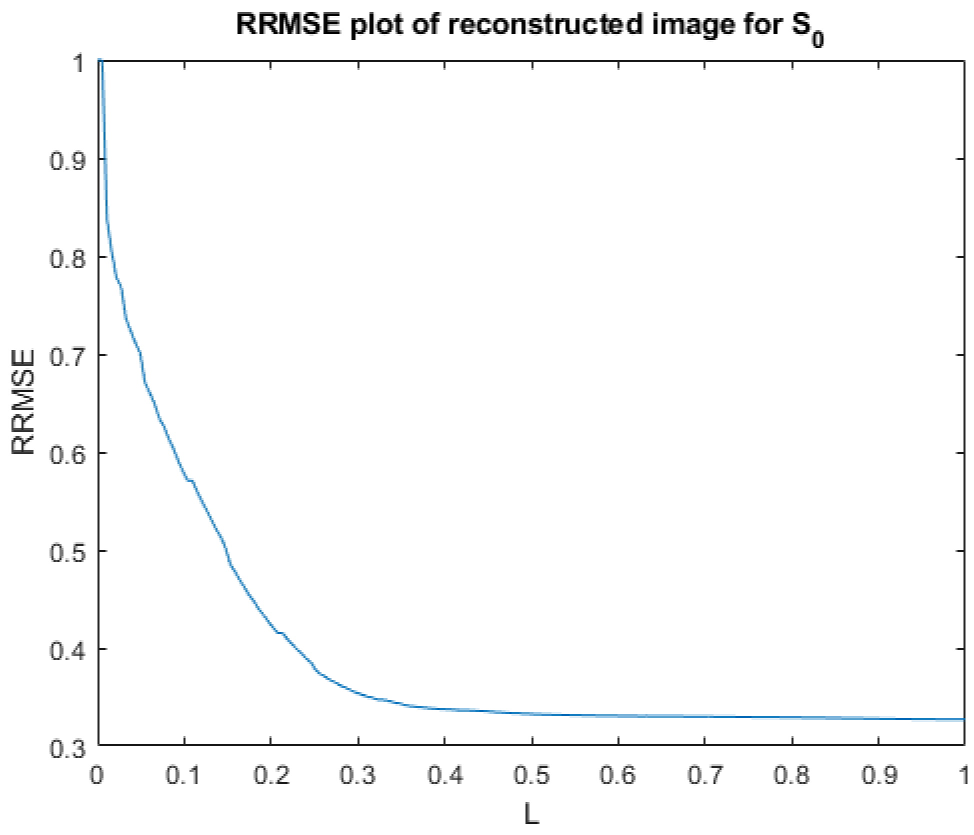
```
RRMSE(S0,R0) : 0.3270
RRMSE(S1,R1) : 0.2079
RRMSE(S5,R5) : 0.2013
```

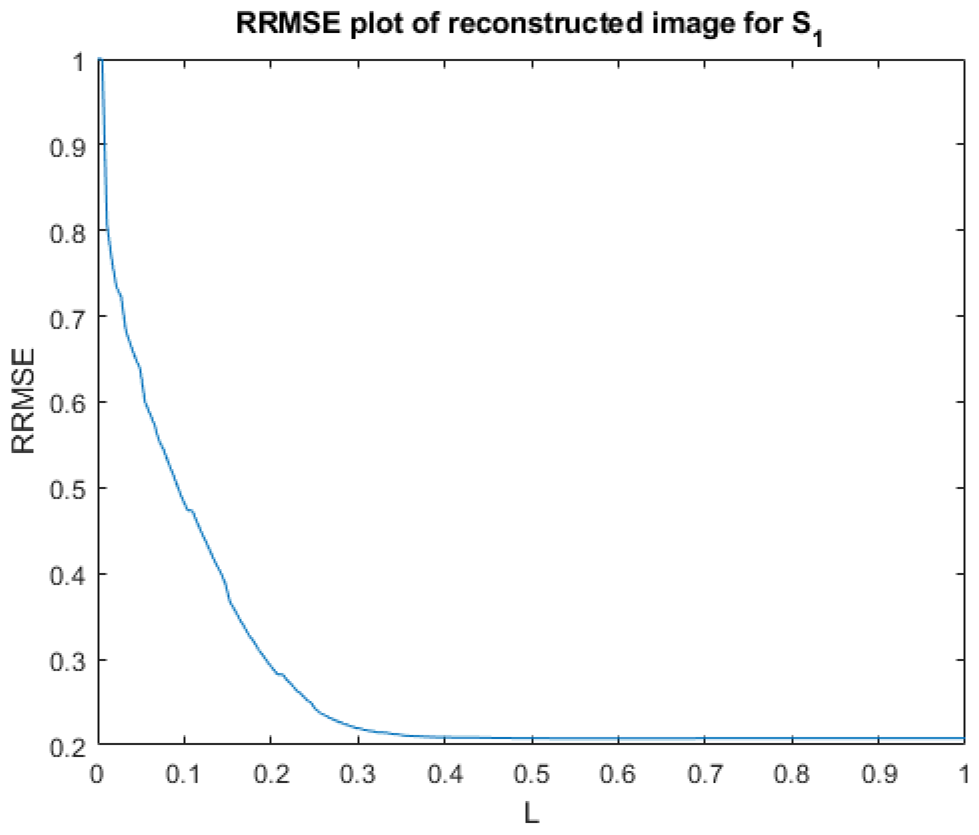
## RRMSE plots for reconstructed images

The observation in the plots here is that the RRMSE keeps decreasing with increasing L. To represent image boundaries and sharpness, we need moderately high frequency values. With a low L, we cut-off the high frequencies and hence these boundaries and sharpness features won't be captured and thereby causing a high RRMSE. As L increases, we are able to capture the high frequency values and hence the sharpness features, causing lower RRMSE.

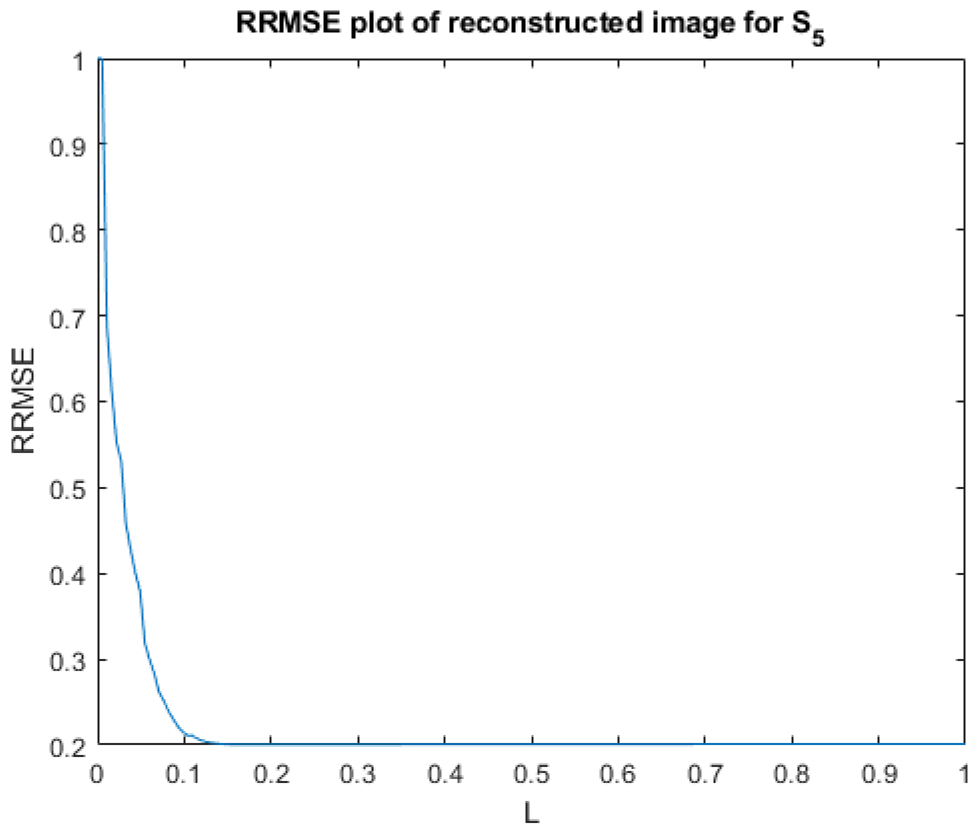
```
L = 0: 2/(size(rad,1)-1) : 1;
error0 = zeros(size(L));
error1 = zeros(size(L));
error5 = zeros(size(L));

for i = 1 : size(L,2)
    % generate filtered radon for particular l
    filt_rad0 = myFilter(rad0,L(i),'Ram-Lak');
    filt_rad1 = myFilter(rad1,L(i),'Ram-Lak');
    filt_rad5 = myFilter(rad5,L(i),'Ram-Lak');

    % generate image from filtered radon
    back_img0 = iradon(filt_rad0,theta,'linear','none',1,n);
    back_img1 = iradon(filt_rad1,theta,'linear','none',1,n);
    back_img5 = iradon(filt_rad5,theta,'linear','none',1,n);

    % compute rrmse
    error0(i) = rrmse(img0,back_img0);
    error1(i) = rrmse(img1,back_img1);
    error5(i) = rrmse(img5,back_img5);
end

figure;
plot(L,error0);
title("RRMSE plot of reconstructed image for S_{0}");
xlabel('L');
ylabel('RRMSE');

figure;
plot(L,error1);
```

```
xlabel('L');
ylabel('RRMSE');
title("RRMSE plot of reconstructed image for S_{1}");

figure;
plot(L,error5);
xlabel('L');
ylabel('RRMSE');
title("RRMSE plot of reconstructed image for S_{5}");
```





RRMSE plot of reconstructed image for $S_1$
RRMSE
L



RRMSE plot of reconstructed image for $S_5$
RRMSE
L

*Published with MATLAB® R2019a*