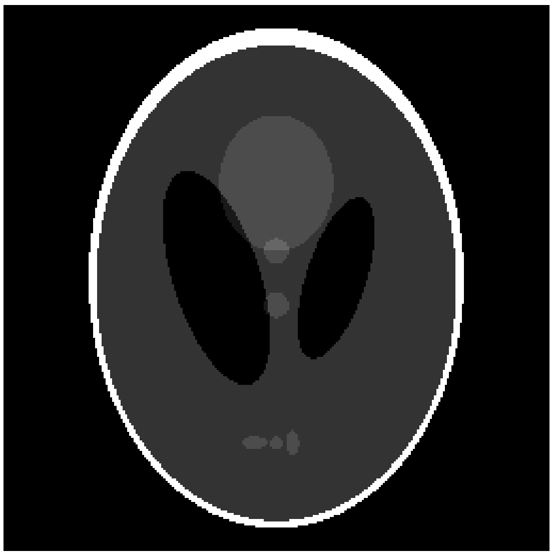
CS 736: Medical Image Processing: Assignment 1 Question 2: Aditya Kumar Akash 120050046, Praveen Agrawal 12D020030

Contents

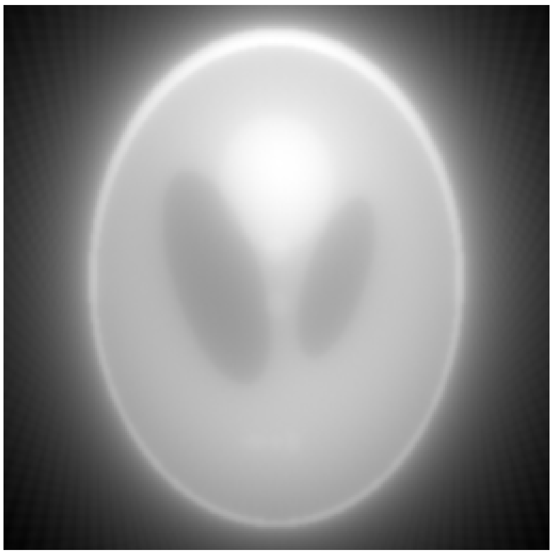
Part a

```
f = phantom(256);
[a, b] = size(f);
theta = (0:3:177);
[R, xp] = radon(f, theta);
backProjection = iradon(R, theta, 'none', a);
figure;
imshow(f);
title('Original phantom image', 'FontWeight', 'bold');

figure;
imshow(mat2gray(backProjection));
title('Unfiltered Back Projection', 'FontWeight', 'bold');

typeOfFilter = cellstr(['RL'; 'SL'; 'C ']);
filterName = cellstr(['Ram Lak Filter    '; 'Shepp Logan filter'; 'Cosine filter     ']);
% RL -> Ram-Lak filter,
% SL -> Shepp-Logan filter
% C -> Cosine filter
L = [0.5, 1];
% L = 0.5 => Wmax/2
% L = 1 => Wmax

for i = 1:1:length(typeOfFilter)
    for j = 1:1:length(L)
        ift = myFilter(R, xp, typeOfFilter(i), L(j));
        filteredBackProjection = iradon(ift, theta, 'none', a);
        figure;
        imshow(mat2gray(filteredBackProjection));
        title(strcat(filterName(i),'with L = ',num2str(L(j)),'*Wmax'), 'FontWeight', 'bold');
    end
end

% Shepp logan filter seems to be performing the best among the three
% filters we are considering. This is evident as the shepp logan filter
% best satisfies the conditions needed for a good practical filter, which
% are, it behaves like |w| near the origin, the sinusoidal function (B(w))
% is smooth near w = 0 and it eliminates high frequencies beyond L.
% However, for a given filter the two values of
% L do not create much of a difference.
```

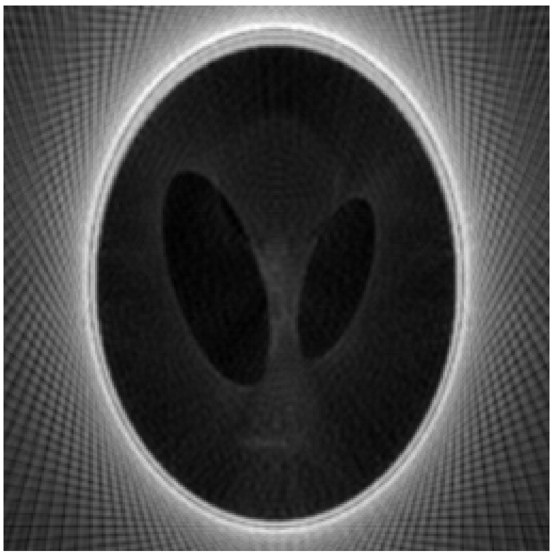
Original phantom image



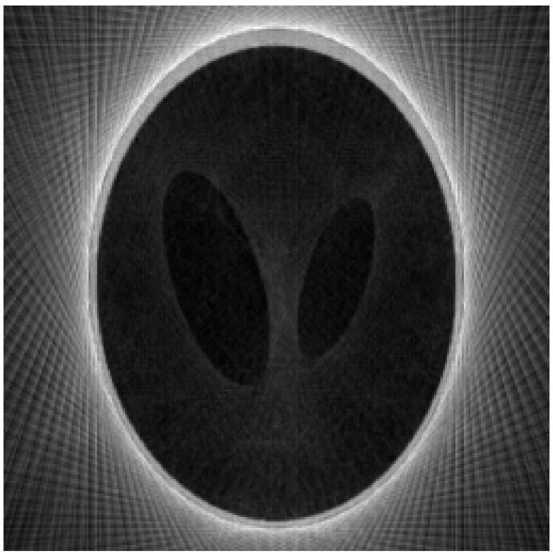
Unfiltered Back Projection



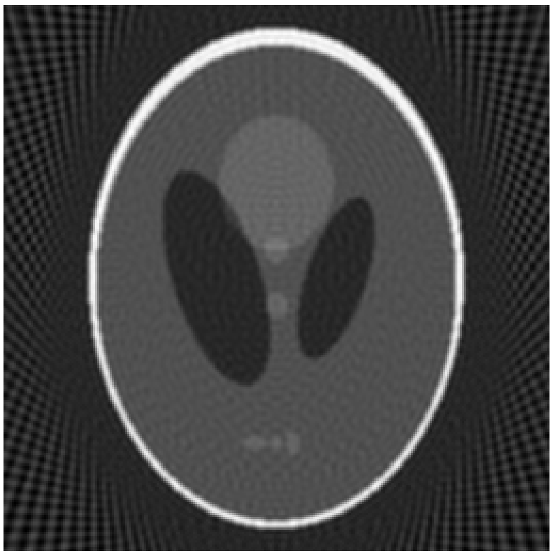
Ram Lak Filterwith L =0.5*Wmax



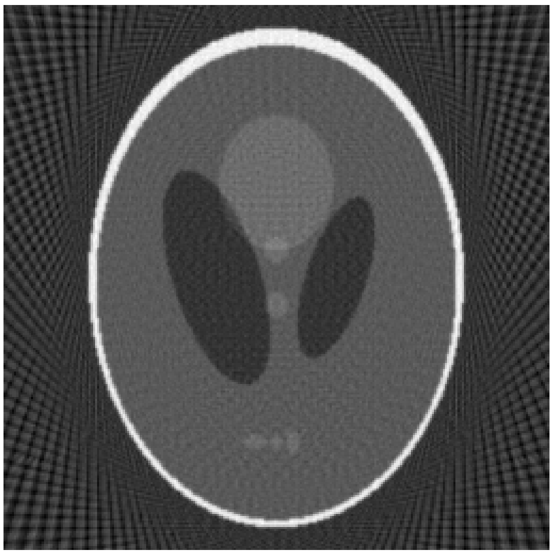
Ram Lak Filterwith L =1*Wmax

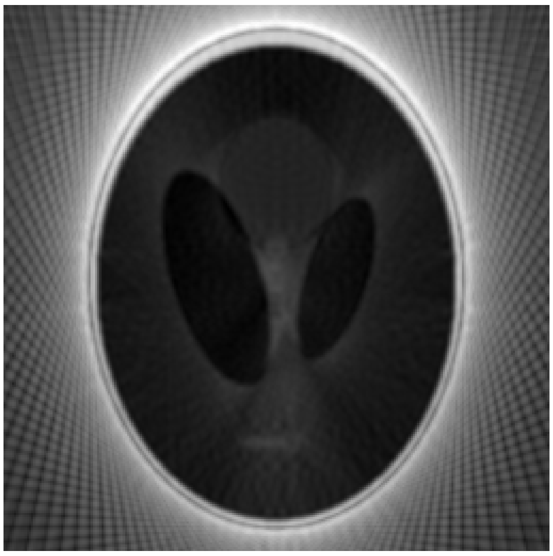


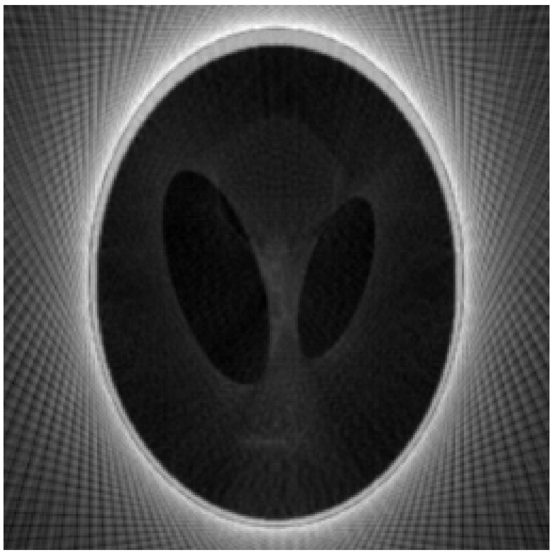
Shepp Logan filterwith L =0.5*Wmax



Shepp Logan filterwith L =1*Wmax





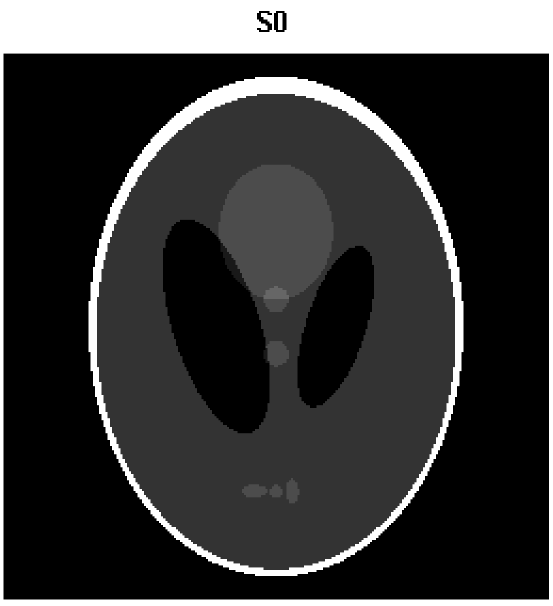


Part b

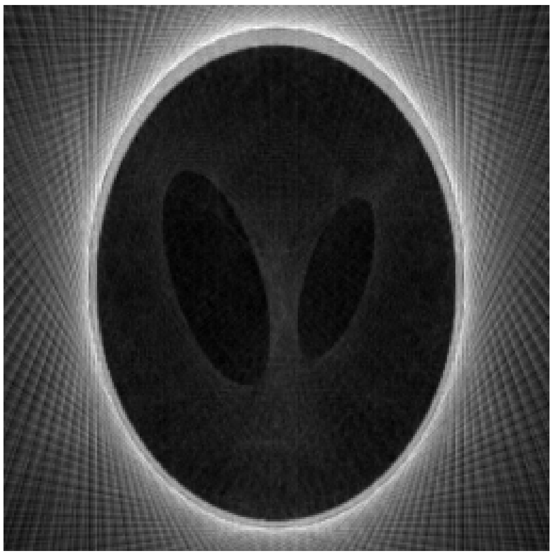
```
S0 = f;
mask = fspecial ('gaussian', 11, 1);
S1 = conv2 (f, mask, 'same');
mask = fspecial ('gaussian', 51, 5);
S5 = conv2 (f, mask, 'same');

[r0, xp0] = radon(S0, theta);
[r1, xp1] = radon(S1, theta);
[r5, xp5] = radon(S5, theta);

ift0 = myFilter(r0, xp0, 'RL', 1);
ift1 = myFilter(r1, xp1, 'RL', 1);
ift5 = myFilter(r5, xp5, 'RL', 1);
```

```
R0 = iradon(ift0, theta, 'none', a);
R1 = iradon(ift1, theta, 'none', a);
R5 = iradon(ift5, theta, 'none', a);

rrmse0 = RRMSE(R0,S0);
rrmse1 = RRMSE(R1,S0);
rrmse5 = RRMSE(R5,S0);

figure;
imshow(mat2gray(S0));
title('S0', 'FontWeight', 'bold');
figure;
imshow(mat2gray(R0));
title('R0', 'FontWeight', 'bold');
figure;
disp(strcat('RRMSE for R0 =  ',num2str(rrmse0)));
imshow(mat2gray(S1));
title('S1', 'FontWeight', 'bold');
figure;
imshow(mat2gray(R1));
title('R1', 'FontWeight', 'bold');
disp(strcat('RRMSE for R1 =  ',num2str(rrmse1)));
figure;
imshow(mat2gray(S5));
title('S5', 'FontWeight', 'bold');
figure;
imshow(mat2gray(R5));
title('R5', 'FontWeight', 'bold');
disp(strcat('RRMSE for R5 =  ',num2str(rrmse5)));
```

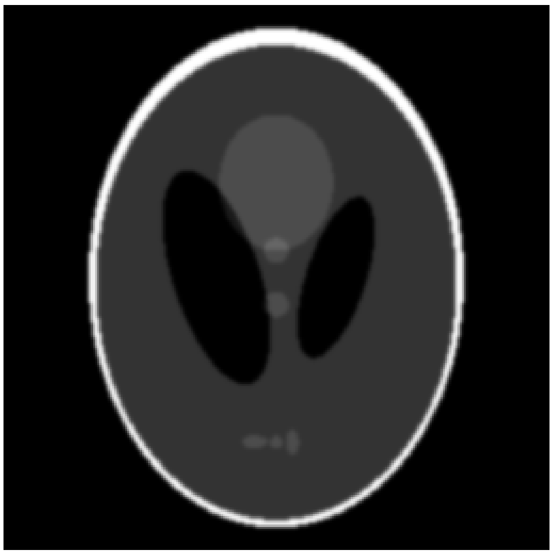
```
RRMSE for R0 =0.875
RRMSE for R1 =0.87911
RRMSE for R5 =0.91819
```



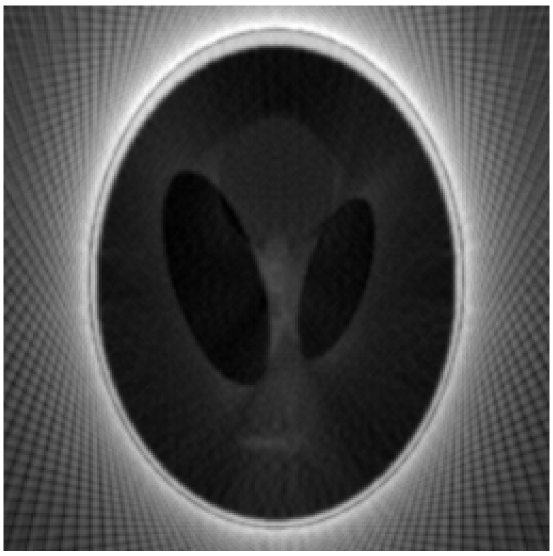
R0



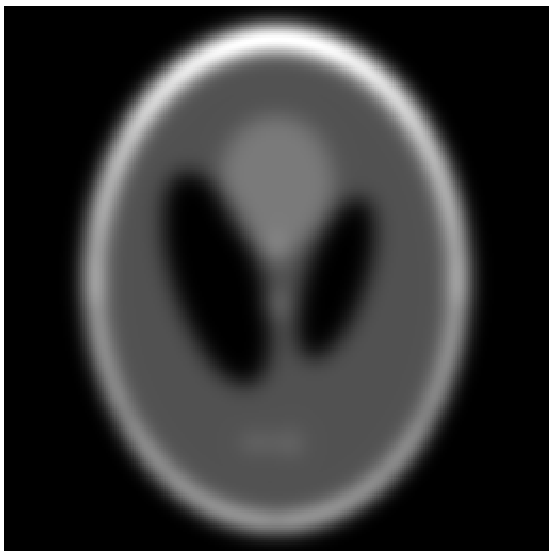
S1

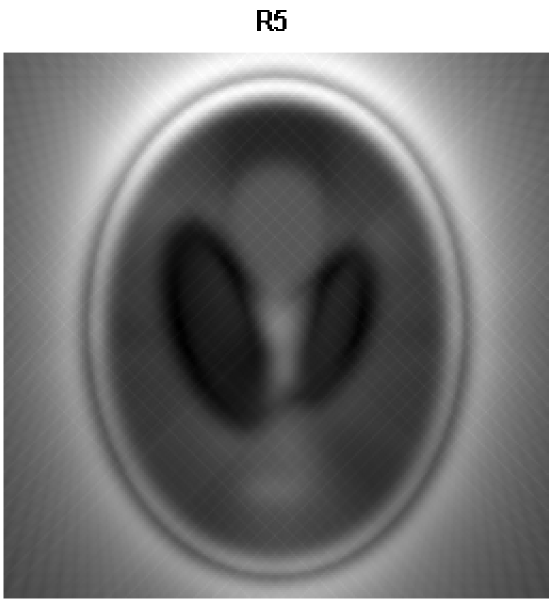


R1



S5

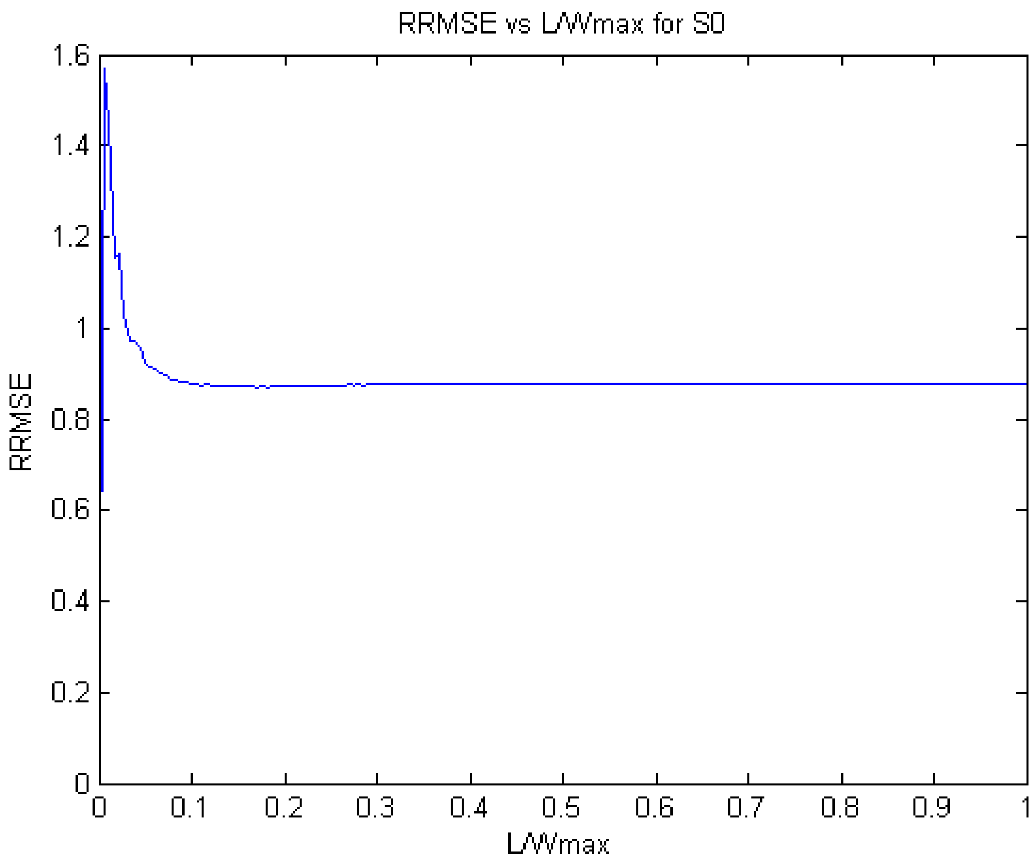


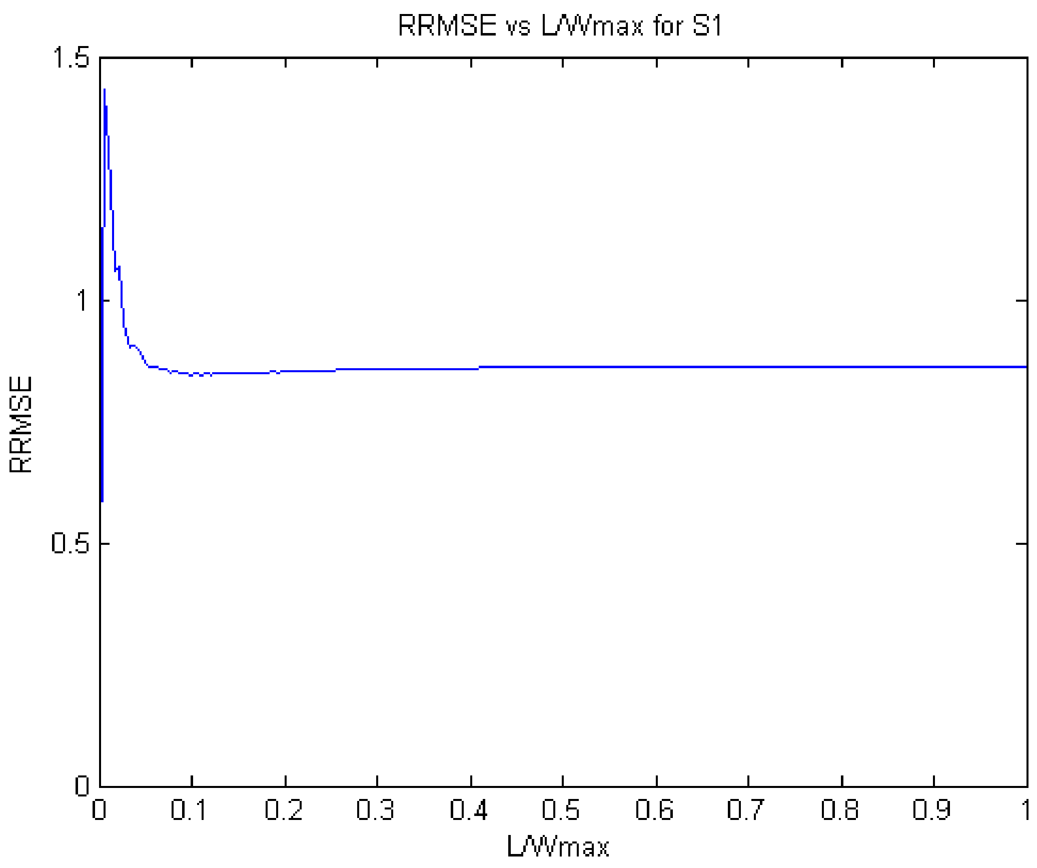


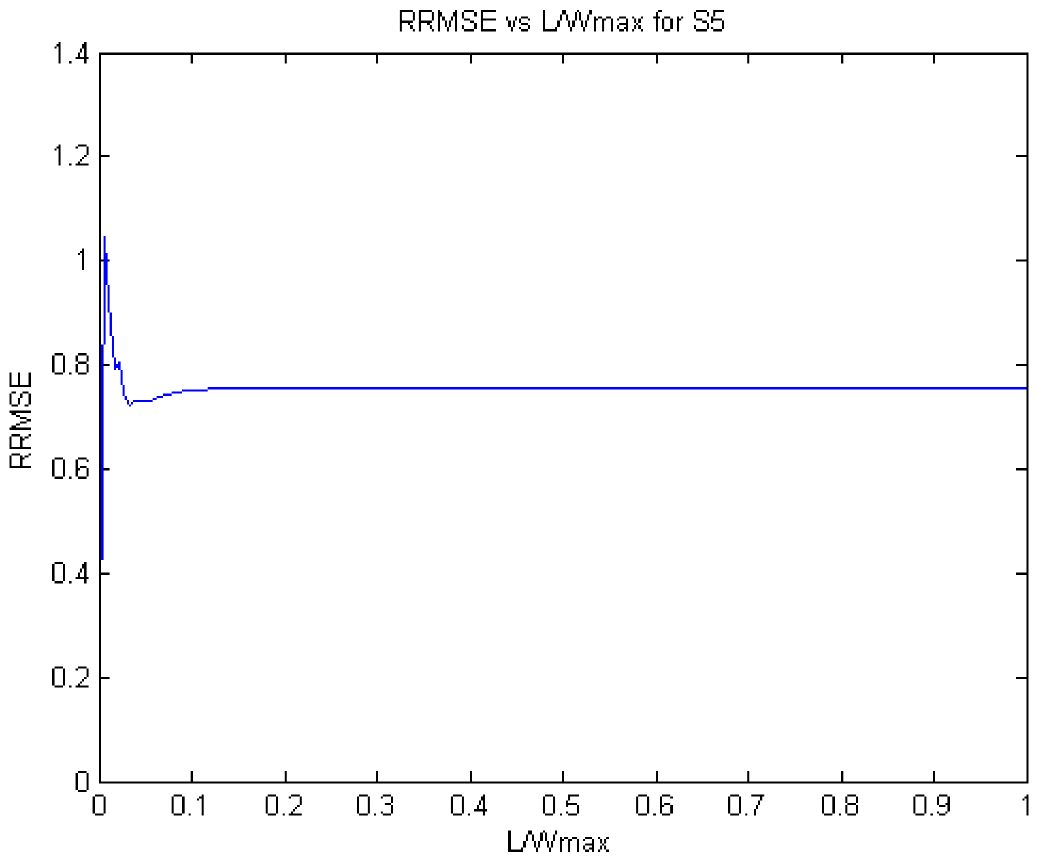
Part c

```
L = linspace(0,1,max(xp0));
rrmse0 = zeros(1,length(L));
rrmse1 = zeros(1,length(L));
rrmse5 = zeros(1,length(L));
for i = 1:1:length(L)
    ift0 = myFilter(r0, xp0, 'RL', L(i));
    ift1 = myFilter(r1, xp1, 'RL', L(i));
    ift5 = myFilter(r5, xp5, 'RL', L(i));

    R0 = iradon(ift0, theta, 'none', a);
    R1 = iradon(ift1, theta, 'none', a);
    R5 = iradon(ift5, theta, 'none', a);

    rrmse0(i) = RRMSE(R0,S0);
    rrmse1(i) = RRMSE(R1,S1);
    rrmse5(i) = RRMSE(R5,S5);
end
figure;
plot(L,rrmse0);
title('RRMSE vs L/Wmax for S0');
xlabel('L/Wmax');
ylabel('RRMSE');

figure;
plot(L,rrmse1);
title('RRMSE vs L/Wmax for S1');
xlabel('L/Wmax');
ylabel('RRMSE');
figure;
plot(L,rrmse5);
title('RRMSE vs L/Wmax for S5');
xlabel('L/Wmax');
ylabel('RRMSE');

% We see that S5 being the most smoothed out image converges to the minimum
% error faster than the other two images.
```

```
Warning: Rank deficient, rank = 0, tol =  0.000000e+00.
Warning: Rank deficient, rank = 0, tol =  0.000000e+00.
Warning: Rank deficient, rank = 0, tol =  0.000000e+00.
```







Published with MATLAB® R2014a