EEN180 - Medical Imaging Systems
Lab on image reconstruction for Positron Emission Tomography

Oscar Jalnefjord, Jennifer Alvén and Ida Häggström

Due: January 3, 2025

Modern medical imaging is generally using iterative reconstruction methods. These methods use the forward model of the imaging process (object to projection, object to k-space etc.) instead of the opposite that is used by analytical reconstruction methods. The forward model is often easier to describe and better known. Iterative reconstruction methods are typically characterized by a reduced noise sensitivity and enables straight-forward ways of including features like prior knowledge or sparsity constraints.

The objective of the project is to reconstruct PET images from sinogram data using iterative reconstruction. These instructions assume that MATLAB is used as the programming environment, but Python is also allowed.

Notes

- The project has a total of 10 pts. To pass, you need ≥ 5 pts.
- This is a group exercise. Discussions between groups are encouraged while copying of code among groups is strictly prohibited. Any evidence of plagiarism will be dealt with according to Chalmers regulations by the teaching team.
- The objective of this assignment is algorithm implementation. You are allowed to use built-in MATLAB functions, but specific functions that are directly related to the assignment such as open-source code for iterative PET/SPECT reconstruction cannot be used, except for your own benchmark of your code.

Theory

The reconstruction method used in this project builds on the principle that the forward projection (object to sinogram) can be described as a linear process $g = Af$ (for each 2D slice). The vector g equals a flattened version of the

sinogram matrix G while the vector f equals a flattened version of the image matrix F.

The matrix A is often called the *system matrix* and is a transformation matrix that transforms image pixel values to projection (sinogram) data. The elements of A, a_{ij}, are thus sensitivity factors that characterize the sensitivity of each voxel to the collection of projection measurements: a_{ij} represents the sensitivity (probability) that an annihilation event in pixel j is measured in line i (line of response, LOR). Many different effects contribute to these probabilities; geometry, object attenuation, individual detector efficiencies, etc. In this project, we only consider the geometry effect on the elements of A. Each pixel in the object (image) can contribute to a count in a certain LOR, with different probabilities determined by the scanner geometry relative to the object. The total count in a LOR i is then the total contribution from all m pixels:

$$g_i = \sum_{j=1}^{m} a_{ij} f_j$$

Thus, to calculate A we need to determine the contribution of each pixel in the image to all LORs in the sinogram, i.e. how much of a specific pixel in the image "ends up" in every single possible LOR.

Data

About the sinogram data

Type `load('PETdata.mat')` to load the data needed for the project. The sinogram data is stored in `G` (noise free) and `G_noisy` (with noise). The size of the sinogram is 87 rows $\times$ 180 columns $\times$ 26 slices. The sinogram was formed based on multiple parallel beam projections of an object from 180 different directions: 0° to 179° with a step size of 1°.

About the reconstructed images

Unlike other medical imaging modalities, nuclear medicine images typically have an inverted gray scale. This means that the images should be viewed such that the small pixel values are light (white) and large pixel values are dark (black), unless otherwise stated. This can easily be done in MATLAB or 3D slicer.

In this project, the size of the first dimension of the sinogram equals the size of the sides of the (square) images that are to be reconstructed, i.e. the size of the output images should be 87 $\times$ 87 $\times$ 26 voxels, where 26 represents the number of slices. Note that this is different from the CT project where the size of the sinogram was larger than the reconstructed image. All reconstructions should be made slice by slice.

Tasks

Using the given sinogram data together with the course material (P&L 9.2.3, Chappell 10.2) and/or the suggested reading, complete the following tasks:

Part A. Forward projection

We know that the forward projection (object to sinogram) can be described as a linear process $g = Af$ using the flattened $G \to g$ and flattened $F \to f$ of each 2D slice. The size of the sinogram matrix G equals $L \times \Theta$ and the size of the image matrix F equals $R \times C$. In this project, we have that $L = R = C$ and that $\Theta = 180$. To reconstruct an image, we need to find the matrix A, and then solve for f. See P&L 9.2.3 for more details.

A1. **(0.5 pts)** What are the sizes of g and f (in unit pixels)? What should the size of A be?

A2. **(1.5 pts)** Write a function that returns the matrix A given the projection angles, the size of the sinogram and the dimensions of the image:

```
function A = compute_forward_matrix(thetas, L, R, C)
    A = zeros(L*length(thetas), R*C)
    for i = 1:length(thetas)
        for j = 1:R*C
            A(1+(i-1)*L:i*L, j) = ...
        end
    end
end
```

Hint: You will find the MATLAB function `radon.m` *helpful. Hint 2: Think about how MATLAB indexes array elements versus the angle in degrees.*

A3. **(1.0 pts)** Write a function that performs the naive reconstruction by solving the problem $g = Af$, i.e. solving for f:

```
function F = NAIVE(A, G, output_size)
    F = zeros(output_size);
    for slice = 1:output_size(3)
        F(:, :, slice) = ...
    end
end
```

Apply your function to the noise free sinogram data to verify that the A you computed in A2 is correct. *Note: This reconstruction is very slow, it is sufficient to reconstruct a single slice (e.g. slice 10). Make sure to save the slice for later use in task C1.* **To include in the report:** A reconstructed slice of the noise-free data.

Part B. Iterative reconstruction

The reconstruction method used in modern PET is based on maximum-likelihood expectation maximization (MLEM)[1]. MLEM is a statistical method used to estimate unknown parameters, in this case the image, based on measured data and a probability distribution describing the measurement noise.

B1. **(1.0 pts)** For PET, the noise distribution is given by the Poisson distribution since the detector counts are related to radioactive decay, i.e.

$$P(g_i) = e^{-\bar{g}_i} \frac{\bar{g}_i^{g_i}}{g_i!},$$

where g_i is the measured count for a given combination of position and angle, i, in the sinogram (i.e. a specific LOR) and where $\bar{g}_i$ is the average count for that LOR. Note that the index i refers to one element in the sinogram, that is, it indexes the flattened vector g. The MLEM algorithm seeks to find the image f that maximizes the logarithm of the Poisson likelihood objective function:

$$L(f) = P(g|f) = \prod_i P(g_i).$$

Derive the log-likelihood function based on the equations above. **To include in the report:** The derivations of the log-likelihood function.

B2. **(0.5 pts)** Iterative reconstruction schemes build on iteratively updating an estimate (guess) of the image to make it more and more aligned with the measured data. What is an appropriate start image $f^{(0)}$?

B3. **(1.5 pts)** It can be shown that the following ML-EM iteration converges to the ML solution:

$$\bar{g}^{(k)} = Af^{(k)},$$
$$f_j^{(k+1)} = \frac{f_j^{(k)}}{\sum_i a_{ij}} \sum_i a_{ij} \frac{g_i}{\bar{g}_i^{(k)}},$$

where k is the iteration number and the index j refers to a given pixel in the reconstructed image.

[1] For computation speed a method called ordered subset expectation maximization (OSEM) is used in practical implementations, but it is based on the same principle.

Write a function that uses this iterative scheme to reconstruct an image:

```
function [F, r_error] = MLEM(A, G, output_size, nbr_iter)
    F = ...
    r_error = zeros(nbr_iter+1, 1);
    for k = 1:nbr_iter
        r_error(k) = ...
        for slice = 1:output_size(3)
            F(:, :, slice) = ...
        end
    end
    r_error(nbr_iter+1) = ...
end
```

You should compute the sum-of-squared difference between the sinogram and the forward projection at each iteration. Apply you function to the noise free sinogram and use the sum-of-squared residuals to decide when the reconstruction algorithm has converged. *Note: Make sure to save your reconstructed image for later use in task C4.* **To include in the report:** A reconstructed slice for different iteration numbers and a plot of the sum-of-squared residuals vs iteration number.

Part C. Iterative reconstruction with noisy data

C1. **(0.5 pts)** Apply the naive reconstruction from task A3 to the noisy sinogram, and compare the reconstructed noisy image with the noise free version. **To include in the report:** A slice of the reconstructed noisy image and comments.

C2. **(1.0 pts)** Write a function that uses filtered back-projection (FBP) to reconstruct an image:

```
function F = FBP(G, theta, output_size)
    F = zeros(output_size);
    for slice = 1:output_size(3)
        F(:, :, slice) = ...
    end
end
```

Apply your function to the noisy and noise-free sinogram. *Hint: You will find the MATLAB function* `iradon.m` *helpful.* **To include in the report:** Slices of FBP reconstructed noisy and noise-free images and comments.

C3. **(1.0 pts)** Apply the MLEM reconstruction from task B3 to the noisy sinogram and view how the image evolve for each iteration up until e.g

100 iterations. **To include in the report:** A reconstructed slice for different iteration numbers with comments. A discussion of how your observations can be explained in terms of *overfitting*.

C4. **(1.0 pts)** Fine-tuning of the optimal number of iterations is clearly necessary. For this project we have the noise-free images available meaning that we can see how the reconstructed noisy image deviates from the true image at each iteration. Monitor the sum-of-squared difference between the noisy reconstructed image and the noise-free image reconstructed in B3 at each iteration. Select an optimal number of iterations. **To include in the report:** A slice of the noisy image obtained from the iterative reconstruction, and comments on the difference to the corresponding slice obtained from FBP in task C2.

C5. **(0.5 pts)** Discuss why the sum-of-squared difference between the sinogram and the forward projection (the error in B3) cannot be used to find an optimal number of iterations given noisy data.

Submission

Each group has to submit (via canvas *only*) either 1 or 2:

1. Preferably a MATLAB live script (.mlx file format) or Python notebook (.ipynb) instead of separate code and reports. The script has to be well-commented and generate all results when executed *without further modifications*, as well as contain all *explicitly* requested answers, comments, discussions and/or images.

2. Separate code + report.

   2.1. A *short* report (.pdf file format) that includes the *explicitly* requested answers, comments, discussions and/or images for each task.

   2.2. Code (.m/.py files). The code should be well-commented, and there should be <u>one</u> main script that generates all results when executed *without further modifications*.

Further reading

- P. P. Bruyant, E. A. Moore, M. J. Graves, M. R. Prince, *Analytic and Iterative Reconstruction Algorithms in SPECT*, Journal of Nuclear Medicine 43(10):1343-1358 (2002)

- (Advanced) K. Lange, R. Carson, *EM Reconstruction Algorithms for Emission and Transmission Tomography*, Journal of Computer Assisted Tomography 8(2):306-316 (1984)