A2 Coursework

Yuiko Ichikawa

March 2025

1 Module 1 PET-CT Image Reconstruction

1.1 Cleaning of sinogram

The CT sinogram was corrected using the provided dark and flat fields. The dark field captures the detector response in the absence of incident radiation, representing background signal or bias. The flat field characterizes the relative sensitivity of each detector element under uniform illumination. These were used to normalize the raw CT measurements as follows:

```
ct_corrected = (ct_sino-ct_dark)/(ct_flat-ct_dark)
```

This correction compensates for detector inhomogeneities and background signal. The normalized data were then converted to attenuation coefficients by taking the negative logarithm. The cleaning process is shown as Figure 1.

The PET sinogram was corrected for spatial sensitivity variations among detectors using the provided gain calibration sinogram. This calibration was obtained by exposing the system to a known homogeneous source. The correction was applied as:

```
pet_corrected = pet_sino / pet_cal
```

1.2 Obtaining attenuation field

To obtain the attenuation field for PET, the CT image must first be reconstructed. Since the sinogram dimensions of PET and CT differ, interpolation is performed in image space after reconstruction.

The CT image was reconstructed using two methods: Filtered Back Projection (FBP) and Ordered Subset Simultaneous Algebraic Reconstruction Technique (OS-SART). For FBP, the iradon function from scikit-image was used. Figure 3 shows the result of the reconstruction using this analytical method.

In addition, OS-SART was implemented. The sinogram was divided into 10 angular subsets, and the reconstruction was iteratively updated. Through empirical tuning, a step size of $\gamma = 0.001$ and approximately 500 iterations

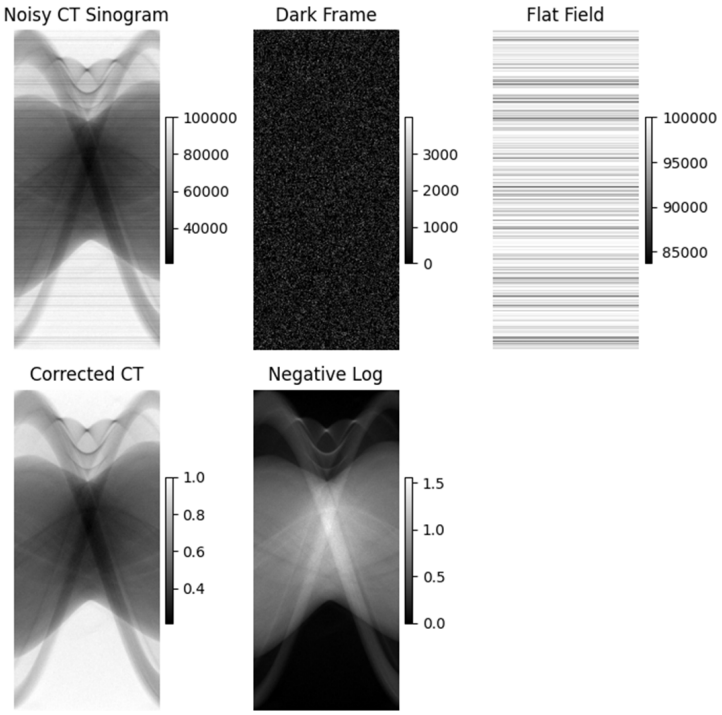

Figure 1: Cleaning process of CT sinogram

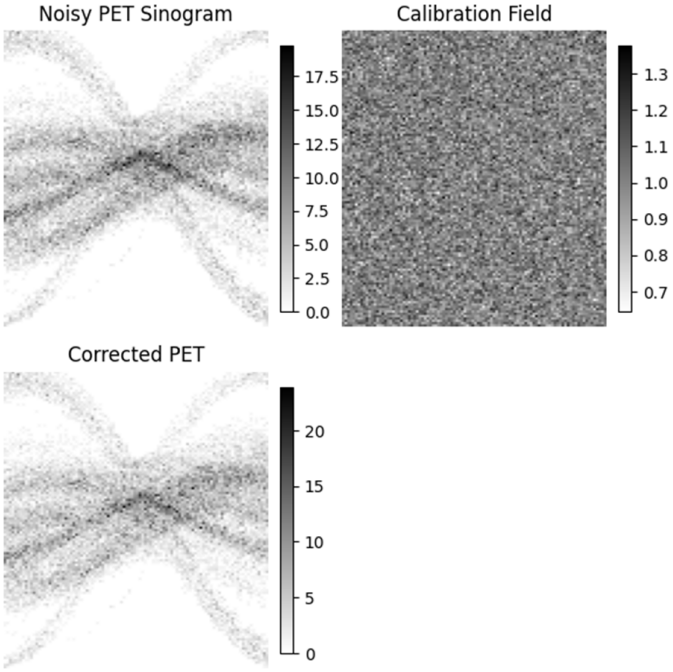

Figure 2: Cleaning process of PET sinogram

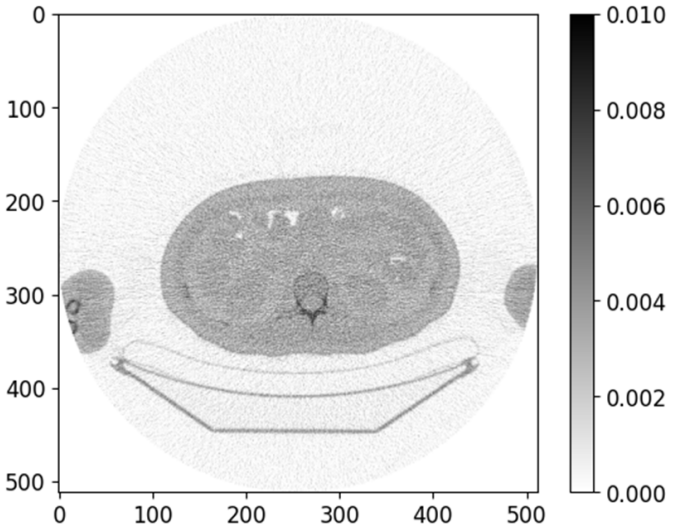

Figure 3: FBP reconstraction of CT image

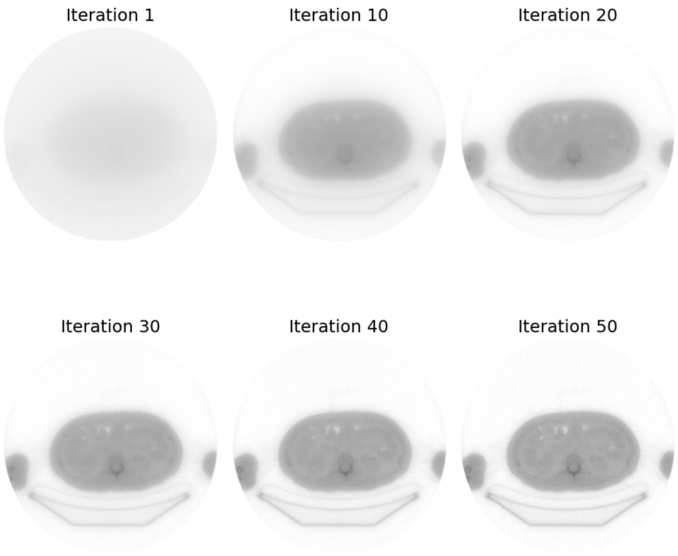

Figure 4: Reconstruction iteration steps of OS-SART

were found to produce an image superior to that of FBP. Figure 4 shows the progressive reconstruction at different iteration steps.

To compare the efficiency of OS-SART with the classical SART method, both algorithms were evaluated in terms of computation time. OS-SART completed 500 steps in approximately 20 seconds on CSD3, whereas SART required a similar amount of time to perform only 50 steps. Moreover, the quality of SART's result after 50 steps was comparable to that of OS-SART after just 100 steps (Figure 5), demonstrating that OS-SART is a more efficient method overall.

The reconstructed CT image, using OS-SART, was then interpolated to match the spatial resolution of the PET image. A forward projection (Radon transform) was applied to the rescaled CT image using the same angular sampling (100 steps) as the PET scan, producing the attenuation sinogram necessary for PET attenuation correction. Figure 6 shows the result of PET attenuation correction.

1.3 Reconstruct PET image

The PET image was reconstructed using both FBP and Ordered Subsets Expectation Maximization (OSEM).

The result of the FBP reconstruction is shown in Figure 7. The reconstructed image exhibits prominent radial streak artifacts.

In contrast, the OSEM algorithm produced a significantly cleaner image, successfully recovering the anatomical structures of the highlighted organs. Figure

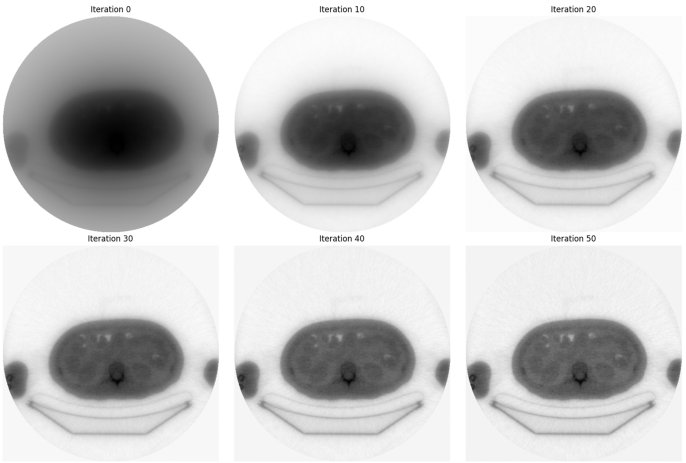

Figure 5: Reconstruction iteration steps of SART

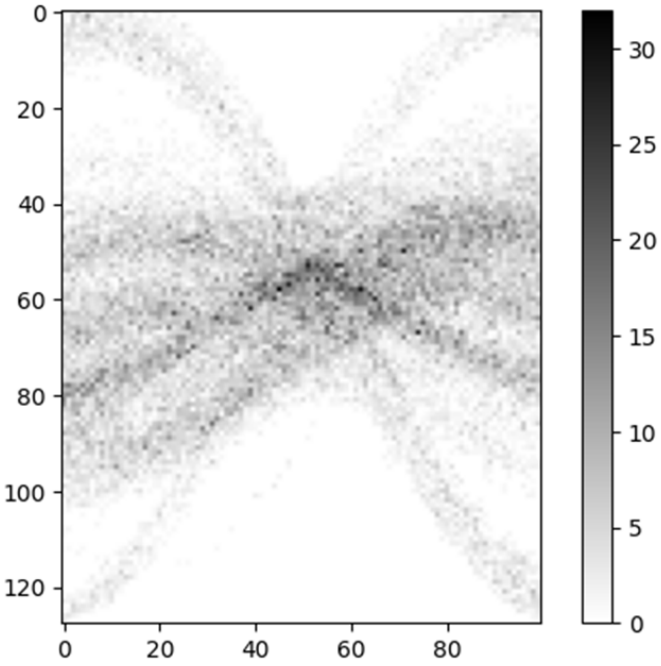

Figure 6: Attenuation corrected PET

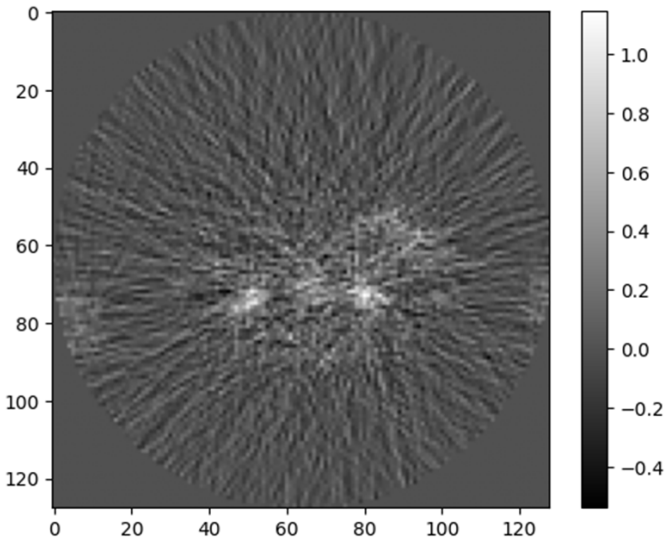

Figure 7: PET reconstruction using FBP

8 presents the results after 10 iterations with 10 subsets, demonstrating improved contrast and reduced artifacts. MLEM was also performed with varying numbers of iterations, and the results showed that, for the same number of iterations, OSEM produced higher-quality images. As shown in Figure 9, the MLEM result after 10 iterations appears more blurred compared to OSEM.

1.4 Theory

1.4.1 The overlay of PET and CT scans

The overlay of the PET and CT scans is presented in Figure 10. The CT scan provides the overall anatomical structure, while the PET scan highlights areas with high concentrations of radioactive tracers.

1.4.2 Accuracy of time measurement required for TOF-PET scanner

In a Time-of-Flight PET (TOF-PET) scanner, the spatial uncertainty along the line of response (LOR) is given by:

$$\Delta x = \frac{c\,\Delta t}{2}$$

where: Δx is the spatial resolution, c is the speed of light (approximately 3×10^8 m/s), Δt is the time resolution of the system.

To localize the positron annihilation point so precisely that image reconstruction is unnecessary, the spatial resolution Δx must be at least as good as the desired image resolution.

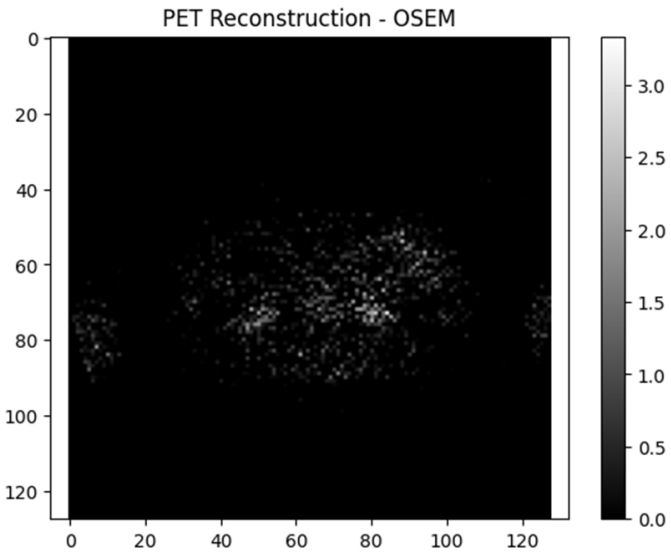

Figure 8: PET reconstruction by OSEM

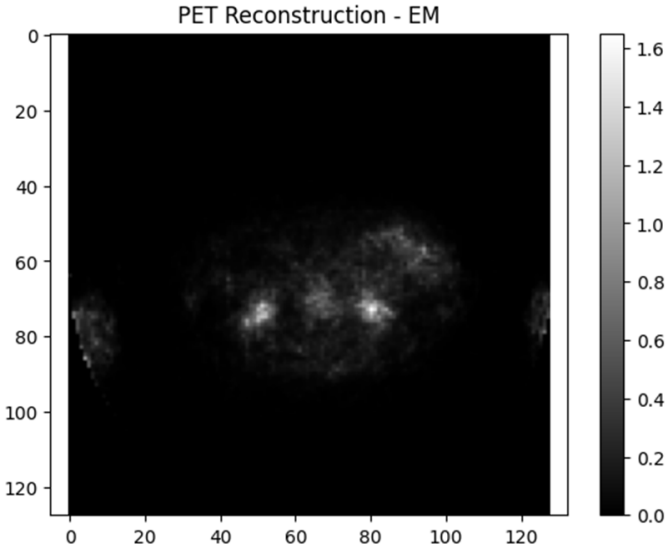

Figure 9: PET reconstruction by EM

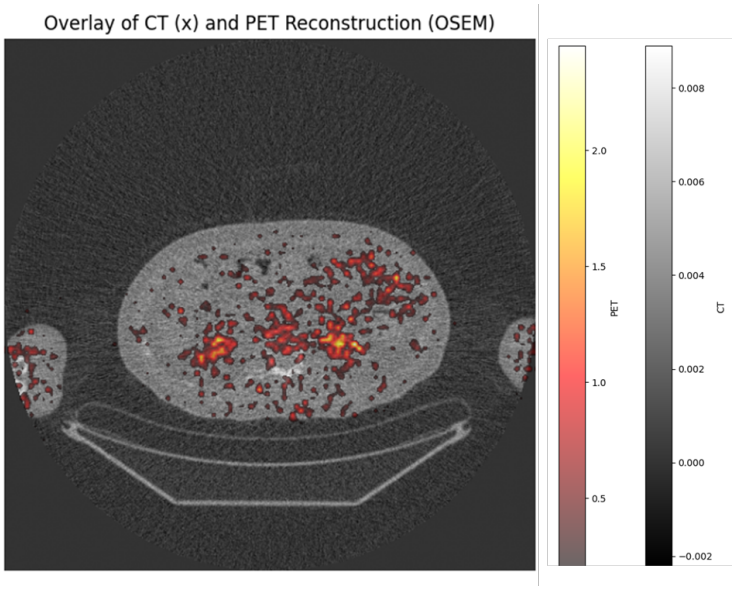

Figure 10: Overlay of CT (gray) and PET (colour) reconstructions. PET image is masked out if magnitude is less than 0.2.

Given a target spatial resolution of $d = 4.26\,\text{mm} = 4.26 \times 10^{-3}\,\text{m}$, we can solve for the required time resolution Δt as follows:

$$\Delta t \leq \frac{2 \times 4.26 \times 10^{-3}}{3 \times 10^{8}} \approx 28.4\,\text{ps}$$

1.4.3 OSEM's working

In PET, the measured data are counts of discrete photon detection events, which follow a Poisson distribution. OSEM models this appropriately by maximizing the likelihood of the observed data under the Poisson noise assumption. In CT, the raw measurement data (line integrals of X-ray attenuation) are much higher in count and can be approximated as Gaussian-distributed due to the Central Limit Theorem. Therefore, likelihood-based approaches will be Gaussian Mixture modelling, which will be computationally inefficient for such large number of data points, and not suitable the task. Instead, analytical methods like Filtered Back Projection (FBP) or iterative gradient-based solvers are more appropriate and computationally efficient for CT reconstruction.

1.4.4 PET-MR scanner

In PET-MR scanners, the main challenge is the lack of direct attenuation information from MRI. Unlike CT, which directly measures X-ray attenuation coefficients, MRI measures proton density and relaxation times, which are not

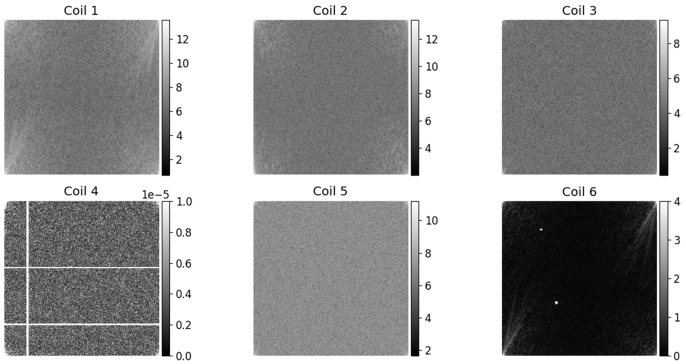

Figure 11: Magnitude and phase images from Coil 1

directly related to photon attenuation. Therefore, MR-based attenuation correction methods are practiced. One such example is segmentation-based method. This method segment MR images into different tissue types and assign corresponding attenuation coefficients.

2 Module 2 MRI Image Denoising

2.1 Exercise 2.1 Visualisation and Identifying Noise

Data was loaded using the numpy.load method. Since the loaded data has a shape of (6, 280, 280) and the number of coils is specified to be six, the 0th dimension corresponds to the coil index.

```
import numpy as np kspace = np.load("kspace.npy") print(kspace.shape)
returns (6, 280, 280)
```

The magnitude images of k-space for each coil are shown in Figure 11. Coils 1–3 and 5 exhibited a similar sandy texture and had magnitudes of the same order. Coil 4 showed a relatively weak signal, with its image dominated by strong crisscrossing noise. Coil 6 displayed a point-like noise pattern, and its signal was significantly weaker than the others.

Similarly, the image-space transformation for the first coil is shown in Figure 12. The magnitude image exhibits a strong signal in the upper half, while the lower half contains little information. However, the phase image reveals a continuous phase structure extending beyond the visible magnitude, suggesting that signals are present even in regions where the magnitude is weak.

Magnitude images from all six coils are shown in Figure 3. the Logarithm was taken to visualize wider range of magnitudes variations. Coils 1–3 display relatively clear structures, although Coil 3 has a slightly lower magnitude, making it appear darker. Despite being noisier, Coil 5 captures a signal in the

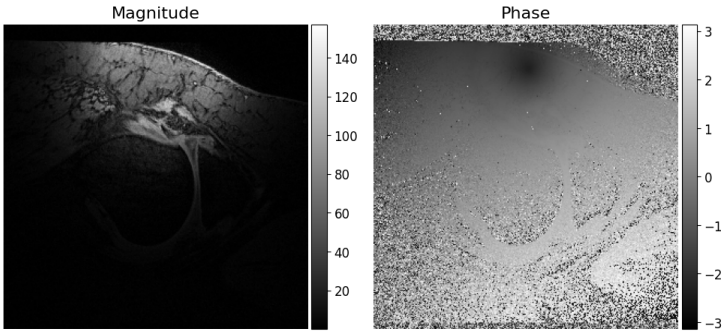

Figure 12: Enter Caption

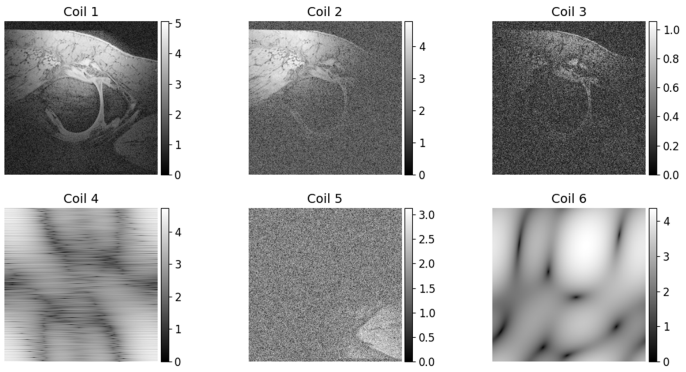

Figure 13: Magnitudes of image-space projection of all coil observations.

lower-right corner of the image, which is not covered by the other coils. The bottom-left image shows a strong zig-zag pattern, while the bottom-right has a vague wave-like texture. Both offer little useful information without further denoising.

Figure 4 shows the result of combining all six coil images using the Root-Sum-Square method. The combined image is dominated by the corduroy pattern from Coil 4 and the wave-like structure from Coil 6, which obscures signals from the other coils. In particular, contributions from Coils 3 and 5—though weaker in magnitude but richer in actual signal—are completely masked by the noise introduced by Coils 4 and 6.

2.2 Exercise 2.3 Visualisation and Identifying Noise

Mean, Gaussian, and wavelet filters were applied to all six images. Before applying these homogeneous filtering methods, targeted denoising was performed to address the high-magnitude noise observed in Coils 4 and 6. For Coil 4, noise was removed by thresholding values above 1.010^4. This threshold was chosen

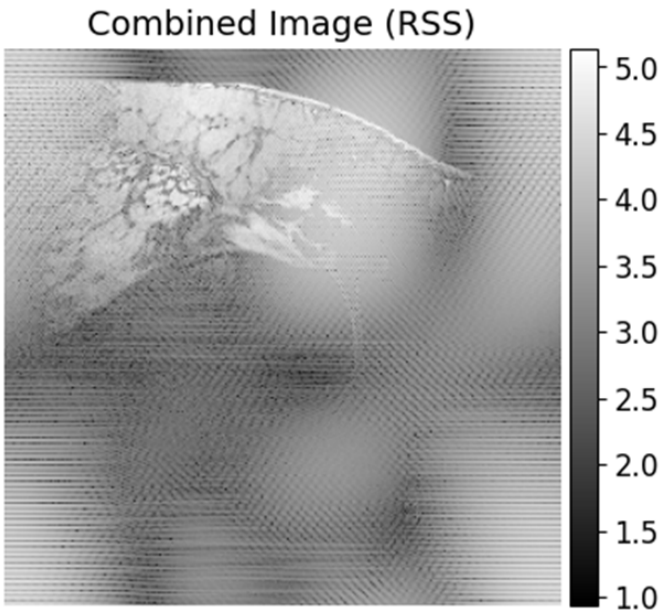

Figure 14: A Root-Sum-Square of the observations from coils 1–6.

arbitrarily. In the case of Coil 6, which exhibited point-like noise only in one half of the image, denoising was performed using complex conjugate mirroring. Figure 5 shows the results after thresholding and mirroring were applied to Coils 4 and 6, respectively. While Coil 4 still lacks clear structure, Coil 6 reveals a more distinct signal in the upper right region of the image.

```
# Threshold denoising
kspace_denoised[3, :, :] = np.where(abs(kspace[3, :, :])>0.00001, 0,
    kspace[3, :, :])
```

```
# Complex conjugate denoising
H, W = kspace.shape[-2], kspace.shape[-1]
for i in range(H):
    for j in range(W // 2):
        i_mirror = (H - i) % H
        j_mirror = (W - j) % W
        kspace_denoised[5, i, j] = np.conj(kspace[5, i_mirror, j_mirror])
```

Mean and Gaussian filters are both low-pass filters that remove high-frequency noise by performing spatial averaging.

Figure 6 shows the mean-filtered images from all six coil observations. The convolution kernel used was a 3×3 matrix with uniform weights. The filtered

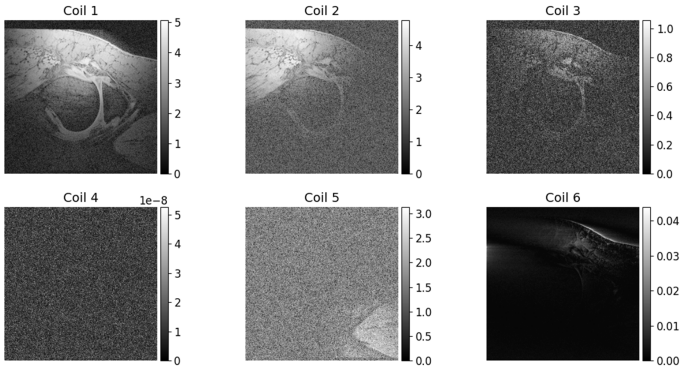

Figure 15: Results after pre-denoising before applying homogeneous filters.

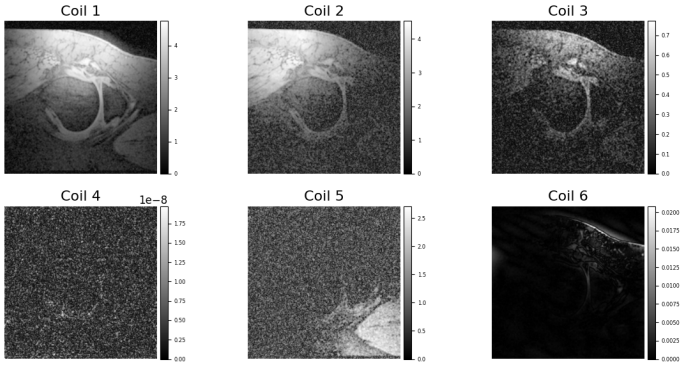

Figure 16: Mean filtered images from all six coils.

images appear less grainy but also more blurred compared to the originals. This blurring results from the spatial averaging applied uniformly across the image, including across edges. The filter suppresses noise with wavelengths smaller than the kernel size, but allows through noise of larger spatial scales. Moreover, due to spatial discontinuities at the image boundaries, the filter introduces high-frequency components in the frequency domain, which may leave some residual high-frequency structures behind.

Figure 7 shows the results of applying a Gaussian filter. The convolution kernel was similar in size to that of the mean filter but had weights distributed according to a Gaussian function centered in the matrix with $\sigma = 1.5$. Because the Fourier transform of a Gaussian is another Gaussian, this filter acts more smoothly in the frequency domain, leaving fewer high-frequency components. Because of this, the Gaussian filter removed high-frequency noise in Coils 4 and 6 more effectively than the mean filter.

Figure 8 shows the result of wavelet-based denoising. This method transforms each image into wavelet subbands and removes noise by thresholding

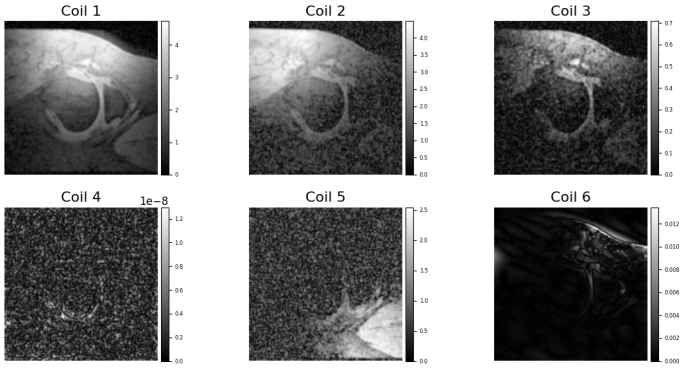

Figure 17: Gaussian filtered images from all six coils ($\sigma = 1.5$).

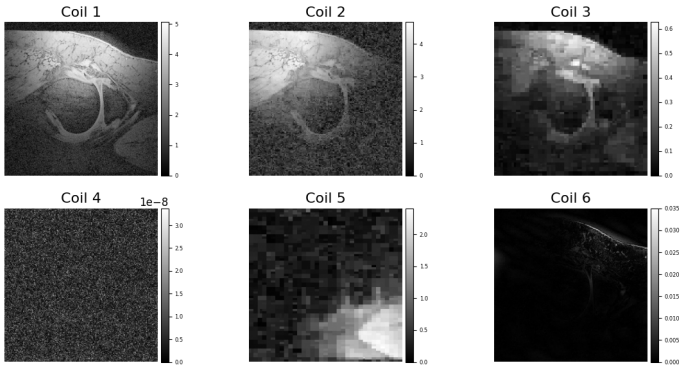

Figure 18: Wavelet filtered images from all six coils.

coefficients below a certain value, with thresholds tailored to each wavelength. Among the three methods tested, wavelet filtering best preserved edge structures—likely because edges correspond to strong, localized features in the frequency domain. However, it failed to remove noise from Coils 4 and 6, possibly because the noise and signal could not be easily separated in wavelet space. Additionally, the filtered images for Coils 3 and 5 appeared pixelated, suggesting that meaningful high-frequency components may have been lost during denoising.

Each filtering method had its own advantages and drawbacks. For Coils 1–2, which had relatively clean signals and well-defined structures, wavelet filtering performed best—effectively preserving edges and small-scale features while removing background noise. For Coils 4 and 6, which were dominated by noise, Gaussian filtering gave the best results. Combining different filtering methods depending on the characteristics of each coil image may offer an effective strategy to improve overall image quality.

Figure 19 shows the Root-Sum-Squared image after applying Gaussian filter-

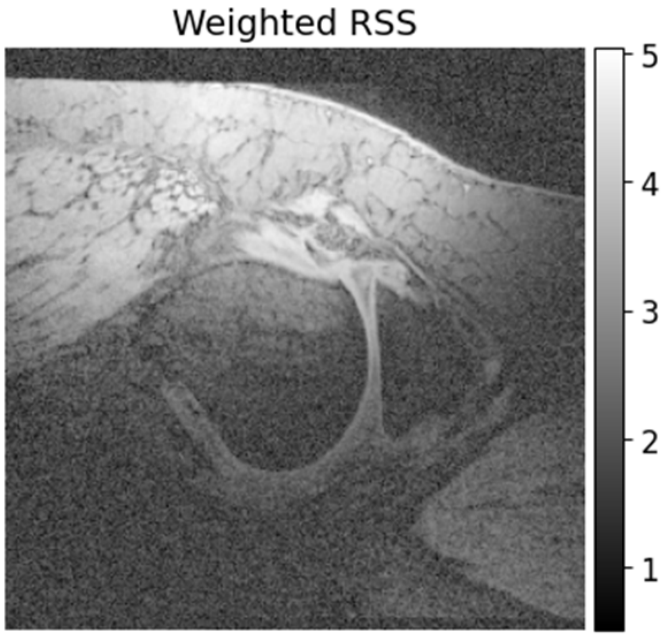

Figure 19: Root-Sum-Squared images after Gaussian filtering ($\sigma = 0.5$).

ing. Since the overall magnitude of Coils 4 and 6 is reduced after denoising, the resulting signal primarily originates from Coils 1–3 and 5. Most of the anatomical structures observable in the individual coils are retained. The grainy texture is still present but noticeably suppressed.

3 Module 3: CT Image Segmentation and Classification

3.1 Image Segmentation

The NIfTI image files were cropped around the tumor regions, leaving a margin of 30 voxels in the x and y dimensions and 5 voxels in the z dimension. The cropped images were saved in 3D NumPy array format. This operation was performed using Python's SimpleITK package.

Based on the voxel value distributions, a min–max thresholding approach was applied. For each patient, the minimum and maximum voxel values corresponding to tumors were determined. Voxels with values between the patient-specific minimum and maximum were classified as tumor. This method performed reasonably well for patients whose tumor voxel values spanned a narrow range, resulting in relatively few false positives. However, for patients with a

broader range of tumor voxel values, it was difficult to distinguish tumor from non-tumor voxels.

To illustrate this, Figure 20 shows histograms of voxel values from Patients 12 and 22, representing a relatively successful and an unsuccessful segmentation case, respectively. For Patient 12, the tumor voxel values fall within a narrow range distinct from the background, while for Patient 22, the tumor value distribution overlaps significantly with non-tumor voxels.

Table 1 presents the confusion matrix of the classification results. The thresholding method was designed to eliminate false negatives, resulting in a 0% false negative rate. However, because a substantial number of non-tumor voxels fell within the same threshold range, the false positive rate was high (approximately 60%), leading to a very low true positive rate.

This performance was expected based on the voxel value distributions and the imbalance between tumor and non-tumor voxels. Since tumor regions are relatively small, even a narrow tumor value range can encompass many non-tumor voxels. This issue persists even in cropped images. To improve the segmentation method, preprocessing steps that better separate tumor characteristics from background may be needed. Additionally, more advanced classification methods—such as linear regression or neural networks—could outperform basic thresholding.

Outcome	**Count**	**Ratio**
height	True Negatives	2,451,716
0.40	False Positives	0
0.00	True Positives	74,707
0.02	height**Total**	4,188,763
1.00	height	

Table 1: Confusion matrix with absolute counts and normalized ratios.

3.2 Image Feature Extraction and Classification

Three features—Energy, Mean Absolute Deviation (MAD), and Uniformity—were computed using the following code. For Uniformity, the number of histogram bins was set to 100. Given the global voxel value range of -1024 to 1733, this bin count provides a resolution of approximately 27 units per bin. The code below demonstrates how each feature was calculated:

```
def extract_features(voxels, n_bins=100):
    if len(voxels) == 0:
        return np.nan, np.nan, np.nan

    # Energy
    energy = np.sum(voxels**2)

    # Mean Absolute Deviation (MAD)
    mean_val = np.mean(voxels)
```

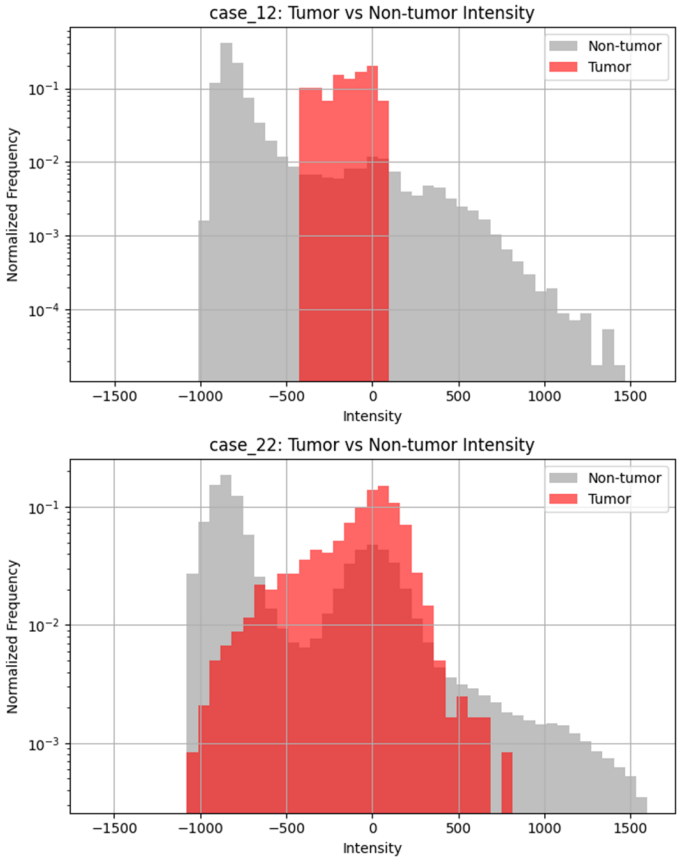

Figure 20: Histograms of voxel values for images cropped around tumors. Red represents tumor voxels; gray denotes non-tumor voxels. Top and bottom panels correspond to Patients 12 and 22, respectively. Histograms are normalized such that the cumulative sum for each class equals 1.

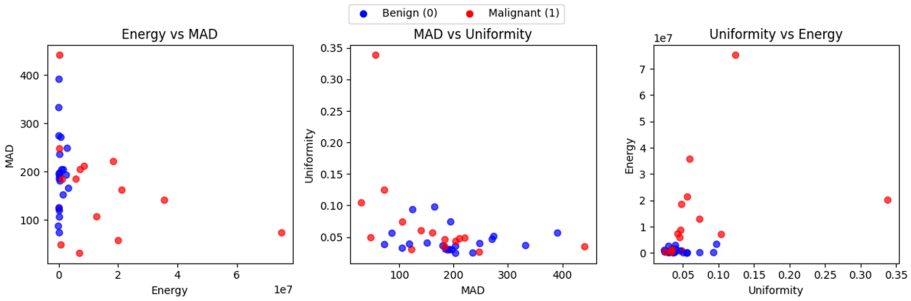

Figure 21: Scatter plots comparing Energy, Mean Absolute Deviation (MAD), and Uniformity. Blue and red indicate benign and malignant labels, respectively.

```
    mad = np.mean(np.abs(voxels - mean_val))

    # Uniformity
    hist, _ = np.histogram(voxels, bins=n_bins, density=False)
    p = hist / np.sum(hist)
    uniformity = np.sum(p**2)

    return energy, mad, uniformity
```

Figure 20 shows 2D scatter plots comparing pairs of these features. Among the three, Energy showed the most distinguishability when used alone; higher energy values tended to be associated with malignancy. The combination of Uniformity and Energy also showed promising separation, with benign and malignant cases forming vague clusters. The best classification performance is likely achieved when using all three features, as each provides complementary information. Based on such observation, a logistic regression classification maybe suggested as the best approach.

3.3 Acknowledgement

Author acknowledges the use of ChatGPT-4o and Google Colaboratory's AI-powered tools for assisting with language refinement and code optimization throughout this work.