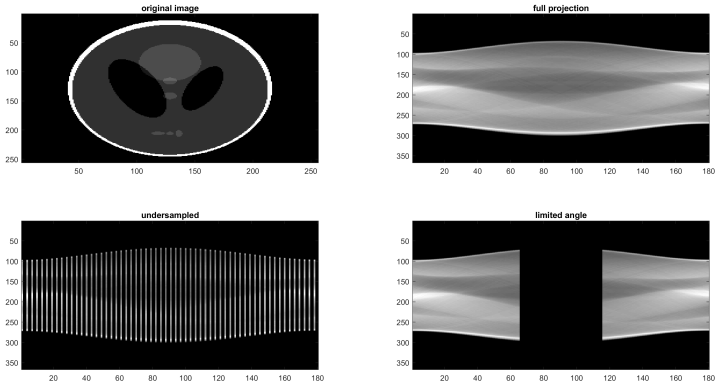
COMP0114 Inverse Problems in Imaging. MiniProject

Part B : Advanced Topics

Choose *one* topic from the following list

Advanced Topic 1: Inpainting in Sinogram Space
You have seen from lectures and from part A of this assignment that the CT reconstuction problem becomes severely ill-posed when limited data is acquired, both for the regularly undersampled and for the limited angle problem. Another way to intepret this is to treat the sinogram as an image with missing data, and use techniques from *InPainting* to fill in the missing projections. After this correction to the data, the inverse problem can be solved by filtered back-projection in the normal way.



As defined in lectures, one way to solve in-painting is to use a diffusion process :

$$\begin{aligned} \frac{\partial \boldsymbol{g}}{\partial t} &= -\alpha \mathcal{L}(\boldsymbol{g})\boldsymbol{g}, \qquad \boldsymbol{g} \in \overline{\Omega} \\ \text{subject to } \boldsymbol{g} &= \boldsymbol{g}^{\mathsf{obs}} \qquad \text{on } \partial\Omega \end{aligned}$$

where Ω, $\overline{\Omega}$ are the regions of the sinogram with measured data and missing data respectively, and $\partial\Omega$ is the boundary between the two. $\boldsymbol{g}^{\mathsf{obs}}$ is the measured data, and $\boldsymbol{g}$ is the full sinogram image.

Things to try

- Create sinogram data for both cases i) full angle with undersampled projections, ii) limited angles. Create a full size sinogram by adding zeros in the columns corresponding to missing measurements.
- Use inpainting based on an istotropic Laplacian $\mathcal{L} = -\nabla^2$ to fill in the missing data; the result is $\boldsymbol{g}^{\mathsf{corrected}}$. Apply filtered backprojection to $\boldsymbol{g}^{\mathsf{corrected}}$ to reconstruct the image $\boldsymbol{f}_{\mathsf{recon}} = \mathcal{R}^{-1}\boldsymbol{g}^{\mathsf{corrected}}$.
- After reconstruction, investigate whether the reconstructed image can be further improved using denoising in the image domain.

- Use an anisotropic regulariser such as smoothed TV for the data correction step and compare your results.

Software

No extra software is required beyond what you have used in part A. You can make use of the example in-painting code (in Matlab) from lectures as a starting point.

Background Literature

Image and Sinogram Inpainting

- A general reference on image inpainting: Shen, Jianhong, and Tony F. Chan. "Mathematical models for local nontexture inpaintings", *SIAM Journal on Applied Mathematics* **62 (3)**, (2002): 1019-1043.
- A specific reference about inpainting in sinogram space: Robert Tovey, Martin Benning, Christoph Brune, Marinus J Lagerwerf, Sean M Collins, Rowan K Leary, Paul A Midgley and Carola-Bibiane Schönlieb, "Directional sinogram inpainting for limited angle tomography", *Inverse Problems* (2019).

Advanced Topic 2: Low photon count and Poisson noise model
Although the Radon transform and its inverse are correct for integral transform data, actual measured data are of course given for a discrete number of detectors and have random variations due to measurement noise. In part A, you used least squares minimisation, corresponding to normally distributed data. However, when the number of photons is low, a better model for emissions measurements is Poisson statistics.

This leads to the negative log Poisson likelihood (NLPL), which is the same as the Kullback-Leibler generalised distance, for the data term.

Things to try

- Generate data by using a Poisson pseudo-random number generator based on the Radon Transform, i.e. $\boldsymbol{g}^{\mathsf{obs}} \sim \mathsf{Poisson}(\mathsf{A}\boldsymbol{f})$.[1]
  You will need to think about the magnitude of the image $\boldsymbol{f}$ (and/or the system model A) to get reasonable values for the mean data $\boldsymbol{g}_{\mathsf{true}} = \mathsf{A}\boldsymbol{f}_{\mathsf{true}}$ (try different values to simulate different levels of noise).
- Weighted least squares reconstruction using Gaussian approximation to Poisson noise, i.e. $\sigma = \sqrt{\boldsymbol{g}_{\mathsf{true}}}$. Include an appropriate regularisation as in part A. As in practice you won't know $\boldsymbol{g}_{\mathsf{true}}$, try to use the "plug-in" method where you use the data itself as variance (i.e. $\sigma = \sqrt{\boldsymbol{g}^{\mathsf{obs}}}$ as in eq. 10.12 from lectures). Take care to avoid dividing by zero if there are zero photons detected. Handle any zeroes in the measured data.
- Implement one-step late MLEM using (eq. 10.23 from lectures). Use both small and large values of the regularisation parameter α. Notice if the algorithm fails when α is too high, e.g. by producing negative values in the reconstruction or by diverging.
- Try to use another optimisation method from numerical libraries and compare the performance with one-step late MLEM.

In all the above tasks, you should be considering only a 2D problem. The matrix A could be explicitly constructed, or the solution could be done with matrix-free methods.

Software You can use the same software as you used for Part A. Both Matlab and Python have built in optimisation packages that you can use for the final task.

Background Literature

- Original description of one-step late MLEM: P.J. Green, "Bayesian reconstructions from emission tomography data using a modified EM algorithm", *IEEE transactions on medical imaging*, **9**(1), 84-93, 1990
- The original BFGS-B paper: Byrd, R. H.; Lu, P.; Nocedal, J.; Zhu, C. "A Limited Memory Algorithm for Bound Constrained Optimization". *SIAM J. Sci. Comput.* **16** (5): 1190–1208, (1995)

[1] Here A is the discrete version of the Radon Transform $\mathcal{R}$

Advanced Topic 3: Learning Based Reconstruction
This project is about learned image reconstruction and generalisation capabilities. **Note:** depending on your hardware the training times for the networks can be long (in the order of multiple hours), so you need to reserve enough time for the project.

Learned image reconstruction with model information can be separated into the two general classes:

i.) Initial reconstruction and learned post-processing

ii.) Learned model-based iterative reconstructions

Both approaches have proven to be powerful in practice, but have shown a different behaviour in terms of generalisation capabilities. In this project we want to examine these generalisation capabilities with a simple experiment: Train a network for each approach with a training set created from random ellipses and test reconstruction quality for the Shepp-Logan phantom.

Given a phantom from the training set $\boldsymbol{f}_{\mathsf{true}}$ and corresponding noisy measurement data $\boldsymbol{g} = \mathcal{R}\boldsymbol{f} + \delta\boldsymbol{g}$. To create an initial reconstruction, apply filtered backprojection $\boldsymbol{f}_0 = \mathcal{R}^{-1}\boldsymbol{g}$.

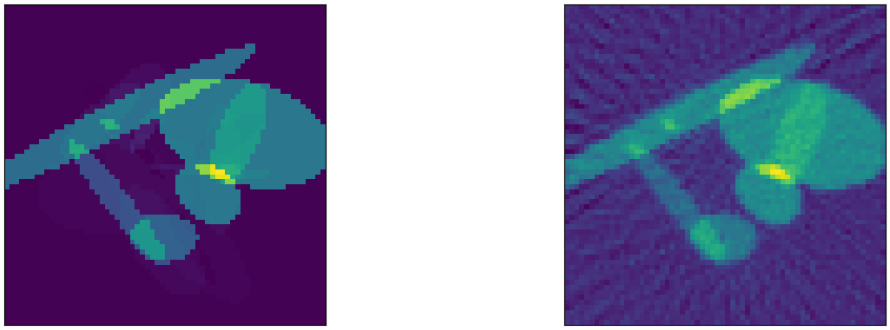


Figure 1: A possible training set: (Left) the true phantom, (Right) filtered-backprojection from noisy data

We now aim to train a convolutional neural network (CNN) G_θ that corrects the initial reconstruction by filtered backprojection. The training is performed in a supervised manner, that means we aim to find an optimal set of parameters such that a loss function is minimized:

$$\text{loss}(\theta; \boldsymbol{g}) = \|G_\theta(\boldsymbol{f}_0) - f_{true}\|_p,$$

for a suitable choice of p.

Things to try

To keep training times reasonsable (especially if no GPU is available), it is advised to restrict the image resolution to 64x64. For the model based approach you can use matrix multiplication for forward and adjoint (see Part A, task 2), this way the gradient for the backpropagation can be easily calculated internally (by automatic differentiation).

- We need a training set; for that write a function that creates a phantoms of random ellipses, for instance between 5 and 20 ellipses that can overlap (remember to normalise all phantoms to [0,1]) (see Figure X).
- Train a post-processing network architectures for approach i.). Here we can use a simple image-to-image ResNet architecture. This architecture consists of a repeated residual block that consists of 2 convolutional layers and an additive residual connection. The residual block is then repeated for a set amount, e.g. between $n = 5, \ldots, 10$. (This could be considered as n iterations)
- Train a model-based network: Treat each residual block as an iteration that outputs one updated reconstruction, then you need to compute the gradient of the data fit $\mathsf{A}^\mathsf{T}(\mathsf{A}\boldsymbol{f} - g)$ before each residual block of your ResNet[2]. By concatenating current reconstruction and gradient you can use this as input to each residual block. This way you obtain a simple n iterations model-based network.
- Finally, test the reconstruction results for the Shepp-Logan phantom.

Software This is strongly recommended to solve with Python. You can use your favourite package for the deep learning environment, such as TensorFlow or PyTorch.

Background Literature

- Training a network for post-processing: Jin, Kyong Hwan, Michael T. McCann, Emmanuel Froustey, and Michael Unser. "Deep convolutional neural network for inverse problems in imaging". *IEEE Transactions on Image Processing*, **26**, no. 9 (2017): 4509-4522
- Model-based iterative networks: Adler, Jonas, and Ozan Öktem."Solving ill-posed inverse problems using iterative deep neural networks". *Inverse Problems* **33**, no. 12 (2017): 124007.
- The network architecture ResNet: He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition". *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778. 2016.

[2]Here A is the discrete version of the Radon Transform $\mathcal{R}$