Expectation Maximization Algorithim for PET image Reconstruction

Table of Contents

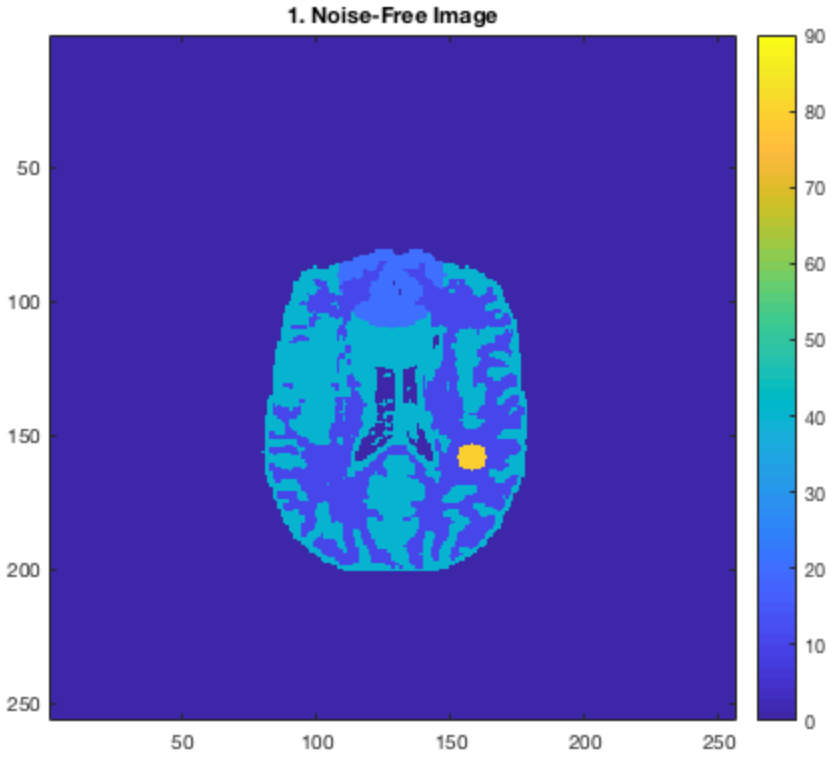
Author: Zachary Nelson. Date: November 11, 2016. Description: This program performs the Expectation Maximization Algorithim for imgage reconstruction given the output signal from a PET machine (brain_image.mat) and the PET system geometry (P.mat).

```
clear; clc; close all;
```

Load the virtual PET system matrix and brain image

```
disp('1. Loading the virtual PET system and brain image')

% load the PET system matrix
imgsiz = [256 256]; % define image grid
prjsiz = [256 192]; % define sinogram size: 256 radial bins, 192
 angular bins
load('P'); % 2D PET sparse system matrix, xy projection

load('brain_image'); % load machine True image.

% display the True brain image
figure(1)
imagesc(x0,[0 90]);
axis image; title('1. Noise-Free Image'); colorbar;
```

```
1. Loading the virtual PET system and brain image
```

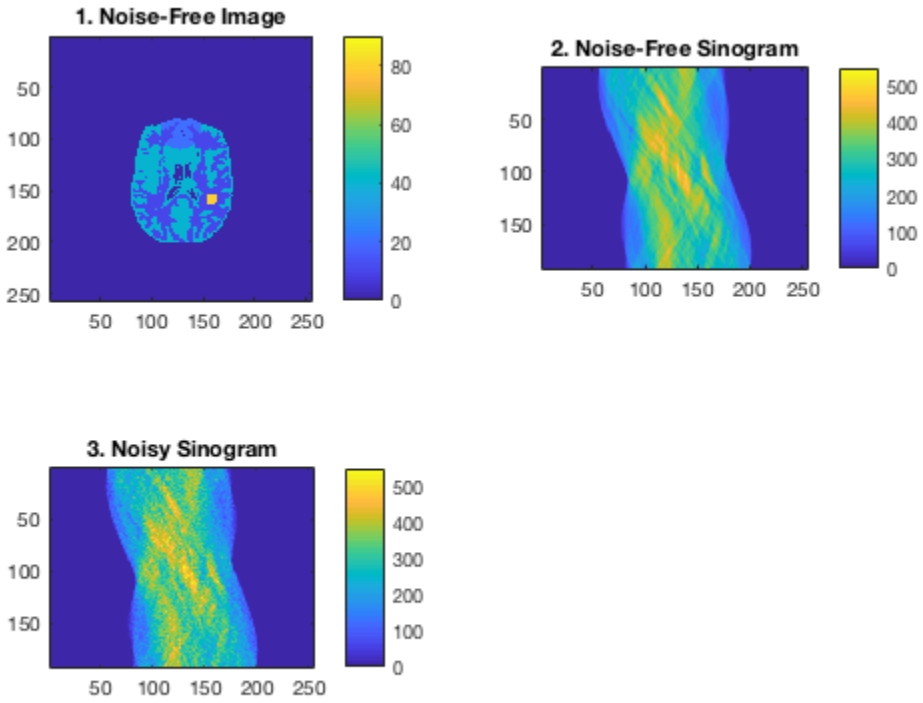


Simulate a PET sinogram by forward projection

```
disp('2. Simulated PET sinogram data')

% noise-free sinogram
y0 = P * x0(:);          % forward projection

y0 = reshape(y0,prjsiz); % reshape it to a matrix

% display the True brain image
figure(2)
subplot(221), imagesc(x0,[0 90]);
axis image; title('1. Noise-Free Image'); colorbar;

% display the noise-free sinogram
subplot(222), imagesc(imrotate(y0,-90),[0 550]);
axis image; title('2. Noise-Free Sinogram'); colorbar;

% Create sinogram with noise.
count = 2000e3;             % 2000K coincidences (photon detection
 error)
cs = count / sum(y0(:));    % scale coincidences
yi = poissrnd(y0*cs)/cs;    % include Poisson noise distribution

% display the nois sinogram
```

```
subplot(223), imagesc(imrotate(yi,-90),[0 550]);
axis image; title('3. Noisy Sinogram'); colorbar;
```

2. Simulated PET sinogram data

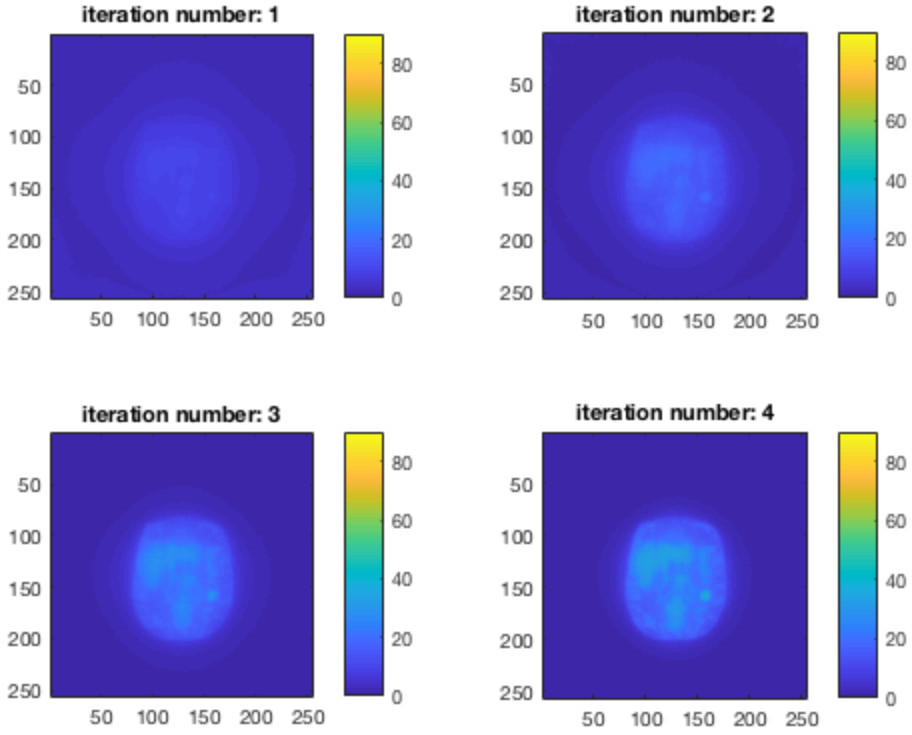


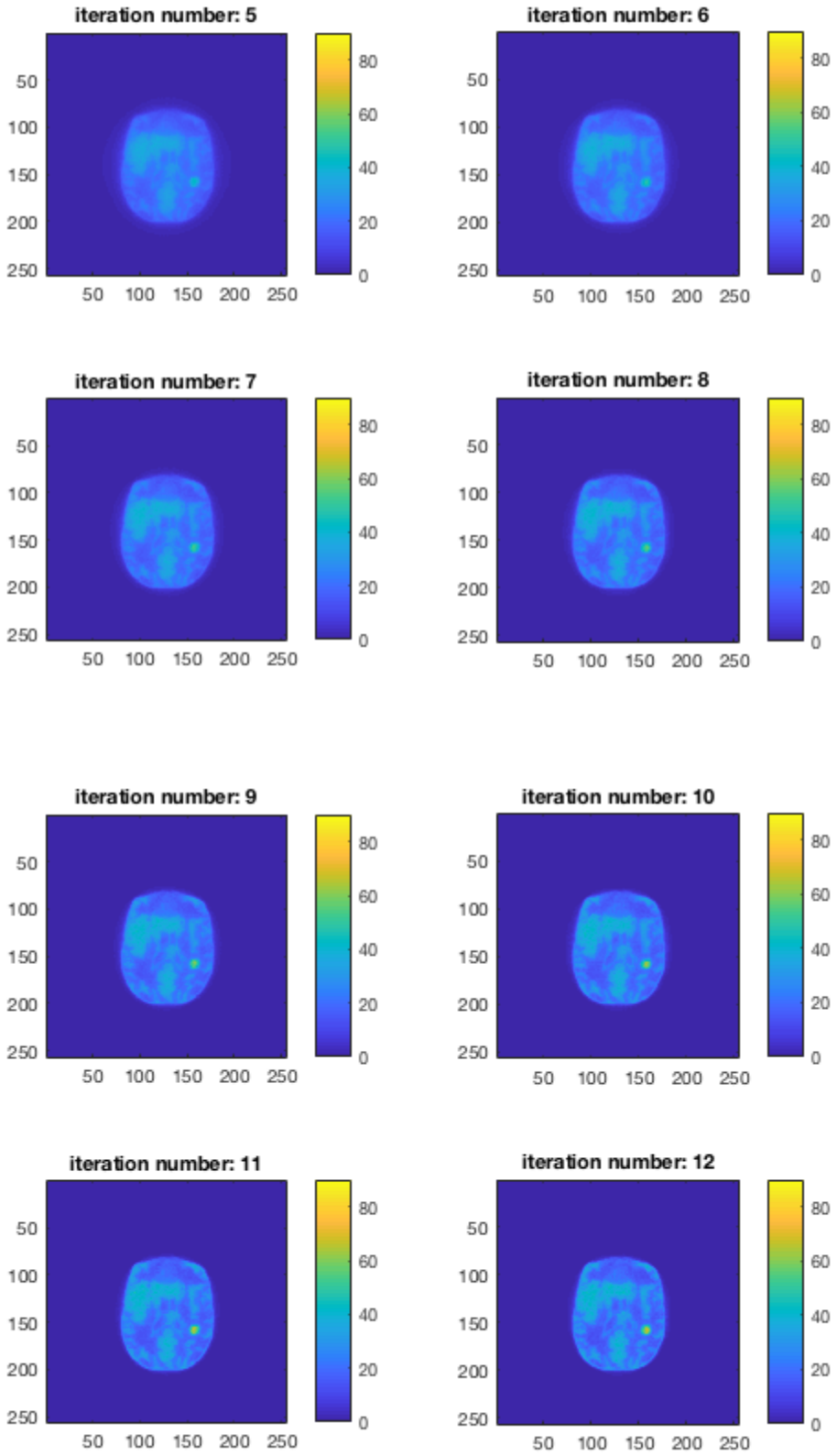
maximum likelihood (ML) expectation maximization (EM) reconstruction

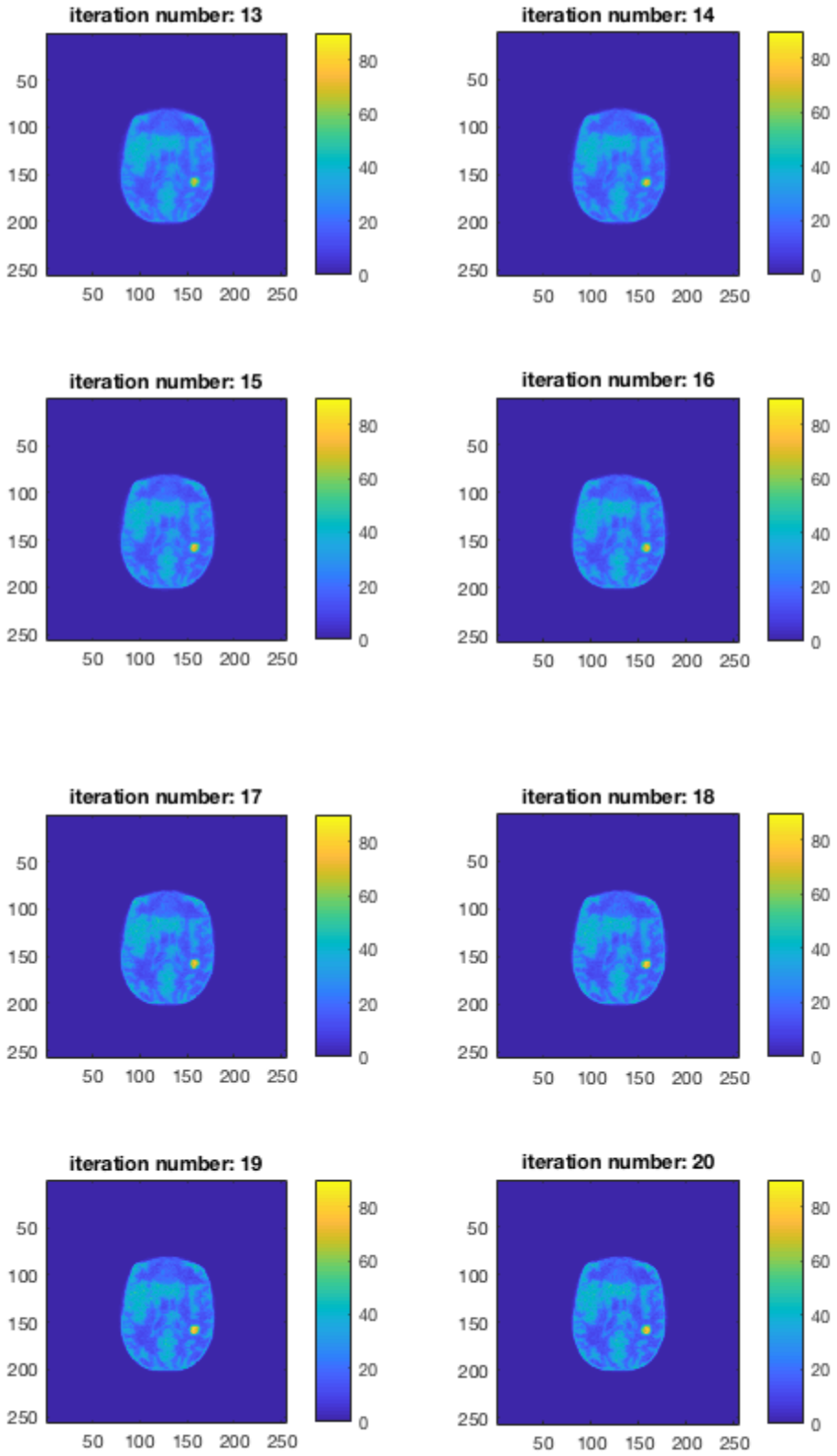
```
disp('3. PET image reconstruction using the ML-EM algorithm')

% establish maximum iteration number
maxit = 40;

% pre-calculate the sensitivity image (backprojection of a uniform
% sinogram)

% algorithm
y1 = ones(prjsiz);
s  = P' * y1(:);

% initial guess, uniform image
xinit = ones(imgsiz);

% iterate
x = xinit(:);
```

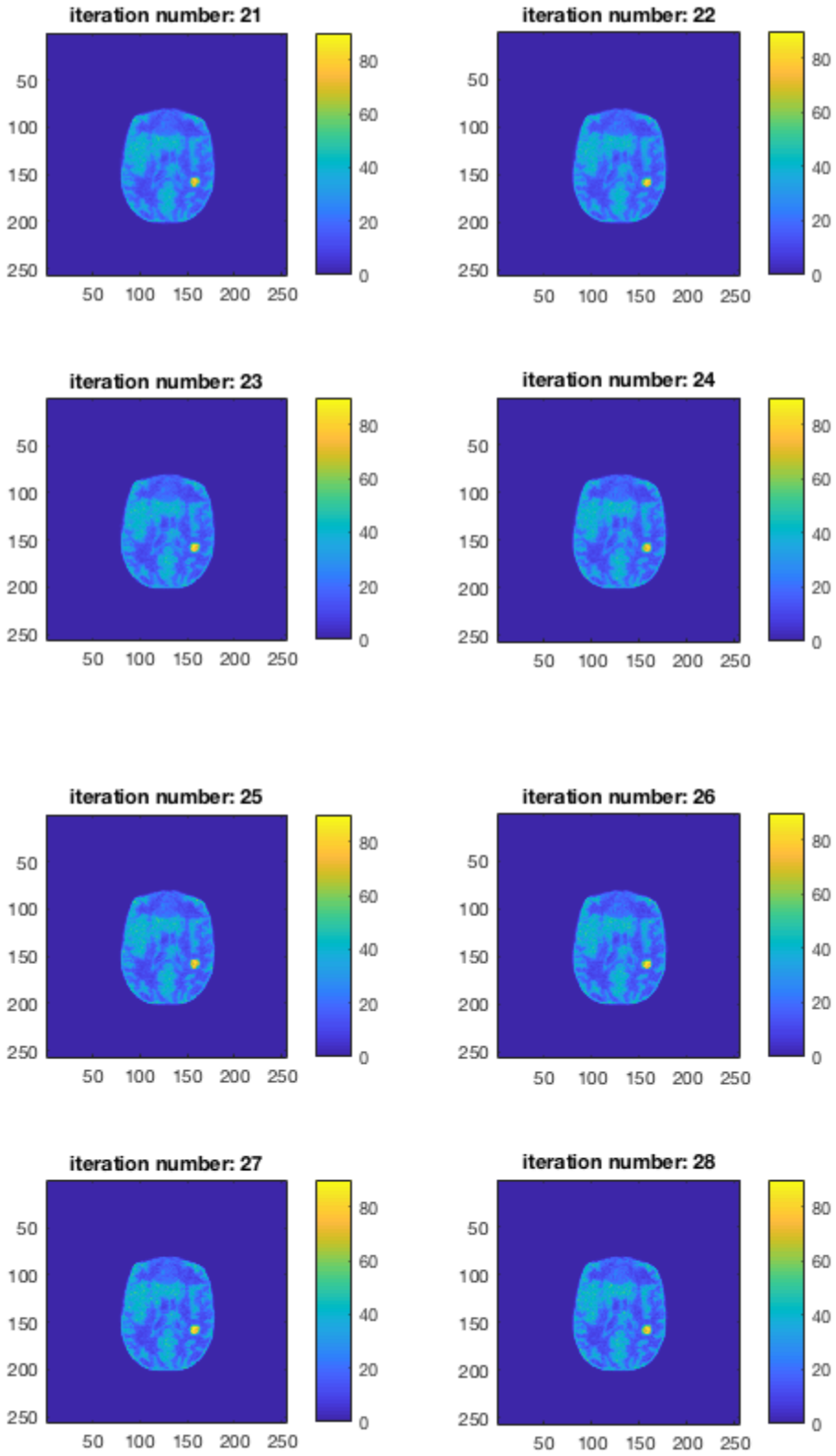
```
k = 10;
kk=1;
for it = 1:maxit

    % display current iteration number
    if rem(it,10)==0
        disp(sprintf('ML EM iteration %d',it));
    end

    % forward projection
    yb = P * x;

    % ratio sinogram
    yr = yi(:)./yb;
    yr(yb==0) = 0;
    yr(yi(:)==0&yb==0) = 1; % deal with zeros

    % backprojection
    xb = P' * yr;

    % new image estimate
    x = x./s .* xb;

    % save image estimate
    X(:,it) = x(:);

    % calculate the Poisson log likelihood
    L(it) = sum(yi(:).*log(yb) - yb);

    %Display Algorithim changes
    if mod(it-1,4)==0
        k=k+1;
        kk=1;
    end
    figure(k);
    subplot(2,2,kk);
    imagesc(reshape(X(:,it),imgsiz),[0 90]); axis image;
    axis image; title(sprintf('iteration number: %d',it)); colorbar;
    kk=kk+1;
end


% display the EM image estimate
figure(2)
subplot(224), imagesc(reshape(X(:,40),imgsiz),[0 90]); axis image;
axis image; title('4. Reconstructed Image'); colorbar;
%individual figure
figure(4)
imagesc(reshape(X(:,40),imgsiz),[0 90]); axis image;
axis image; title('4. Reconstructed Image'); colorbar;

figure(1)
imagesc(x0,[0 90]);
```

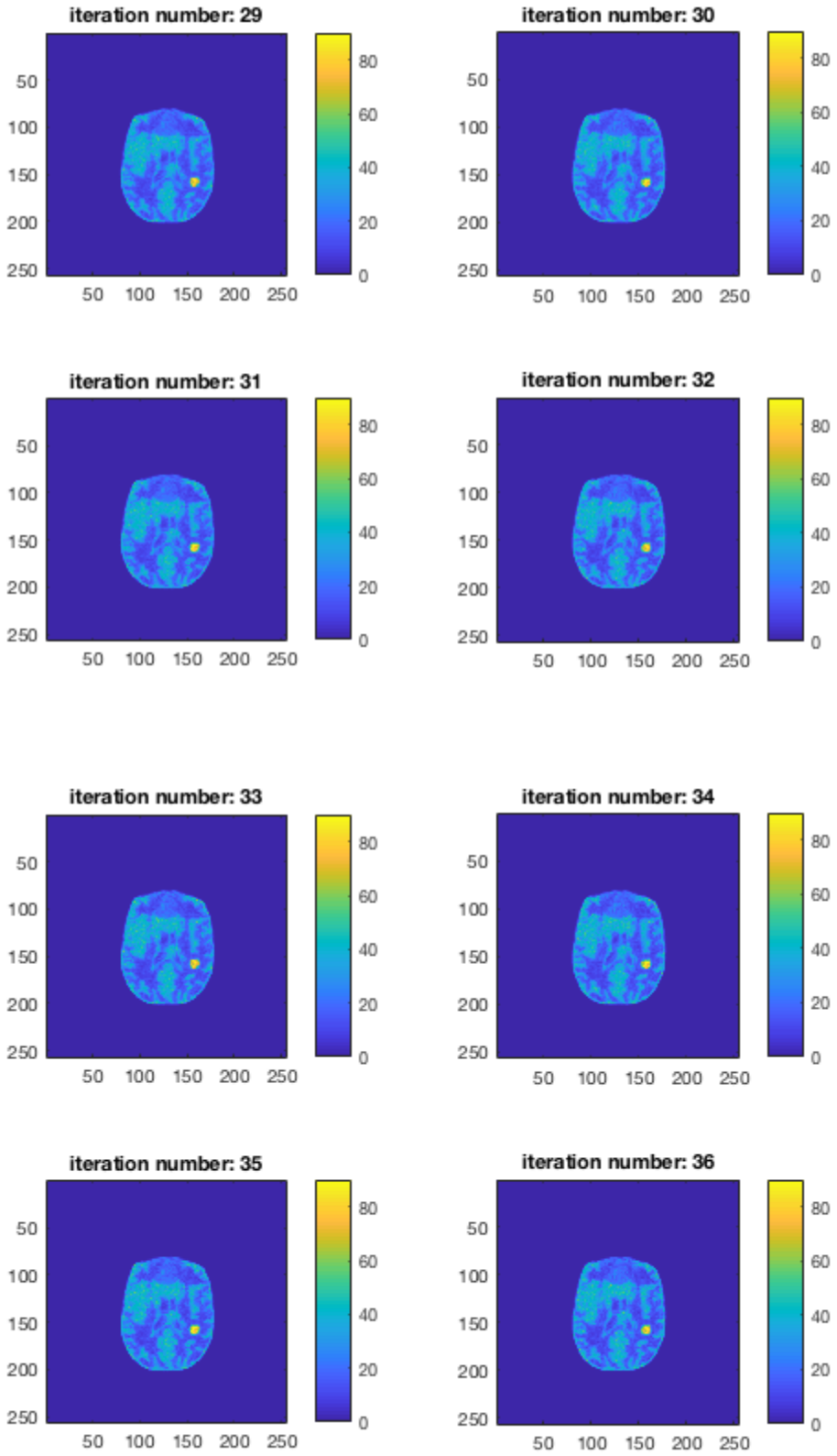
```
axis image; title('1. Noise-Free Image'); colorbar;

3. PET image reconstruction using the ML-EM algorithm
ML EM iteration 10
ML EM iteration 20
ML EM iteration 30
ML EM iteration 40
```

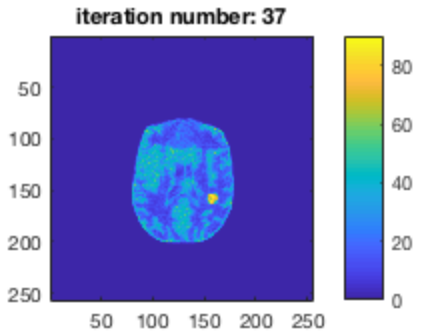


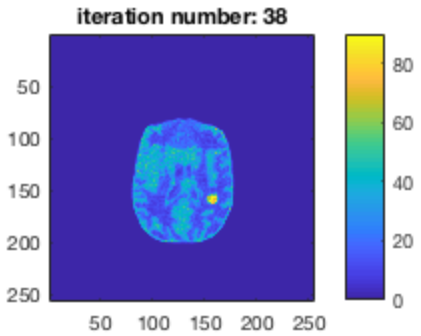

iteration number: 5
iteration number: 6
iteration number: 7
iteration number: 8
iteration number: 9
iteration number: 10
iteration number: 11
iteration number: 12
50
100
150
200
250
0
20
40
60
80

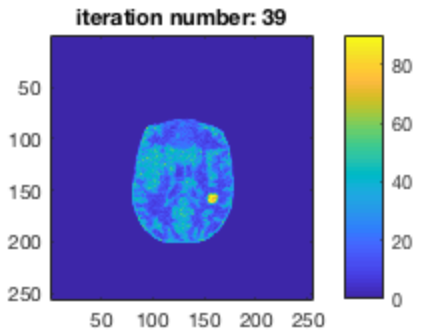

iteration number: 13
iteration number: 14
iteration number: 15
iteration number: 16
iteration number: 17
iteration number: 18
iteration number: 19
iteration number: 20
50
100
150
200
250
0
20
40
60
80

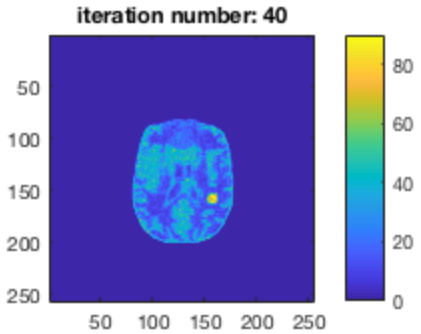

iteration number: 21
iteration number: 22
iteration number: 23
iteration number: 24
iteration number: 25
iteration number: 26
iteration number: 27
iteration number: 28
50
100
150
200
250
0
20
40
60
80

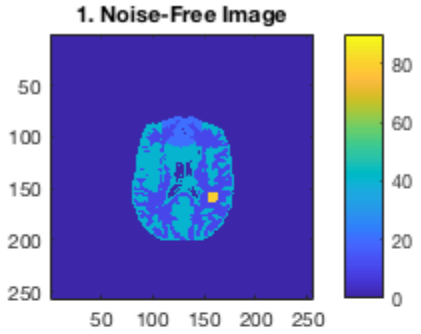

iteration number: 29
iteration number: 30
iteration number: 31
iteration number: 32
iteration number: 33
iteration number: 34
iteration number: 35
iteration number: 36
50
100
150
200
250
0
20
40
60
80

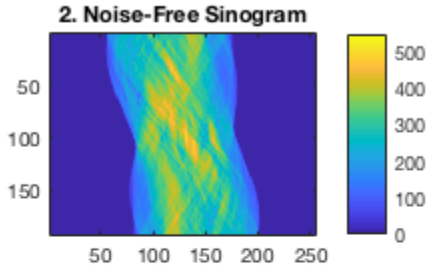

iteration number: 37
50
100
150
200
250
50 100 150 200 250
80
60
40
20
0

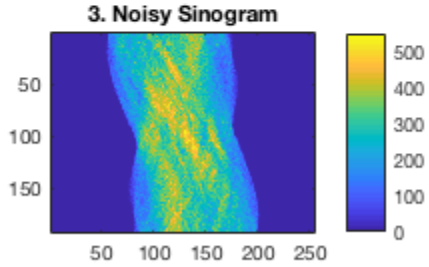

iteration number: 38
50
100
150
200
250
50 100 150 200 250
80
60
40
20
0

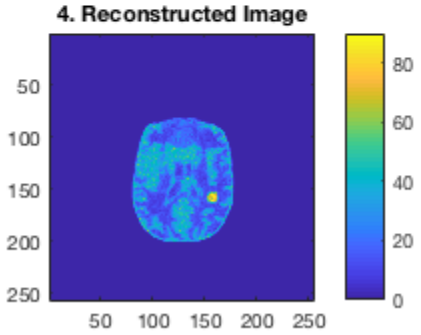

iteration number: 39
50
100
150
200
250
50 100 150 200 250
80
60
40
20
0

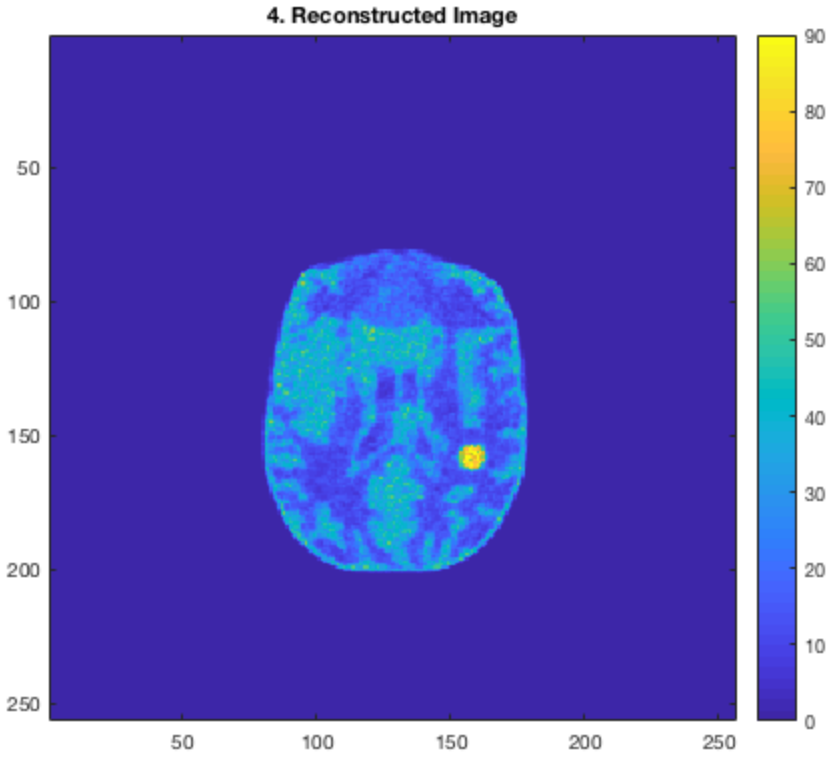

iteration number: 40
50
100
150
200
250
50 100 150 200 250
80
60
40
20
0

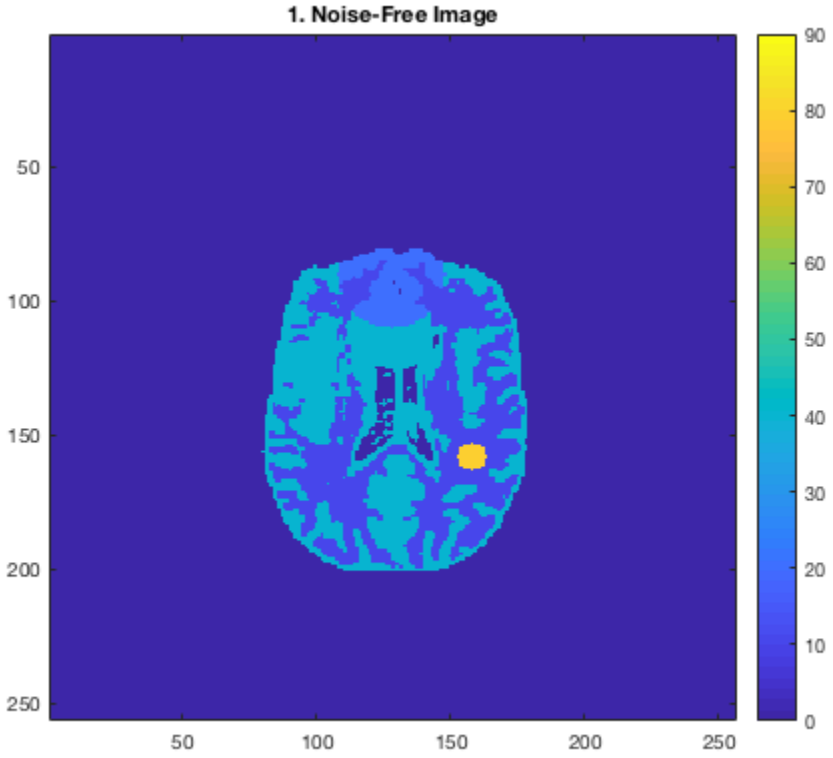

1. Noise-Free Image
50
100
150
200
250
50 100 150 200 250
80
60
40
20
0


2. Noise-Free Sinogram
50
100
150
50 100 150 200 250
500
400
300
200
100
0


3. Noisy Sinogram
50
100
150
50 100 150 200 250
500
400
300
200
100
0


4. Reconstructed Image
50
100
150
200
250
50 100 150 200 250
80
60
40
20
0

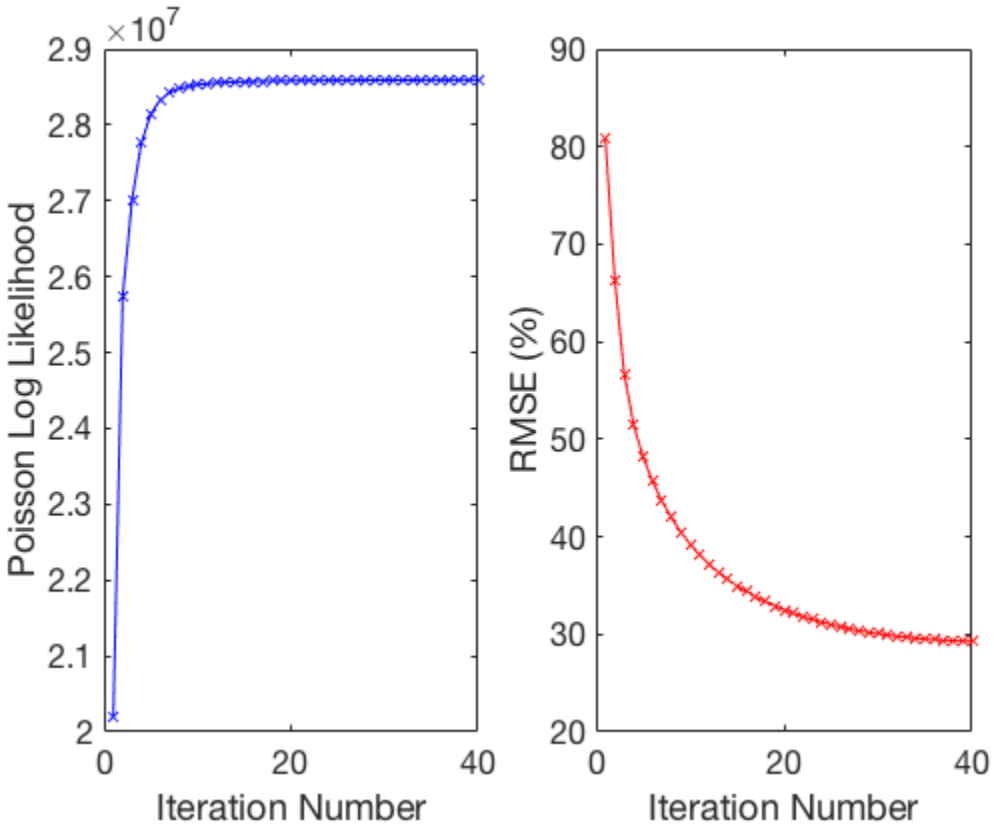

4. Reconstructed Image
50
100
150
200
250
50
100
150
200
250
90
80
70
60
50
40
30
20
10
0


1. Noise-Free Image
50
100
150
200
250
50
100
150
200
250
90
80
70
60
50
40
30
20
10
0

Evaluation of likelihood and reconstruction error

```
disp('4. Evaluation of Poisson log likelihood and reconstruction
 error')

% Assess reconstruction error of EM iterates using RMSE
for it = 1:maxit
    rmse = sqrt(mean((X(:,it)-x0(:)).^2)); % absolute RMSE
    RMSE(it) = rmse / sqrt(mean((x0(:)).^2)) * 100; % percent RMSE
end

% plot of the Poisson log likelihood as a function of iterations
ii = 1:maxit;
figure(5)
subplot(121), plot(ii,L,'b-x');
set(gca,'FontSize',16);
xlabel('Iteration Number'); ylabel('Poisson Log Likelihood');

% plot of RMSE as a function of iterations
subplot(122), plot(ii,RMSE,'r-x');
set(gca,'FontSize',16);
xlabel('Iteration Number'); ylabel('RMSE (%)');
```

4. Evaluation of Poisson log likelihood and reconstruction error



Published with MATLAB® R2017a