Medical Image Computing

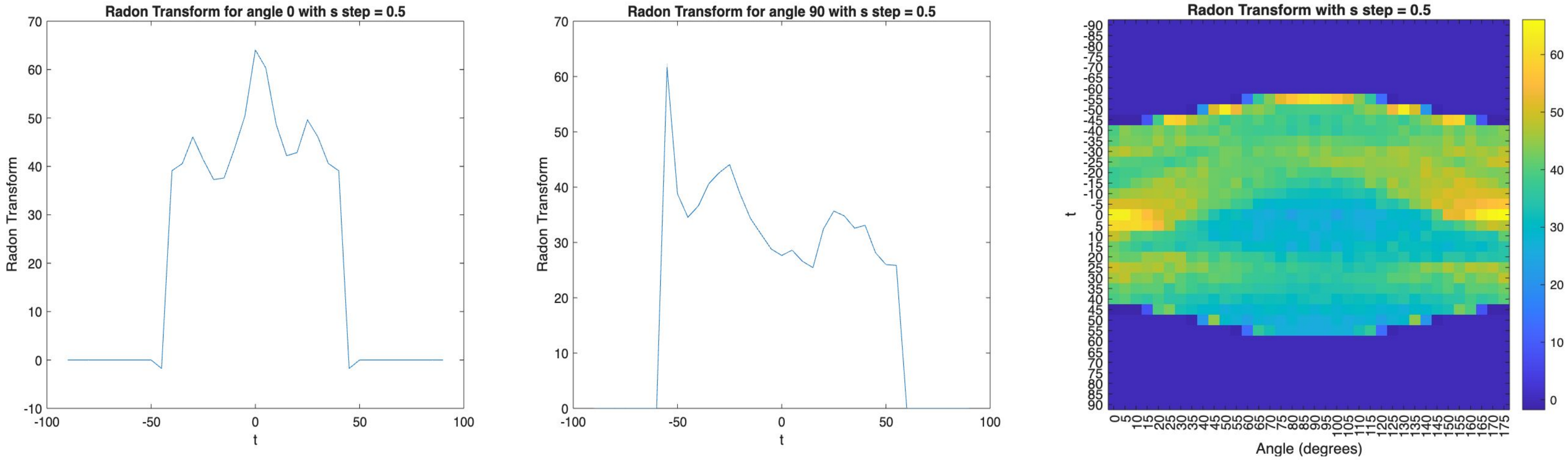
Assignment 2

Shreyas Grampurohit (21D070029)
Soham Nivargi (21D070074)

Q1 - X-Ray Computed Tomography: Radon Transform

ds <= 1, nyquist criterion

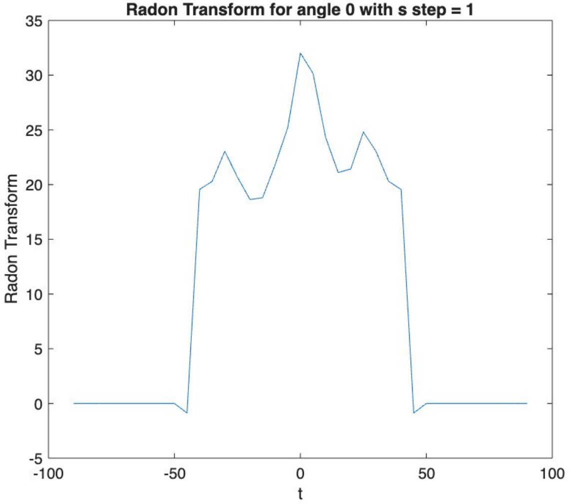
Step size = 0.5

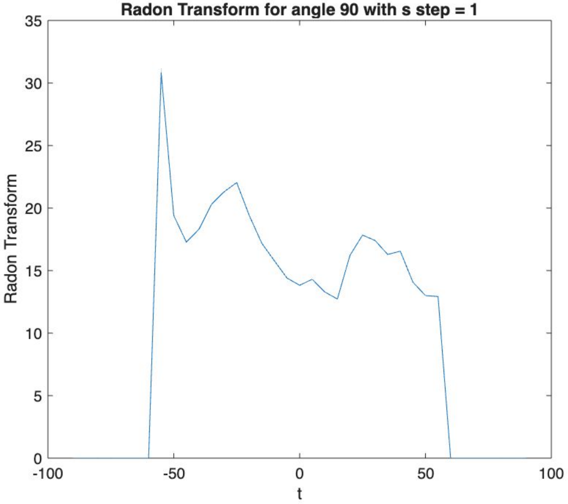


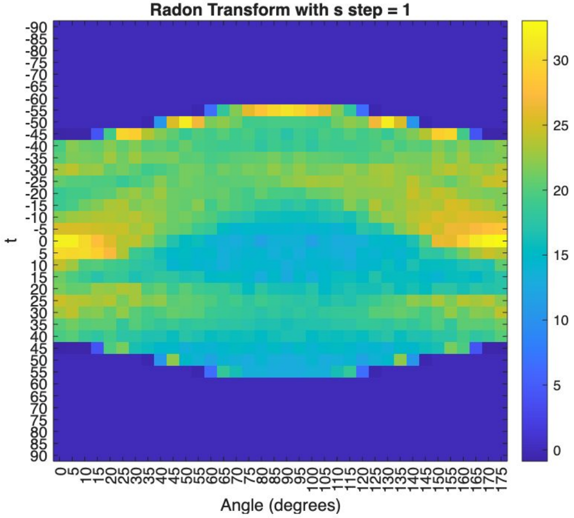
bicubic interpolation. uses large neighbourhood – so deals better with rough images

Q1 - X-Ray Computed Tomography: Radon Transform

Step size = 1

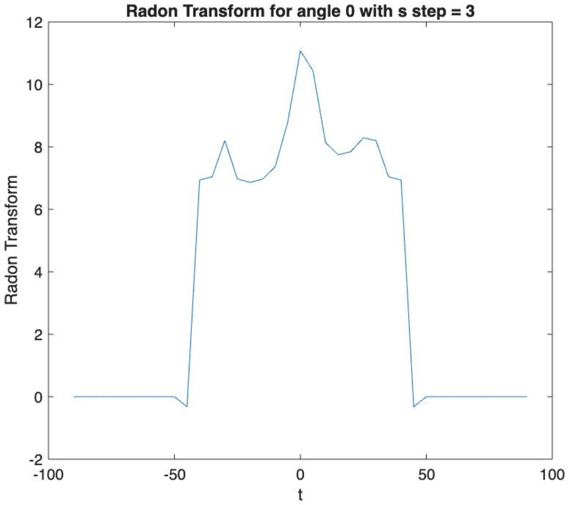


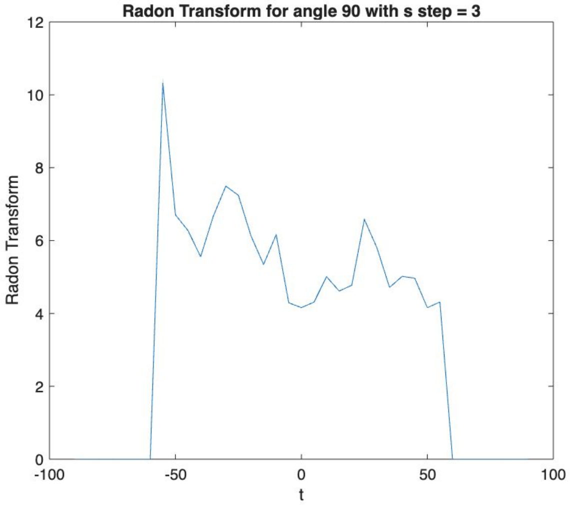


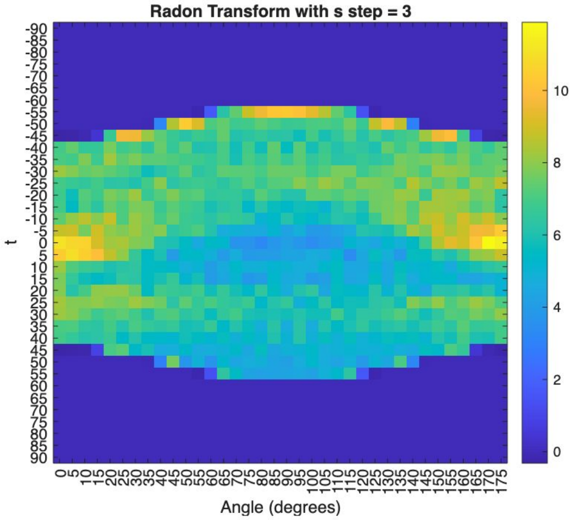


Q1 - X-Ray Computed Tomography: Radon Transform

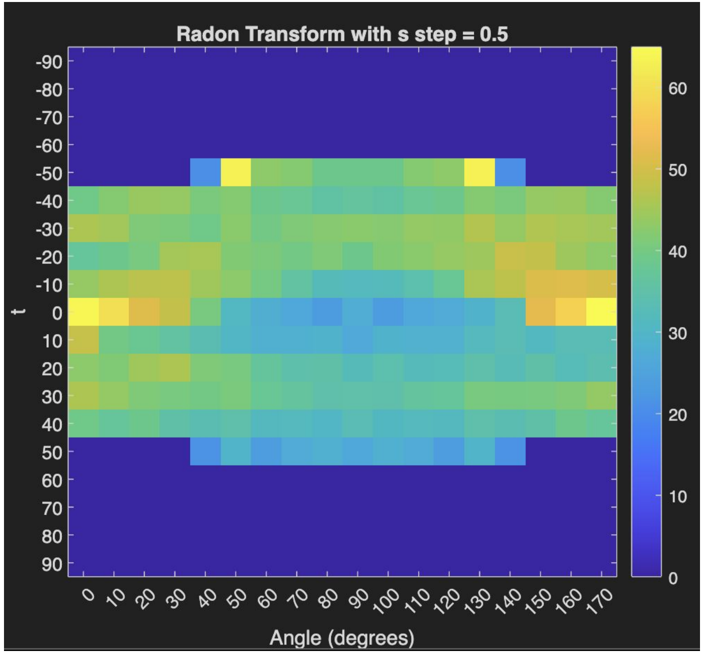
Step size = 3





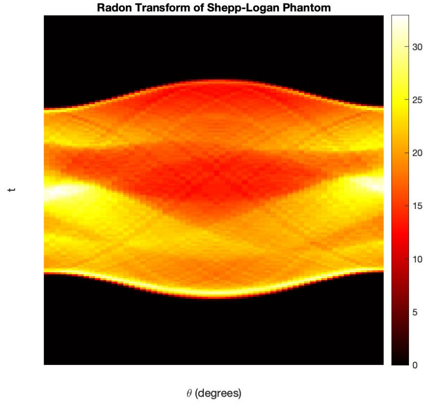


with half the number of angles and ts – lesser radiations, but information loss

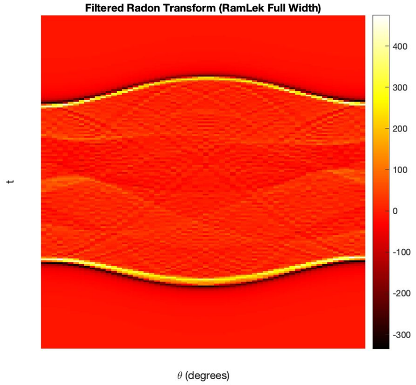


Q2 - **X-Ray Computed Tomography: Reconstruction by Filtered Backprojection (FBP)**

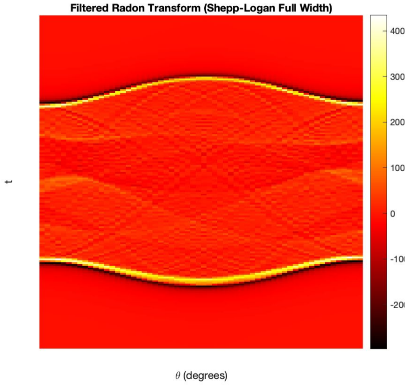
Radon Transform of phantom vs Filtered Radon Transforms for Full-width and Half-width Filters



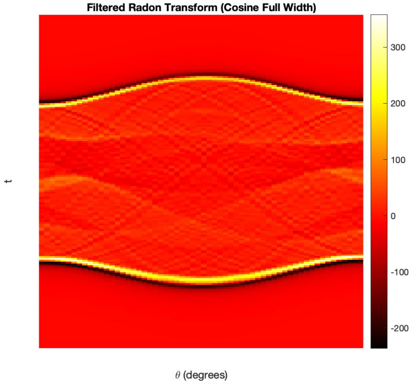
Radon Transform



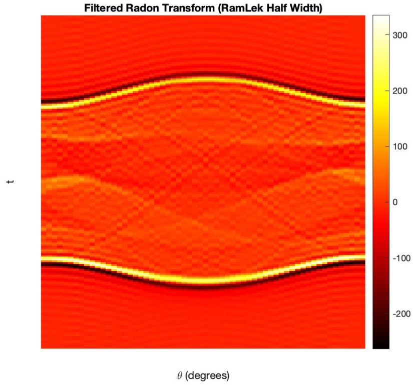
Full-width RamLek FRT



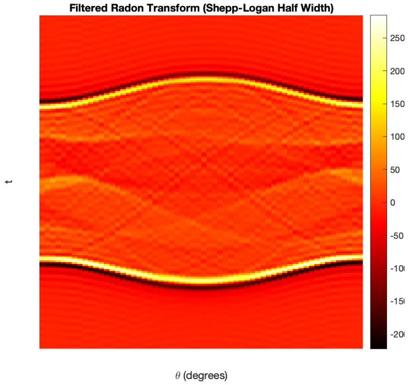
Full-width Shepp-Logan FRT



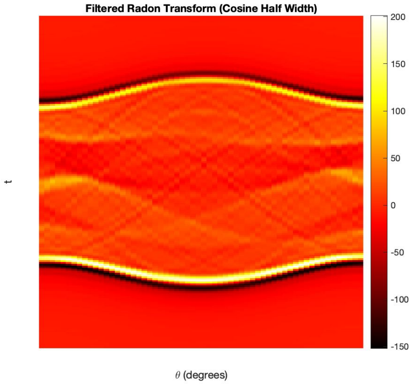
Full-width Cosine FRT



Half-width RamLek FRT



Half-width Shepp-Logan FRT

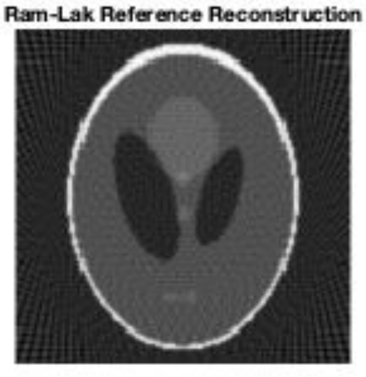


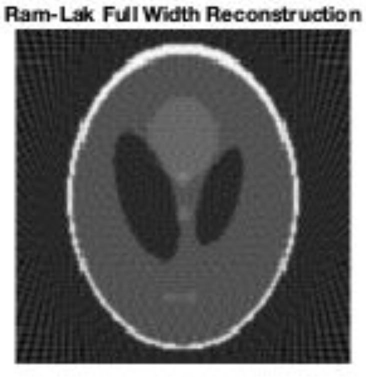
Half-width Cosine FRT

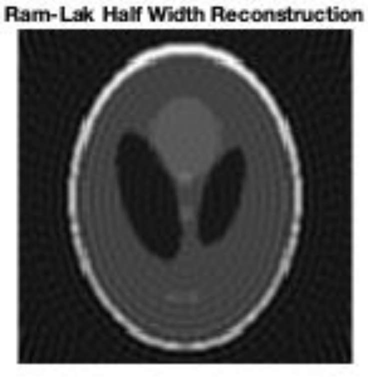
FBP Analysis

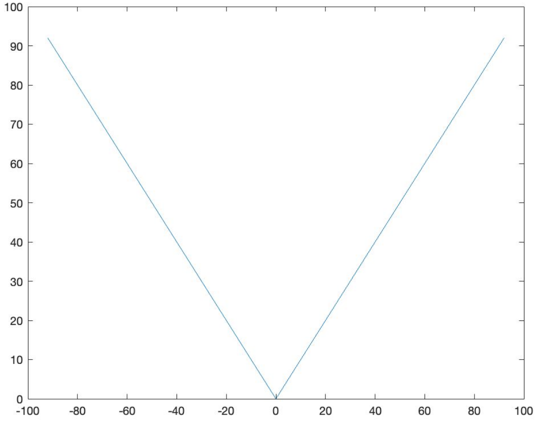
- The RamLak filter -
  - A(w) = |w|*rect(L); which amplifies higher frequencies, ie., noise and thus creates a noisy reconstruction than the other two filters. For half-width L, the image omits higher frequencies which results in image smoothing but some unavoidable loss of information
- The Shepp-logan filter -
  - A(w) = |w|*rect(L)*sinc(w/2L) which lessens the amplification of noise and is thus better than the RamLak filter. Similar to RamLak half-width, the image gets smoothened.
- Cosine Filter
  - A(w) = |w|*rect(L)*cos(πw/2L) which further lessens amplification of noise and better than both other filters.

Filtered Backprojection(FBP) for Full-width and Half-width Filters vs FBP using Matlab In-built library

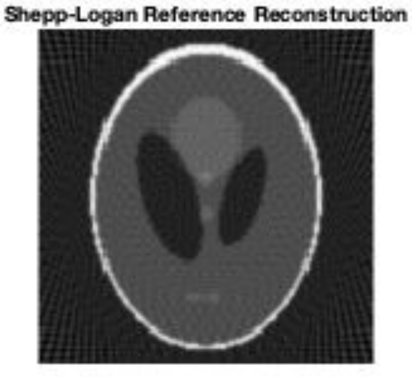


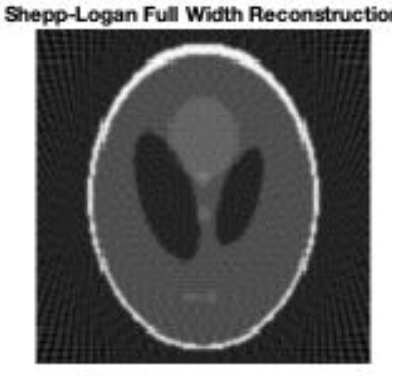


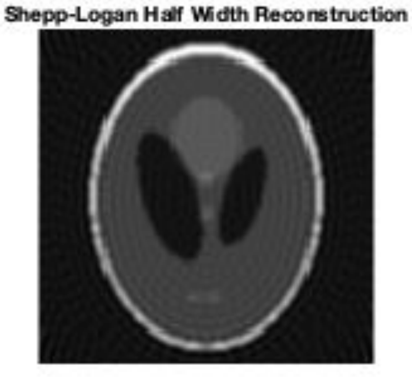


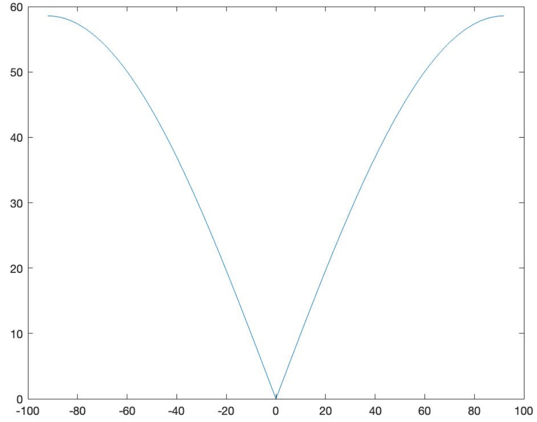


Filtered Backprojection(FBP) for Full-width and Half-width Filters vs FBP using Matlab In-built library

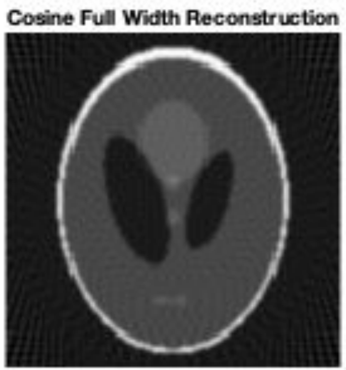


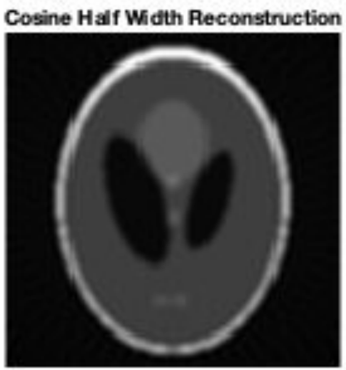


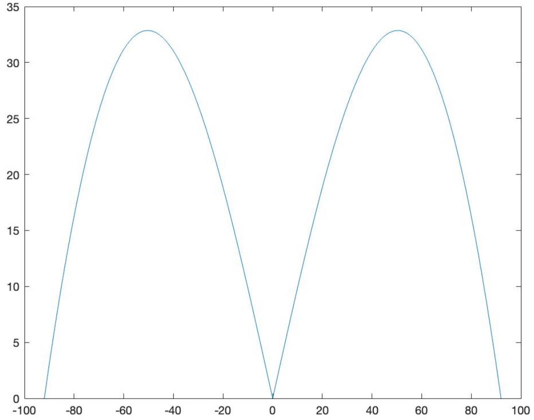




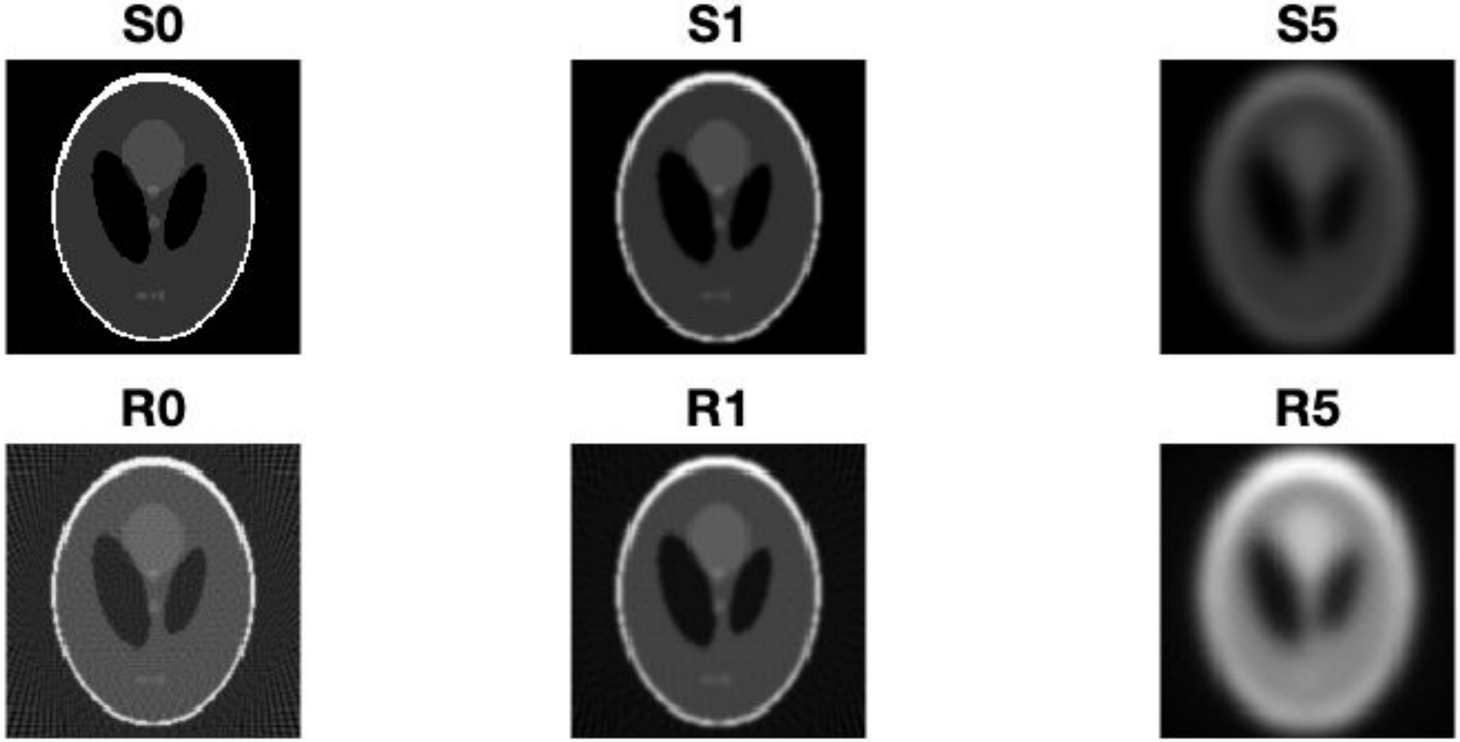
Filtered Backprojection(FBP) for Full-width and Half-width Filters vs FBP using Matlab In-built library







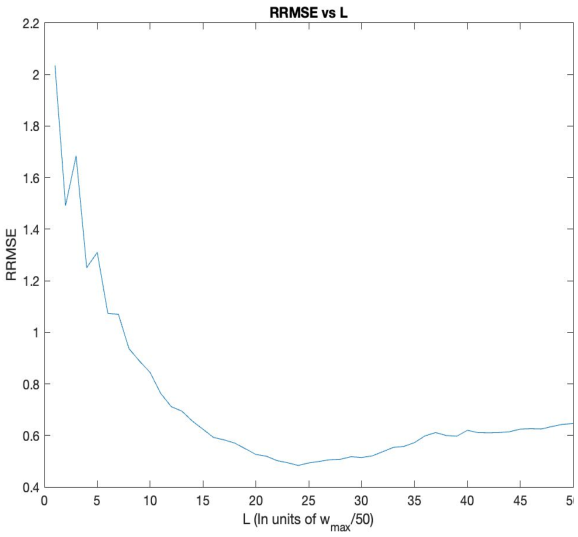
Ram-Lak Filtered Backprojection(FBP) for Blurred Images



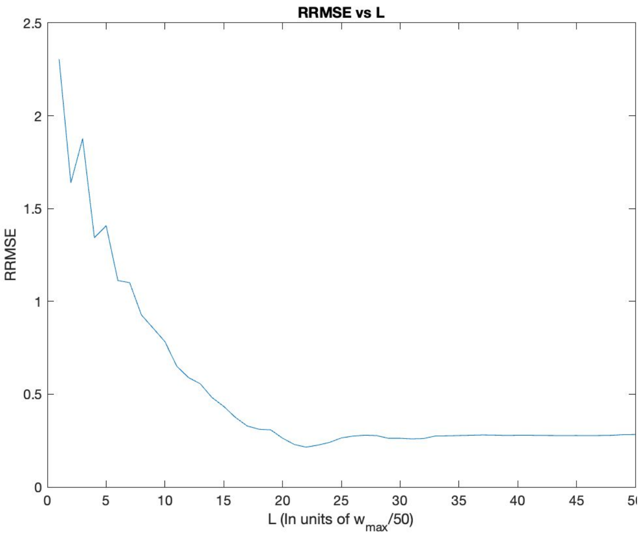
RRMSE = 0.6463 RRMSE = 0.2828 RRMSE = 1.6110

Since, the whole frequency band is considered, the image with gaussian blurring with a standard deviation of 1 already smoothens the input image and thus the reconstruction is closer to it. The image with standard deviation smoothens the image too much causing artifacts in reconstruction.

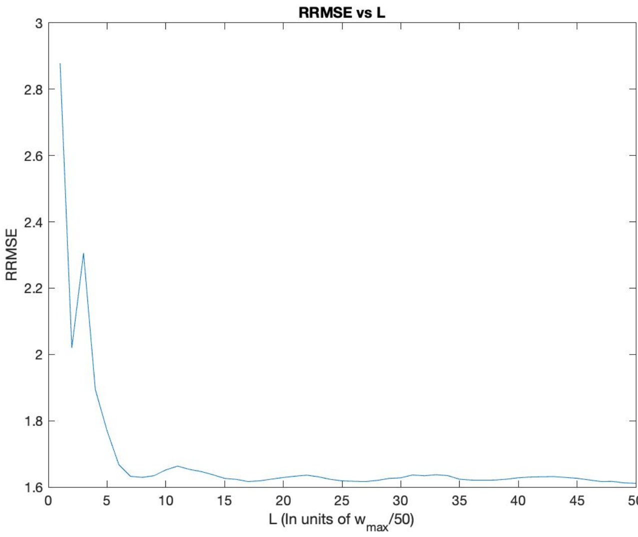
RRMSE for Ram-Lak Filtered Backprojection(FBP) for different Filter Lengths



S0 image



S1 image



S5 image

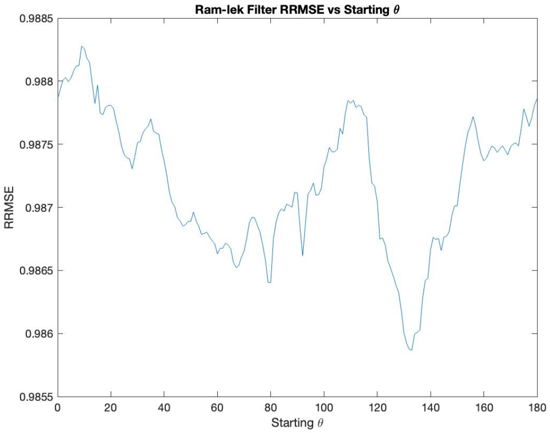
The RRMSE reduces as a larger length filter is taken, it reaches a minimum almost around half width of maximum frequencies and almost remains constant later.

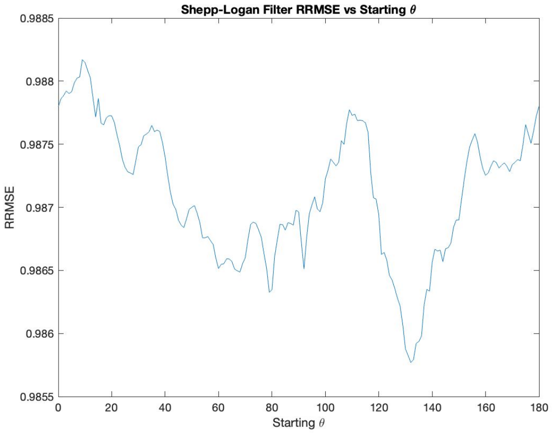
Q3 - **X-Ray Computed Tomography: Incomplete Data**

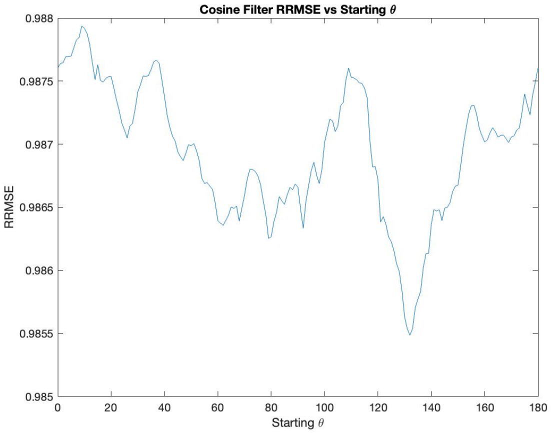
Chest CT (Original)

My Phantom (Original)

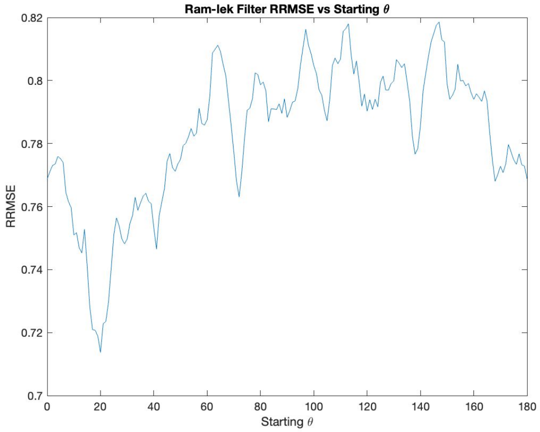
CHEST CT Data

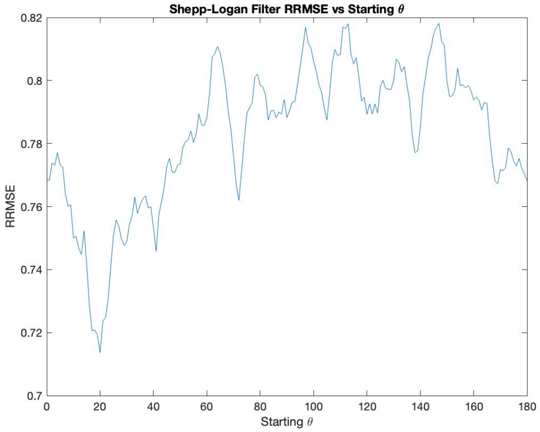


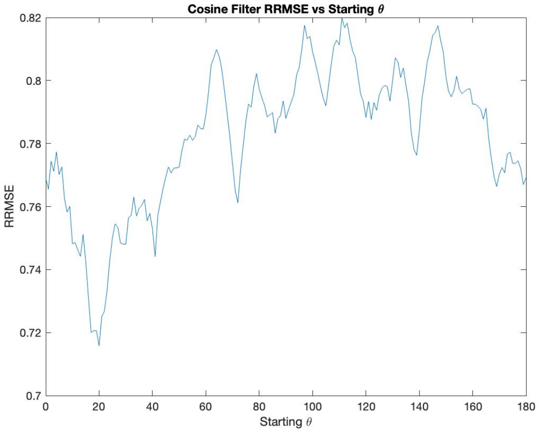




My phantom Data





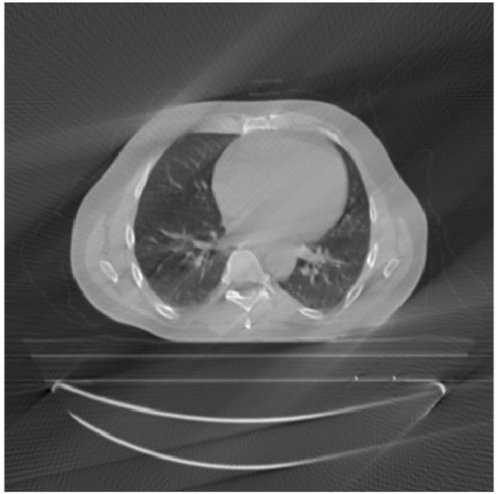


RRMSE vs Starting angle(θ) (all three filters for both datasets)

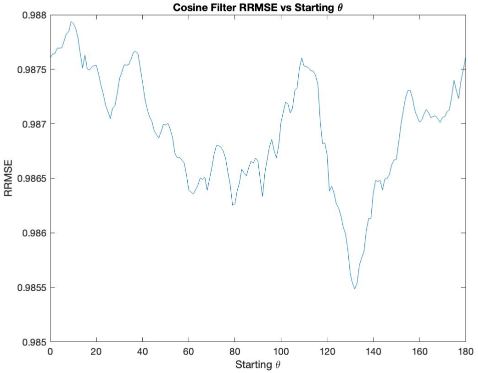
FBP with θ for which RRMSE is least

Chest CT (Original)

Chest CT (reconstructed)

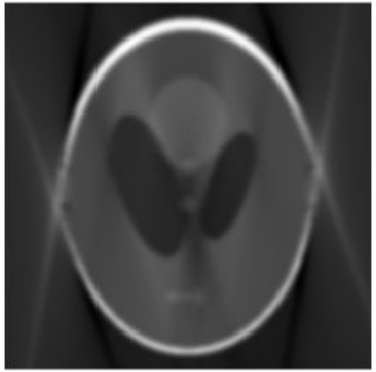


Chosen θ for Chest CT data = 132⁰ (using cosine filter)

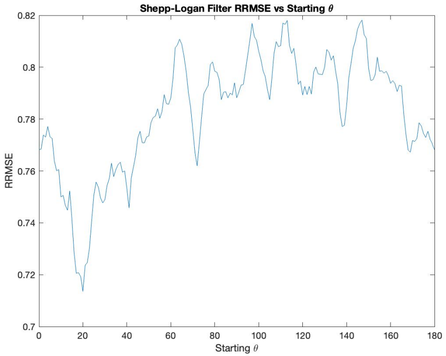


My Phantom (Original)

My phantom (reconstructed)

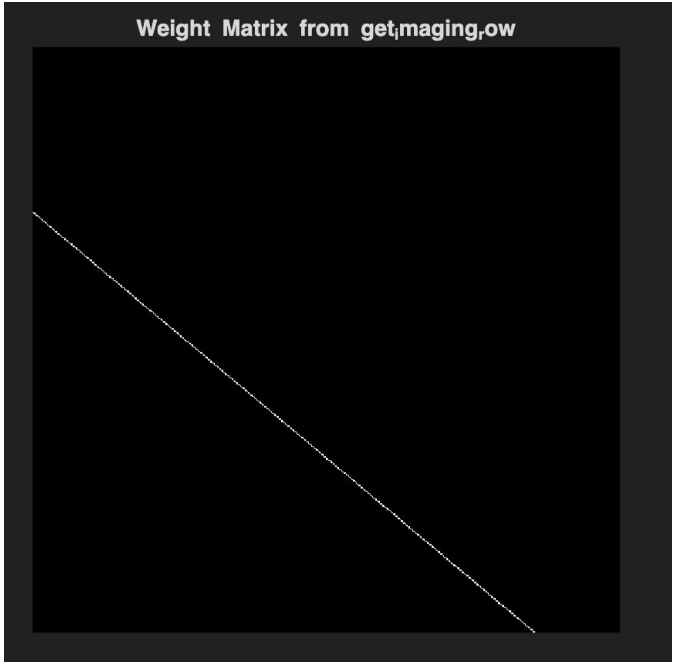


Chosen θ for My phantom data = 20⁰ (using Shepp-Logan filter)



Q4: Algebraic Reconstruction Technique (ART)

```
unction row = get_imaging_row(n, angle, t, integration_params)
    weight_matrix = zeros(n, n);
    x0 = 1 + (n - 1) / 2;
    y0 = 1 + (n - 1) / 2;

    s_step = integration_params.s_step;
    s_range = - (n - 1) * sqrt(2) / 2 : s_step : (n - 1) * sqrt(2) / 2;

    for s = s_range
        x = x0 + t * cos(angle) - s * sin(angle);
        y = y0 + t * sin(angle) + s * cos(angle);

        x_round = round(x);
        y_round = round(y);

        if (x_round < 1 || x_round > n || y_round < 1 || y_round > n)
            continue;
        end

        weight_matrix(x_round, y_round) = weight_matrix(x_round, y_round) + 1;
    end

    row = weight_matrix(:)';
nd
```



s/shreyasgrampurohit/Documents/College_Academia/Sem8/mic/Medical-Image-Computing/code/q4/myART.m

```
function x = myART(x0, permutation, b, ts, angles, art_params, integration_params)
    lambda = art_params.lambda;
    max_iter = art_params.max_iter;
    global_tol = art_params.tol;

    x = x0;
    n = sqrt(length(x0));
    for iter = 1:max_iter
        x_old = x;
        for i = permutation
            angle = angles(floor((i-1)/length(ts)) + 1);
            t_idx = mod(i-1, length(ts)) + 1;
            t = ts(t_idx);
            a_i = get_imaging_row(n, angle, t, integration_params);

            % ART update
            residual = b(i) - dot(a_i, x);
            x = x + lambda * residual * a_i / norm(a_i)^2;
        end

        % Global convergence check
        if norm(x - x_old) < global_tol
            disp(['Converged at iteration ', num2str(iter)]);
            break;
        end
        if iter == max_iter
            disp('Maximum iterations reached');
        end
    end
end
```