Image Reconstruction of Computed Tomography

SSY-186 Diagnostic Imaging

Project Report

Qixun Qu [redacted]

Yankun Xu [redacted]

Zihui Wang [redacted]

2017/05/05

Part A. Back Projection (BP)

A1. Write down the expression for reconstruction using back-projection and explain the meaning of each term in the equation.

In back projection method, the sinogram $g(l, \theta)$ is back projected to reconstruct the slice over each angle. In this case, there are 180 angles in total, and 144 detections are collected under each angle. As shown in Equation 1, $b_\theta(x, y)$ is the interpolated result from sinogram at one angle. To reconstruct the whole slice $f_b(x, y)$, all interpolations will be integrated over all angles as shown in Equation 2.

$$b_\theta(x, y) = g(x \cos\theta + y \sin\theta, \theta) \qquad \textit{Eq. 1}$$

$$f_b(x, y) = \int_0^\pi b_\theta(x, y)\, d\theta \qquad \textit{Eq. 2}$$

A2. Compute and plot the image from the back projection of the sinogram data at $\theta = 30°$.

Back projection of sinogram data at 30° is shown as Figure 1.

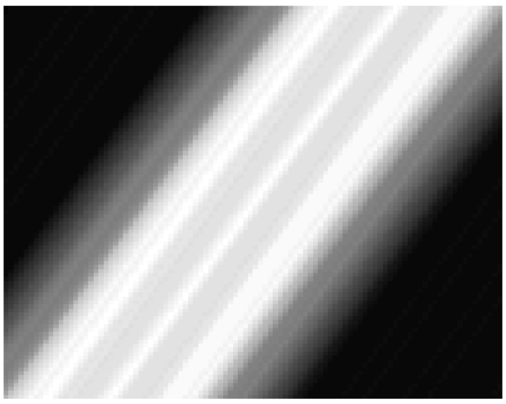
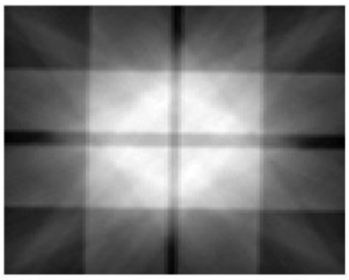


Figure 1. Back Projection of Sinogram Data at 30°

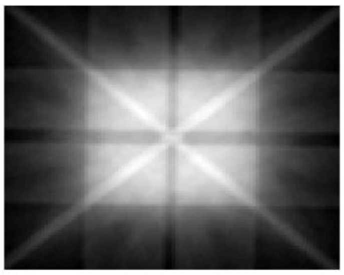
A3. Compute and plot the back projection summation image, with *N* = 6, 12, 30, 60 and 180, where *N* is the number of back projections. Multiply the resultant image by the factor of $\pi / (2 \times N)$ where *N* is the number of projections. Compare and comment on the results based on different values of *N*.

As shown in Figure 2, different back projection summation images are obtained based on varies number of projections. In general, more projections are applied to make reconstruction, clearer image that is displayed. There are too much artifacts in the results with *N* = 6, 12, 30 and 60. When *N* is equal to 180, by contrast, reconstructed images obviously have higher quality with fewer artifacts (though plenty of artifacts still exist).

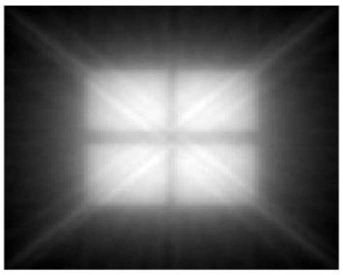
However, there is no doubt that it will take more time to reconstruct with more projections to ensure the quality of image.



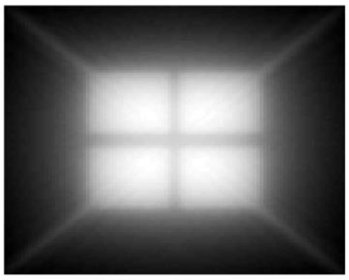
(a) N = 6



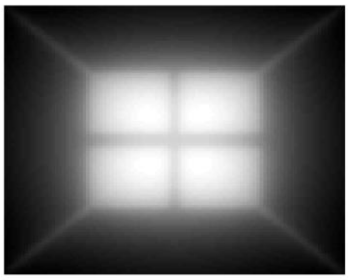
(b) N = 12



(c) N = 30



(d) N = 60



(e) N = 180

Figure 2. Back Projection Summation Image with Different N

B. Filtered Back Projection (FBP)

B1. Draw a flow chart to illustrate how to compute the filtered back projection from the sinogram data. List the relevant equations and explain how they are related to each other.

In contrast with BP, each projection of the sinogram is filtered by a high-pass filter in frequency domain before reconstructing projections in FBP process. Figure 3 shows the flow chart of filtering the projection. In first step, the 1D Fourier transform of each projection in $g(l,\theta)$ is computed to form $G(w,\theta)$ as Equation 3. After which, in frequency domain, $G(w,\theta)$ is multiplied by the high-pass filter H (i.e. $|w|$). (The Ram-Lak filter is used in Question B2. The Hamming window is applied in Question B3.) The inverse 1D Fourier transform is applied on $G(w,\theta)$ to generate a new sinogram $g'(l,\theta)$ to proceed back projection process. The calculation of new sinogram is displayed as Equation 4, and Equation 5 indicates the back projection with new sinogram.

$$G(w,\theta) = \int_{-\infty}^{+\infty} g(l,\theta) e^{-j2\pi wl} dl \qquad \textit{Eq. 3}$$

$$g'(l,\theta) = \int_{-\infty}^{+\infty} |w| G(w,\theta) e^{j2\pi wl} dw \qquad \textit{Eq. 4}$$

$$f_b(x,y) = \int_0^{\pi} g'(x\cos\theta + y\sin\theta, \theta)\, d\theta \qquad \textit{Eq. 5}$$

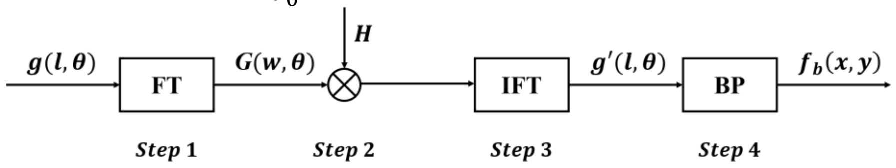
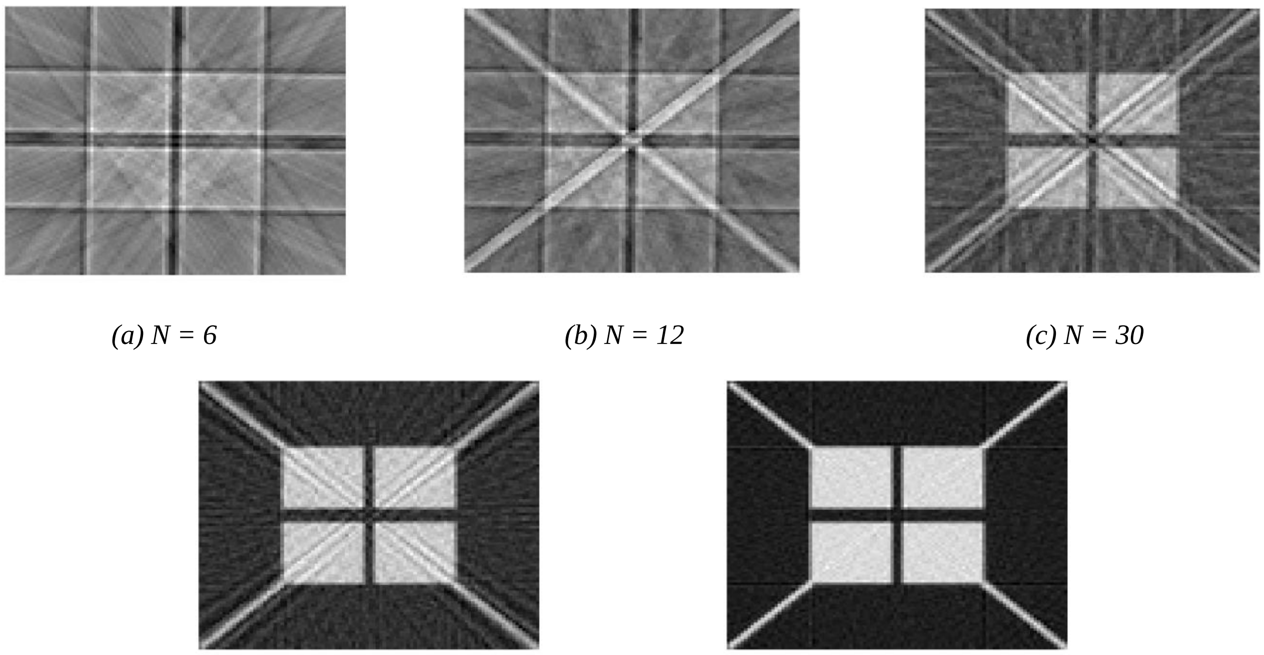


Figure 3. Flow Chart of the Process of Filtered Back Projection

B2. Compute and plot the filtered back projection, with *N* = 6, 12, 30, 60 and 180, where *N* is the number of back projections. Multiply the resultant image by the factor of $\pi / (2 \times N)$ where *N* is the number of projections. Compare and comment on the results based on different values of *N*. (Question: Which value of *N* should we use?)

In FBP, as shown in Figure 4, more projections are applied to reconstruct image, the higher-resolution image is obtained with fewer artifacts. When *N* equals to 180, almost all artifacts are removed. Thus, it's appropriate to set *N* as 180. Compared with the results of BP, images which reconstructed by FBP are much sharpener. Since the Ram-Lak filter is able to amplify the high-frequency part of projection, and at the same time the low-frequency part of projection is attenuated as shown in Figure 5. Thus, more details are provided from the filtered projection. But the disadvantage is that the noise is also enlarged.


(a) N = 6　*(b) N = 12*　*(c) N = 30*

(d) N = 60　*(e) N = 180*

Figure 4. Filtered Back Projection Summation Image with Different N and Ram-Lak Filter

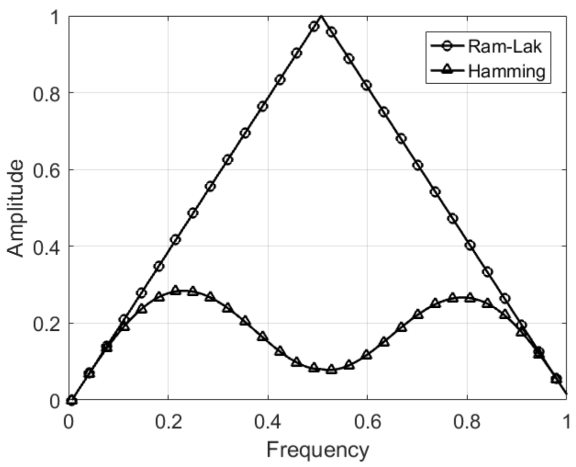
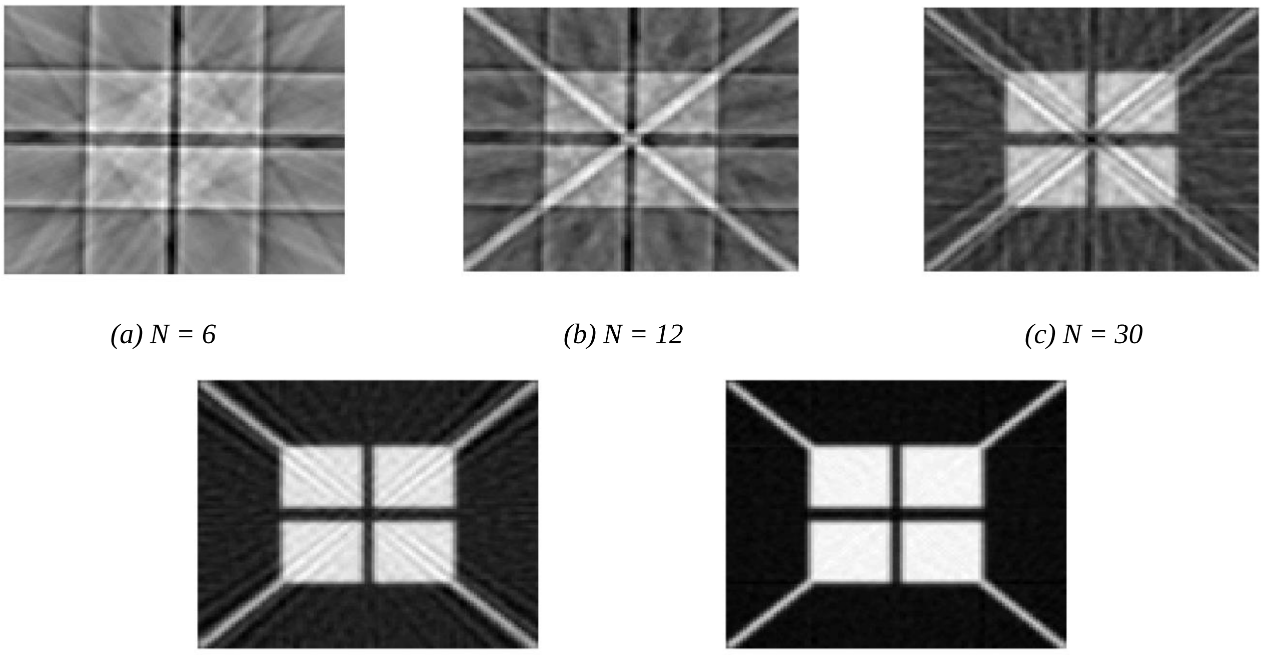


Figure 5. Ram-Lak Filter and Hamming Window

B3. Repeat B2 with a Hamming window included in the filtering process. Comment on the results as compared to those in Part B2.

When the Hamming window is utilized in FBP, the reconstruction results have fewer noise than the results in Part B2. The reason is that the high-frequency part of projection is also attenuated by the Hamming window as shown in Figure 5. Therefore, the images in this question is a bit fuzzier than images in former question.



(a) N = 6 *(b) N = 12* *(c) N = 30*

(d) N = 60 *(e) N = 180*

Figure 6. Filtered Back Projection Summation Image with Different N and Hamming Window

C. Convolution Back Projection (CBP)

C1. Explain the differences between filtered back projection and convolution back projection.

In FBP, the 1D Fourier transform of each projection in sinogram is multiplied by a high-pass filter in frequency domain, and compute the inverse 1D Fourier transform of the result to get new sinogram.

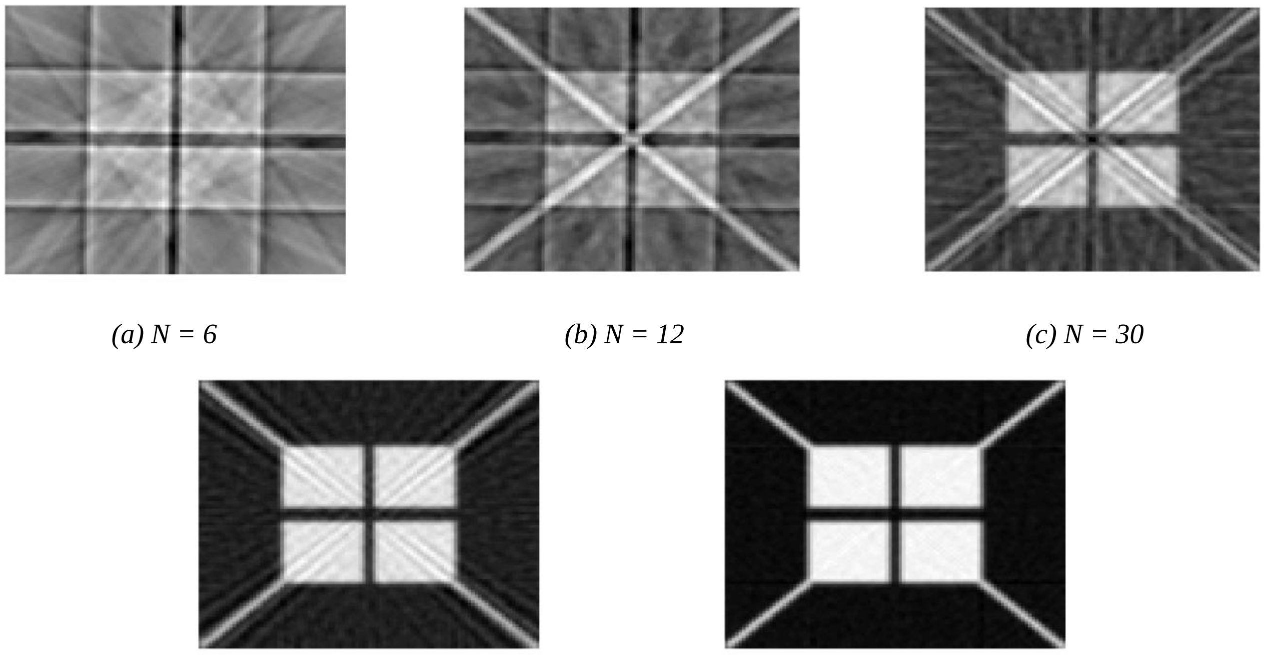
In CBP, the inverse Fourier transform of the high-pass filter is convoluted with projections in sinogram to generate the new sinogram in spatial domain.

C2. Implement the convolution back projection algorithm with a Hamming window. Comment on the results as compared to those in Part B3.

The reconstructed images are displayed in Figure 7. The results are equivalent with those in Part B3 but differ in computation method. Since the identical filter, Hamming window, is used to filter projections. The difference is that the projection is filtered in frequency domain in FBP, but in spatial domain in CBP.

C3. Compare and comment on the advantages and disadvantages of using FBP versus CBP.

Compared with FBP, CBP is more efficient. The convolution in spatial domain requires less computation than the implementation in frequency-domain by applying twice Fourier transform method. It is the reason that the CBP method is more useful than the FBP approach in the real solution. Also CBP is easier to reach since Fourier transform is more complex than convolution approach.



(a) N = 6 *(b) N = 12* *(c) N = 30*

(d) N = 60 *(e) N = 180*

Figure 7. Convolution Back Projection Summation Image with Different N and Hamming Window

D. Fan-Beam Projections

D1. What is fan-beam projection and fan-beam reconstruction? How many different geometries are there for fan-beam projection? Describe each of these.

Compared to back projection methods mentioned above which use parallel beam to perform linear scan, fan-beam projection uses fan beam to cover image region with single X-ray source. After each projection, X-ray source will rotate the specific angle to generate next projection.

Fan-beam reconstruction means that image is reconstructed by fan-beam projections rather than parallel-ray projections. Usually, fan-beam CBP method is used to reconstruct images.

A general fan-beam geometry is shown in Figure 8, fan-beam projection $p(\gamma,\beta)$ is measured at each rotational position of the X-ray source , where γ is the angular position of a given detector and β is the angular position of X-ray source. There are three fan-beam geometries as follows: (1) those that have equal angles between the measured beam, (2) those that have equal space between detectors, (3) those that have equal angles and equal space. The third geometry can only be satisfied if the detectors are placed along a circular arc whose center is at the source.

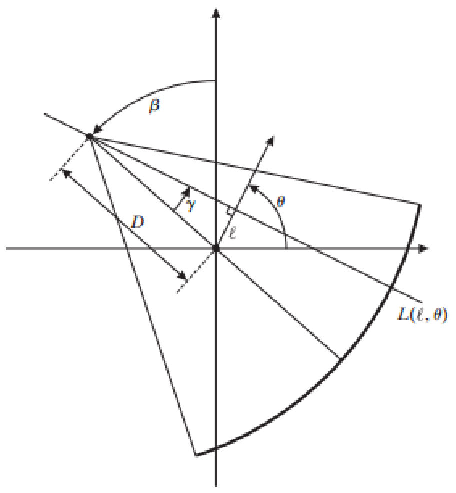


Figure 8. Basic Geometry of Fan Beam Projection

[Jerry L. Prince, Jonathan M. Links, "*Medical Imaging - Signals and Systems 2nd Edition*", 2014, p.209.]

D2. Can we directly apply the implemented algorithm(s) in part B and C to measurements from some of the fan-beam projection geometries described in D1? Explain why.

We cannot apply CBP or FBP algorithm discussed above directly to reconstruct image with fan-beam projections. The fan-beam reconstruction formula begins with the equation for parallel-ray back projection. In practice, we need take some mathematical tricks initially to transform fan-beam projection $p(\gamma, \beta)$ into the $L(l, \theta)$ in order to use back projection algorithm. Look back the Figure 8, l is the length of back projection point from object center, and D is the length of source from object center, θ is the same angle as in parallel-beam case. The transformation is computed as follows:

$$\theta = \beta + \gamma \qquad \text{Eq. 6} \qquad\qquad l = D \times sin(\gamma) \qquad \text{Eq. 7}$$

After generating the new sinogram $L(l, \theta)$ from detected data $p(\gamma, \beta)$, FBP or CBP approach can be applied to do reconstruction.

E. Real CT-image

E1. Reconstruct this image using the three reconstruction methods that you have implemented. You can do this for *N* = 180, otherwise use the settings for each algorithm that resulted in the best reconstruction.

As shown in Figure 9, three images reconstructed by BP, FBP and CBP respectively. 9(b) and 9(c) are generated with Hamming window.

E2. Compare these three images and comment on similarities or differences in the reconstruction results. What do you need to do to improve the reconstruction even further?

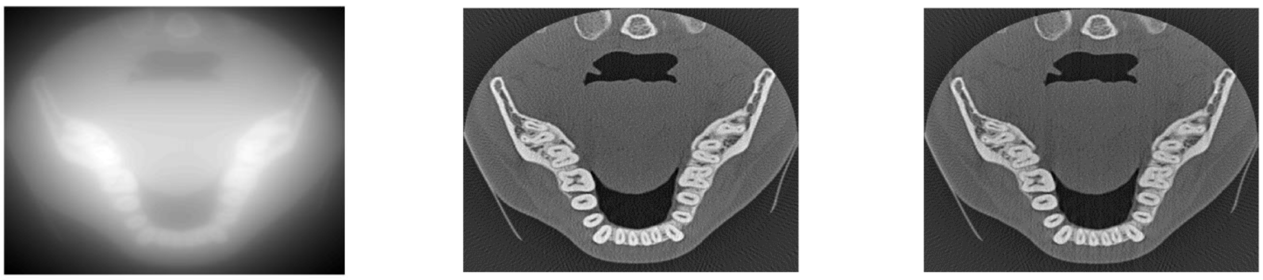
The image which is reconstructed by BP is much more blurring than images built by FBP and CBP. Results of FBP and CBP have high resolution that shows more details. To improve the reconstruction further, several methods can be approached as follows:

1. A better filter is ought to be implemented in back projection progress. The filter has a better performance of its frequency response, which attenuate more low-frequency part as well as high-frequency noise.

2. Apply iterative reconstruction. Take the result of FBP as example. After reconstructing image by FBP, the sinogram is projected from the result. The generated sinogram has the same dimension as the original one. After which, compute the difference between the generated sinogram and original sinogram, and correct the back projection result with the difference. Repeat the correction process until the difference is lower than a given threshold. The output of iterative reconstruction has fewer noise with higher resolution.

3. Apply image processing algorithm to enhance the contrast of reconstructed image.

4. This way is only for the improvement of BP result. Compute the 2D Fourier transform of the reconstructed image, and it will be filtered in frequency domain. Inverse 2D Fourier transform is applied on filtered result to obtain the reconstructed image with higher resolution. However, twice 2D Fourier transforms take more time than FBP and CBP.



(a) BP *(b) FBP* *(c) CBP*

Figure 9. Real CT Image Reconstructed by BP, FBP and CBP

Code of Back Projection

```
function img_bp = back_projection(sg, N, angle)
%%BACK_PROJECTION Implement the back projection approach to reconstruct
%image from given sinogram.
%     Input argument:
%     - sg : the known sinogram
%     - N : the number of projections used in construction,
%              default is the number of columns of sinogram
%     - angle : back projection of one data in sinogram at given angle
%     Output:
%     - img_bp : the reconstruction result

% Set the default value for the number of
% projection that applied to reconstruct
if nargin < 2 || isempty(N)
    N = size(sg, 2);
end

% Set the default value of angle
if nargin < 3 || isempty(angle)
    angle = -1;
else
    N = length(angle);
end

% Obtain the size of sinogram and compute the
% size of reconstructed image
[gl, gt] = size(sg);
hfgl = floor(gl / 2);

iw = 2 * floor(gl / (2 * sqrt(2)));
hfiw = iw / 2;

% Back Projection
% Initialize the reconstructed image
img_bp = zeros(iw);

% Compute some arguments for back projection
% Positions map of reconstructed image
[posX, posY] = meshgrid((1:iw) - hfiw);

if angle == -1
    % If back projection is generated from
    % several data in sinogram
    % Calculate the degree interval
    igt = floor(gt / N);
    angles_array = 1:igt:gt;
else
    % If back projection is created only by
    % one data in sinogram at given angle
    angles_array = angle;
end

% Run N times back projection
for t = angles_array

    % Calculate the position in sinogram
    pos = posX * cosd(t) + posY * sind(t) + hfgl;
    % Accumulate projection of each degree sinogram
    img_bp = img_bp + interp1(1:gl, sg(:, t), pos);
end

% Multiply the factor
img_bp = img_bp * (pi / (2 * N));

% Plot results
% Plot back projection result that
% multiplies the factor
figure
imagesc(img_bp), colormap gray
axis('off')

end
```

Code of Filtered Back Projection

```
function img_fbp = filtered_back_projection(sg, N, filter, angle)
%%FILTERED_BACK_PROJECTION Implement the filtered back projection approach
%to reconstruct image from given sinogram.
%     Input argument:
%     - sg : the known sinogram
%     - filter : the name of the filter used to filter the projection,
%         default is 'hamming'
%     - N : the number of projections used in construction,
%              default is the number of columns of sinogram
%     - angle : back projection of one data in sinogram at given angle
%     Output:
%     - img_bp : the reconstruction result

% Set the default value for the number of
% projection that applied to reconstruct
if nargin < 2 || isempty(N)
    N = size(sg, 2);
end

% Set the default value of filter
if nargin < 3 || isempty(filter)
    filter = 'hamming';
end

% Set the default value of angle
if nargin < 4 || isempty(angle)
    angle = -1;
else
    N = length(angle);
end

% Obtain the size of sinogram and compute the
% size of reconstructed image
[gl, gt] = size(sg);
hfgl = floor(gl / 2);

iw = 2 * floor(gl / (2 * sqrt(2)));
hfiw = iw / 2;

% Filtered Back Projection
% Compute the Ramlak filter
gx = [0:hfgl, hfgl - 1:-1:1];
```

```
if mod(gl, 2) ~= 0
        gx = [gx, 0];
end

ramlak = 2 * gx / gl;

switch filter

        case 'ramlak' % Use Ramlak filter
                H = ramlak;
        case 'hamming' % Use Hamming filter
                hamming = 0.54 - 0.46 * cos(2 * pi * (0:gl-1) / gl);
                H = [hamming(hfgl:gl), hamming(1:hfgl-1)] .*
ramlak;
        otherwise
                fprintf('Wrong filter, it should be "ramlak" or
"hamming".')

end

% Compute Fourier transformation of sinogram
gf = fft(sg, [], 1);
% Multiply the filter in frequency domain
gff = bsxfun(@times, gf, H');
% Do inverse Fourier transformation
gffi = real(ifft(gff, [], 1));

% Initialize the reconstructed image
img_fbp = zeros(iw);

% Compute some arguments for back projection
% Positions map of reconstructed image
[posX, posY] = meshgrid((1:iw) - hfiw);

if angle == -1
        % If back projection is generated from
        % several data in sinogram
        % Calculate the degree interval
        igt = floor(gt / N);
        angles_array = 1:igt:gt;
else
        % If back projection is created only by
        % one data in sinogram at given angle
        angles_array = angle;
end

% Run N times back projection
for t = angles_array

        % Calculate the position in sinogram
        pos = posX * cosd(t) + posY * sind(t) + hfgl;
        % Accumulate projection of each degree sinogram
        img_fbp = img_fbp + interp1(1:gl, gffi(:, t), pos);

end

% Multiply the factor
img_fbp = img_fbp * (pi / (2 * N));

% Plot results
% Plot back projection result
figure
imagesc(img_fbp), colormap gray
axis('off')

end
```

Code of Convolution Back Projection

```
function img_cbp = convolution_back_projection(sg, N,
filter, angle)
%%CONVOLUTION_BACK_PROJECTION Implement
the convolution back projection
%approach to reconstruct image from given sinogram.
%    Input argument:
%    - sg : the known sinogram
%    - filter : the name of the filter used to filter the
projection,
%        default is 'hamming'
%    - N : the number of projections used in construction,
%              default is the number of columns of sinogram
%    - angle : back projection of one data in sinogram at
given angle
%    Output:
%    - img_bp : the reconstruction result

% Set the default value for the number of
% projection that applied to reconstruct
if nargin < 2 || isempty(N)
        N = size(sg, 2);
end

% Set the default value of filter
if nargin < 3 || isempty(filter)
        filter = 'hamming';
end

% Set the default value of angle
if nargin < 4 || isempty(angle)
        angle = -1;
else
        N = length(angle);
end

% Obtain the size of sinogram and compute the
% size of reconstructed image
[gl, gt] = size(sg);
hfgl = floor(gl / 2);

iw = 2 * floor(gl / (2 * sqrt(2)));
hfiw = iw / 2;

% Convolution Back Projection
% Compute the Ram-Lak filter
gx = [0:hfgl, hfgl - 1:-1:1];
if mod(gl, 2) ~= 0
        gx = [gx, 0];
end

ramlak = 2 * gx / gl;

switch filter

        case 'ramlak' % Use Ramlak filter
                H = ramlak;
        case 'hamming' % Use Hamming filter
```

```
                hamming = 0.54 - 0.46 * cos(2 * pi * (0:gl-1) / gl);
                H  =  [hamming(hfgl:gl),  hamming(1:hfgl-1)]  .*
ramlak;
        otherwise
                fprintf('Wrong  filter,  it  should  be  "ramlak"  or
"hamming".')

end

% Compute inverse Fourier transformation of filter
c = real(ifftshift(ifft(H)));
% Convolve the sinogram with the IFT of filter
sgc = convn(sg, c', 'same');

% Initialize the reconstructed image
img_cbp = zeros(iw);

% Compute some arguments for back projection
% Positions map of reconstructed image
[posX, posY] = meshgrid((1:iw) - hfiw);

if angle == -1
        % If back projection is generated from
        % several data in sinogram
        % Calculate the degree interval
        igt = floor(gt / N);
        angles_array = 1:igt:gt;
else
        % If back projection is created only by
        % one data in sinogram at given angle
        angles_array = angle;
end

% Run N times back projection
for t = angles_array

        % Calculate the position in sinogram
        pos = posX * cosd(t) + posY * sind(t) + hfgl;
        % Accumulate projection of each degree sinogram
        img_cbp = img_cbp + interp1(1:gl, sgc(:, t), pos);

end

% Multiply the factor
img_cbp = img_cbp * (pi / (2 * N));

% Plot results
% Plot back projection result
figure
imagesc(img_cbp), colormap gray
axis('off')

end
```