PSBC Assignment 2: Tomography

March 18, 2015

Louis Runcieman

Department of Mathematics

University of Manchester

Academic year 2014-2015

Abstract

Tomography is a prodigious tool in substantial fields of medicine and has applications in archaeology, materials science, astrophysics and oceanography. It refers to imaging a sample by sectioning, using penetrating waves such as X-rays or gamma rays to reconstruct images through reconstruction algorithms [1]. This paper explores the basic ideas surrounding the theory of tomography with all programming done through GNU Octave (please note, this may induce compatibility issues with MATLAB).

1 Introduction to Tomography

Consider the space of all straight lines $\mathbb{L} \subseteq \mathbb{R}^2$ and let $\mathbf{x} = (x, y) \in \mathbb{R}^2$. The two dimensional Radon transform, Rf, is defined by the line integral $Rf(\mathbb{L}) = \int_{\mathbb{L}} f(\mathbf{x})|\mathrm{d}\mathbf{x}|$ of a function f, known as the linear attenuation coefficient [2]. By parametrising each straight line $L \in \mathbb{L}$ this can be written in the form,

$$Rf(s,\theta) = \int_{-\infty}^{\infty} f(s\Theta^{\perp} + t\Theta)\mathrm{d}t,$$

for arc length t, distance s of line L from the origin, angle θ between the x axis and the normal vector to L and $\Theta = (\cos\theta, \sin\theta)$. In hard field tomography, radiation travels in a straight line (such as X-rays or gamma rays); the linear attenuation coefficient f represents an unknown density and the Radon transform represents the projection data obtained as the output of a tomographic scan. Thus image reconstruction can be achieved through the inverse of the Radon transform.

The discrete version of the Radon transform will be regarded as a sparse matrix and its inversion investigated through a linear system of equations. This estimates to a reasonable approximation the sum of the pixel values, giving the attenuation coefficient along each ray (represented by a line). For each ray a linear system of equations,

$$Ax = b, \tag{1}$$

is constructed [1], where measurements of data are contributed to the vector b and the object of interest is the vector x - representing the image pixel values. Hence, A is the *number of measurements* by *number of pixels* matrix; its entries are zero when the ray in question does not hit that pixel. The test image will be a low resolution Modified Shepp-Logan phantom [3], which serves as the model of a human head and is used to test numerical accuracy of two dimensional image reconstruction algorithms.

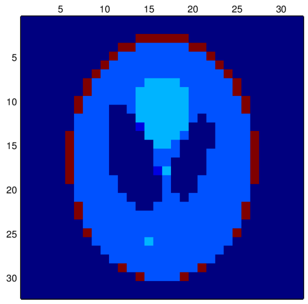


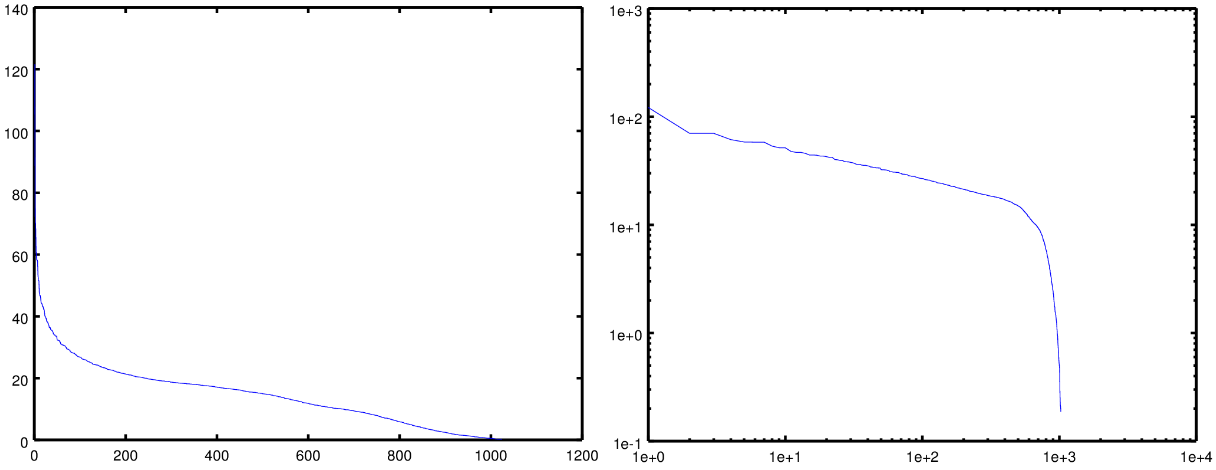
Figure 1: Image of modified Shepp-Logan phantom.

Listing 1: Generating test image and matrices for corresponding linear system of equations.

```
Image=phantom(32); %Modified Shepp-Logan phantom.
figure; imagesc(Image) %Image of Modifed Shepp-Logan phantom.
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'XAxisLocation','top','linewidth',2,'FontSize',12);%Edit figure styling.
```

```
5
6  [A,data]=siddonS(Image);
7  ind=find(sum(A,2)); %Detects all rows whose row sum is non-zero.
8  A=A(ind,:); %Removes all zero rows from A.
9  b=data(:); %Reshapes data into a column vector, labelled b.
10 b=b(ind); %Removes entries corresponding to zero rows of A.
11 %In this case, ind is full; lines 8 and 10 make no alterations. Code remains to
12 %be used later.
```

The matrix $A \in M_{m\times n}(\mathbb{R})$ has real entries, which by the Singular Value Decomposition Theorem [4] gives the existence of a factorisation form, $A = U\Sigma V^T$, such that U and V are $n \times n$ orthogonal matrices (whose columns are known as the left and right singular vectors of A respectively) and Σ is a non-negative $n \times n$ diagonal matrix. This decomposition yields explicit representation for the range (and null space) of the matrix, in particular the left singular vectors corresponding to the non-zero singular values of A (which is the same value as the number of non-zero entries of Σ).



(a) Plot.

(b) Natural log-log plot.

Figure 2: Plot and natural log-log plot of the singular values of A.

The decay of singular values shown in Figure 2 is explained by numerical rounding errors from floating point arithmetic leading to small, but non-zero singular values in rank deficient matrices. Such singular values are neglected and in doing so the effective rank of a matrix can be determined. The effective rank cannot be determined by Figure 2(a) as the singular values do not have a distinct decay point, however, Figure 2(b) has a sharp decay of values at approximately 800, suggesting this is the effective rank of A.

Listing 2: Plotting graphs for Figure 2.

```
S=svd(full(A)); %Ouputs singluar values of A in full form (non-sparse).
figure; plot(S) %Plot of singular values of A.
figure; loglog(S) %Natural log-log plot of singular values of A.
```

```
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'linewidth',2,'FontSize',12); %Edit figure styling.
```

1.1 Ill Fitting Test Objects

Missing rows from the top and bottom of the sparse matrix A corresponds to the test object not fitting into the width of the sensor. To examine this, the top 2088 and bottom 2088 rows of A will be deleted, equating to 40% of the test object not fitting into the sensor, symmetrically missing 20% of each side.

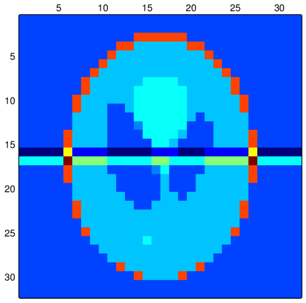
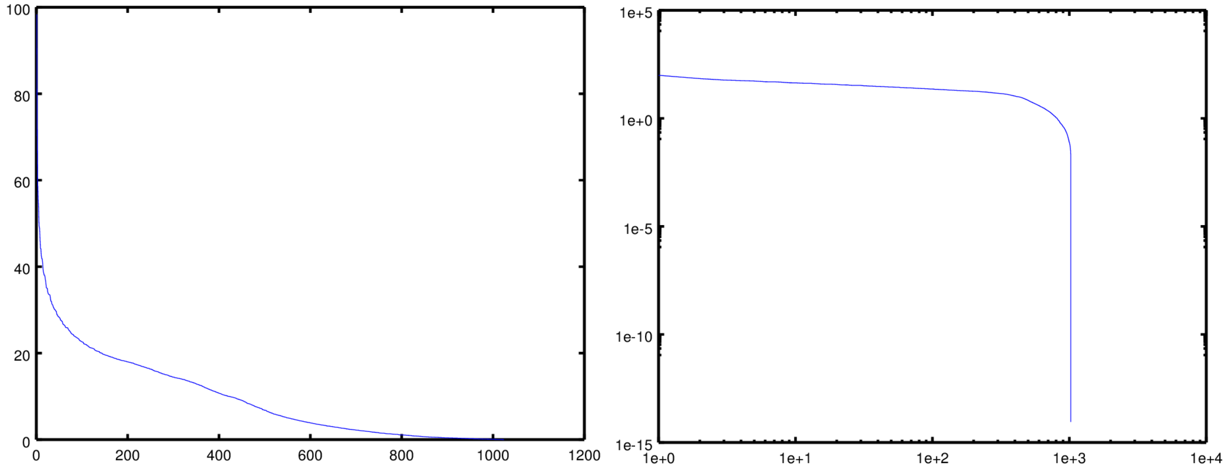


Figure 3: Reconstructed image of A with deleted rows.

As expected, data is missing from the reconstructed image. Figure 3 represents the image given when the sensor only scanned the middle 60% of the human head test object.



(a) Plot.

(b) Natural log-log plot.

Figure 4: Plot and natural log-log plot of singular values of A with deleted rows.

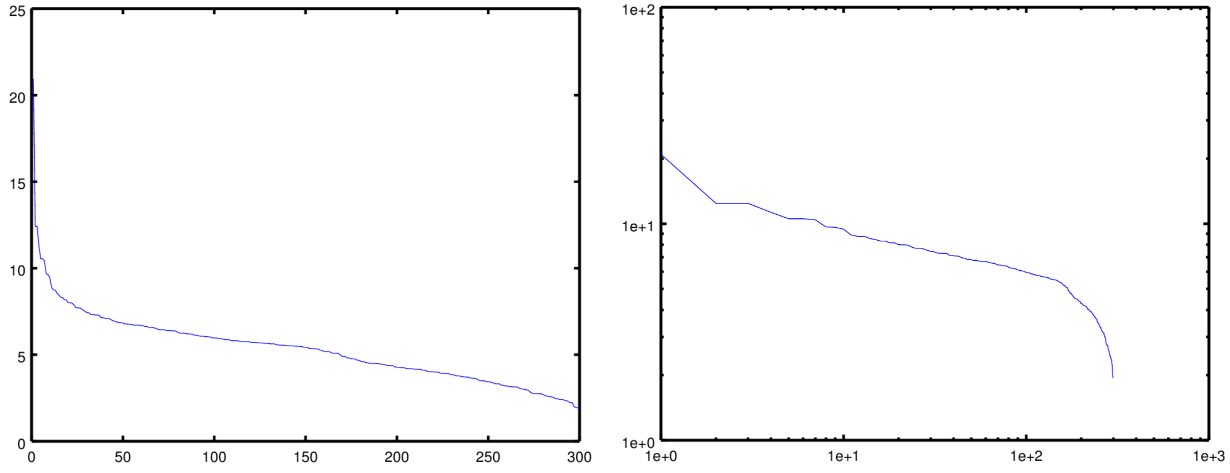
The singular values, plotted in Figure 4 are comparatively smoother than the plots in Figure 2 which is explained by less data interference. With a similar argument about effective rank given for Figure 2, the effective rank of can be more clearly approximated from Figure 4(b) to be 1000. This value is close to the more accurate effective rank of A without any deleted rows, as expected.

Listing 3: Plotting graphs for Figure 4 for `A` with deleted rows.

```
delete=[1:2088 8353:10440]; %Produces vector representing the top 2088 and
                            %bottom 2088 rows/entries to be removed.
Adel=A; %Creating new matrix Adel to represent A with deleted rows.
Adel(delete,:)=[]; %Removes rows specified by delete.
bdel=b; %Creating new vector bdel to represent b with deleted entries.
bdel(delete)=[]; %Removes entries specified by delete.
x=Adel\bdel; %Least squares solution to linear system of equations Ax=b.
figure;imagesc(reshape(x,32,32)); %Reshape reutrns x from vector to matrix form.
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'XAxisLocation','top','linewidth',2,'FontSize',12);%Edit figure styling.
S=svd(full(Adel)); %Ouputs singluar values of A in full form (non-sparse).
figure; plot(S) %Plot of singular values of A.
figure; loglog(S) %Natural log-log plot of singular values of A.
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'XAxisLocation','top','linewidth',2,'FontSize',12);%Edit figure styling.
```

1.2 Minimal Source Rotations

The sensor not being able to complete a full half-turn corresponds to the `nAngles` input of `siddonS` being less than the default value of `180` degrees. In particular, `nAngles` will be set to `30` degrees, with a corresponding value of `10` for `nProj`.



(a) Reduced angle plot of singular.

(b) Reduced angle log-log plot of singular.

Figure 5: Plot and loglog.

The singular values of A have a less distinguishable decay when comparing with a full half rotation. The result of this leaves the effective rank of A not being able to be approximated without great error.

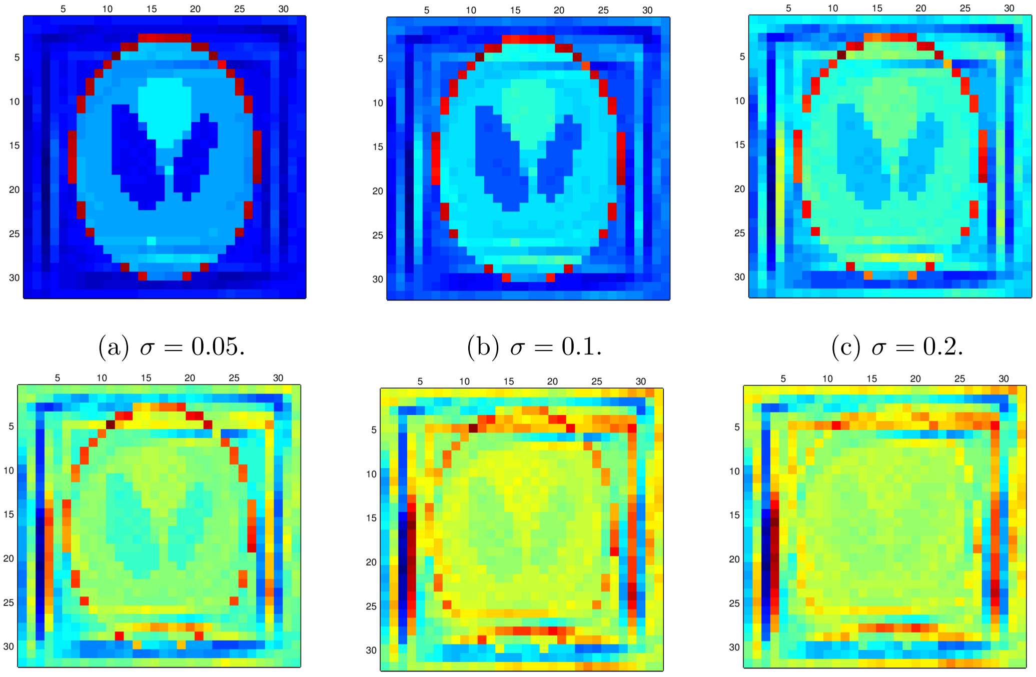
Listing 4: Plotting graphs for Figure 5.

```
[A,data]=siddonS(Image,10,30); %Specifying nProj=10, nAngles=30.
S=svd(full(A)); %Ouputs singluar values of A in full form (non-sparse).
figure; plot(S)
figure; loglog(S)
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'linewidth',2,'FontSize',12);%Edit figure styling.
```

2 Reconstruction of Images and Investigating Errors

Simply inverting the linear system (1) to give a solution of the form $x = A^{-1}b$ is generally not viable since A is generally not square. Data can be measured several times to allow for this; in an overdetermined system, matrix analysis gives a least squares solution to such a linear system of equations. This solution minimises the sum of squares of errors made as a result of each equation. The Octave backslash operator, `x=A\b`, where `A` and `b` must have the same number of rows, returns the least squares solution to (1).

In tomography as well as many other inverse problems, solutions obtained in such a manner are highly unstable when errors are present (which tomography measurements are particularly prone to). To simulate this more complex, realistic scenario, pseudo-random errors with standard deviation σ will be added to `b`, with varying values of σ, depicted by the following figure.



(a) $\sigma = 0.05$.

(b) $\sigma = 0.1$.

(c) $\sigma = 0.2$.

(d) $\sigma = 0.4$.

(e) $\sigma = 0.8$.

(f) $\sigma = 1.6$.

Figure 6: Least squares solution reconstructed image, with errors of standard deviation σ.

In the simplistic instance in which `b` has no added error ($\sigma = 0$), the image of the solution is an exact match to Figure 1. As expected, Figure 6 shows for relatively small standard deviation valuable detail is quickly lost, before the data becomes inapt of any use at around $\sigma = 1.6$ - as explained by the instability of solutions in this form.

Listing 5: Creating reconstructed images in Figure 6.

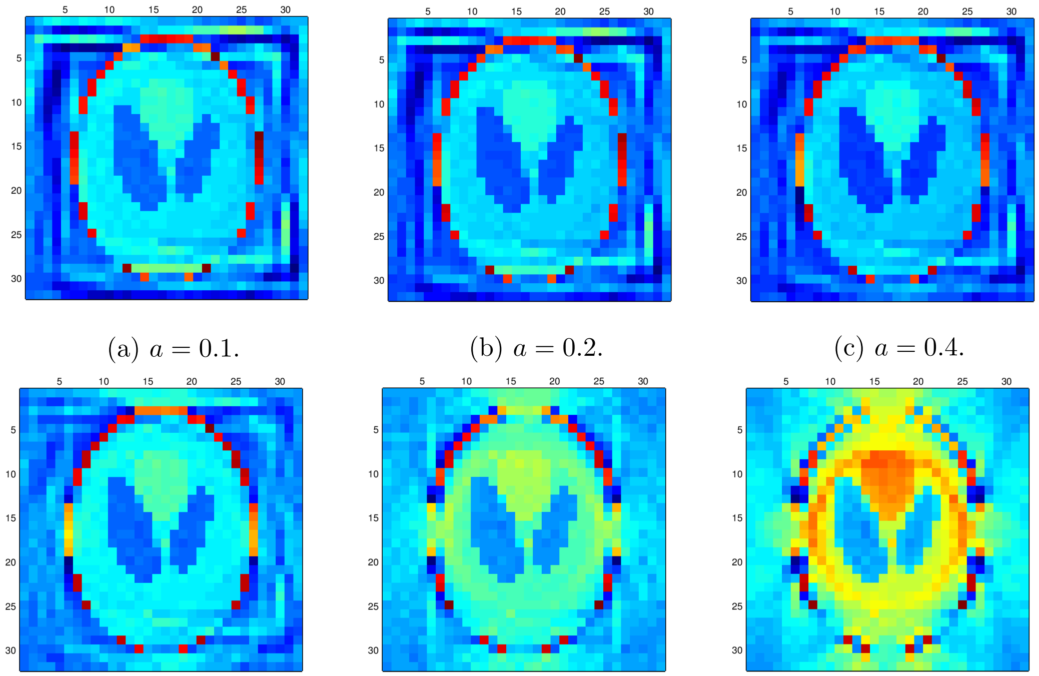
```
x=A\b; %Least squares solution to linear system of equations Ax=b.
figure; imagesc(reshape(x,32,32));%Reshape reutrns x from vector to matrix form.
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'XAxisLocation','top','linewidth',2,'FontSize',12); %Edit figure styling.

noise=randn(size(b)); %Keeping the added error constant up to a factor sigma,
                      %allowing the image plots to be compared.
for sigma=[0.05 0.1 0.2 0.4 0.8 1.6] %Various error standard deviation values.
xnoise=A\(b+sigma*noise); %Least squares solution to noisy linear system.
figure; imagesc(reshape(xnoise),32,32))
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'XAxisLocation','top','linewidth',2,'FontSize',12);%Edit figure styling.
end
```

3 Tikhonov Regularisation

Tikhonov regularisation is a simple, and computationally inexpensive method to stabilise such a solution [5]. Ordinarily, least squares solutions seek to minimize the sum of squared errors, or residuals, which can be written as $||Ax - b||^2$, where $|| \cdot ||$ is the Euclidean norm. A regularisation term can be introduced, $||Ax - b||^2 + ||\Gamma x||^2$, to give preference to a solution of the minimisation with desirable properties for an appropriate choice of Γ, the Tikhonov matrix. In this case a preference to solutions with smaller norms gives the choice of $\Gamma = aI$, where I is the identity matrix and a is known as the Tikhonov factor. Again a least squares solution will be obtained and is of the form,

$$x = (A^T A + aI) \backslash b.$$

The Tikhonov factor depends on the units and size of entries in A and the amount of noise in b, with greater noise requiring a greater Tikhonov factor. This compromises between extreme solution values and fitting the given data. To implement Tikhonov regularisation, a psuedo-random error with standard deviation of $\sigma = 0.2$ will be chosen, since this size of error has not become large enough to dominate the data, as shown by Figure 6(c).



(a) $a = 0.1$.

(b) $a = 0.2$.

(c) $a = 0.4$.

(d) $a = 0.8$.

(e) $a = 3.2$.

(f) $a = 12.8$.

Figure 7: Tikhonov regularisation with Tikhonov factor a.

Generally the optimal Tikhonov factor is unknown and in practise, it is found experimentally through various approaches such as Bayesian interpretation or cross-validation [6]. It can been seen from Figure 7 that a Tikhonov factor of approximately $a = 0.4$ yields the greatest stabilisation of the solution, and as the Tikhonov factor exceeds $a = 10$ the reconstructed image becomes distorted and loses valuable data.

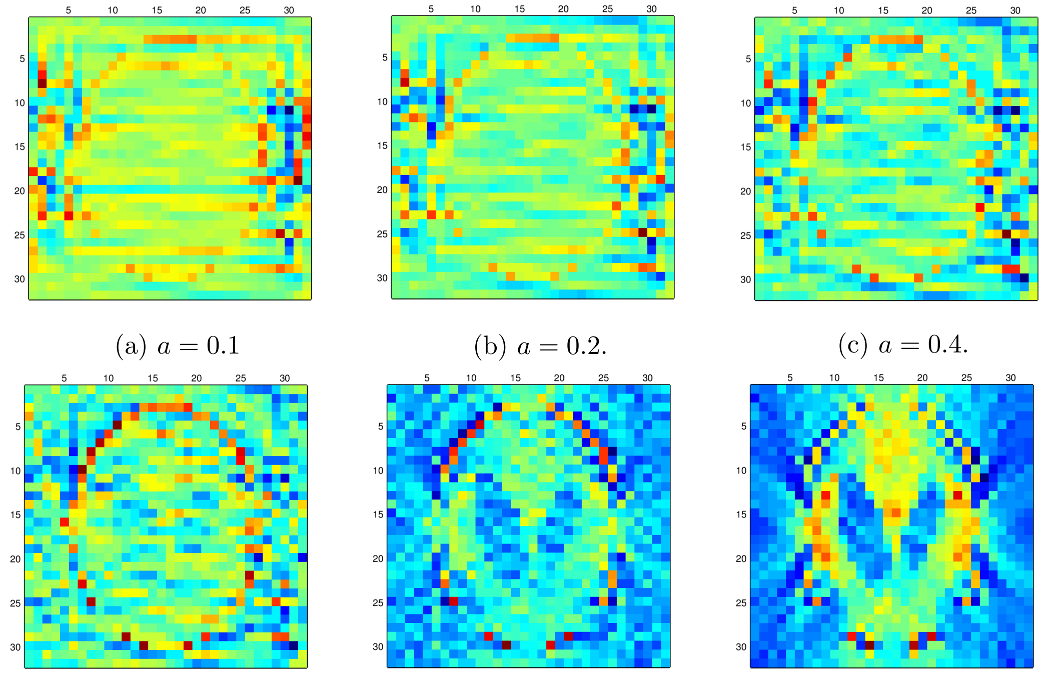
When attempting the same regularisation with psuedo-random errors of standard devation $\sigma = 0.8$, the reconstructed image does no stabilise to a useul state and any σ value greater than this produces greater distortion in the reconstructed image.

Listing 6: Implementing Tikhonov regularisation and creating reconstructed images in Figure 7.

```
bnoise=b+0.2*randn(size(b)); %Adding a fixed noise to b.
for a=[0.1 0.2 0.4 0.8 3.2 12.8]; %Various values of Tikhonov factor.
Atik=A'*A+a^2*eye(size(A,2)); %Implementing Tikhonov regularisation.
btik=A'*bnoise; %Implementing Tikhonov regularisation.
figure; imagesc(reshape(Atik\btik,32,32)) %Image reconstruction.
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'XAxisLocation','top','linewidth',2,'FontSize',12);%Edit figure styling.
end
```

3.1 Ill Fitting Test Objects and Tikhonov Regularisation

Tikhonov regularisation implemented on a test object with missing data, such as `Adel`, does not stabilise the reconstructed image as effectively, since missing data can be is misinterpreted as an error in the solution.



(a) $a = 0.1$

(b) $a = 0.2$.

(c) $a = 0.4$.

(d) $a = 0.8$.

(e) $a = 3.2$.

(f) $a = 12.8$.

Figure 8: Tikhonov regularisation with Tikhonov factor a, for A with deleted rows.

As seen in Figure 12, a Tikhonov factor of 12.8 yields a reconstructed image similar to Figure 7(f), however this reconstruction is distorted and has lost significant data.

Listing 7: Implementing Tikhonov regularisation for `Adel` and reconstructing images for Figure 12.

```matlab
bdelnoise=bdel+0.8*randn(size(bdel)); %Adding a fixed noise to bdel.
for a=[0.1 0.2 0.4 0.8 3.2 12.8];
Adeltik=Adel'*Adel + a^2*eye(size(Adel,2)); %Implementing Tikhonov regularisation.
bdeltik=Adel'*bdelnoise; %Implementing Tikhonov regularisation.
figure; imagesc(reshape(Adeltik\bdeltik,32,32))
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'XAxisLocation','top','linewidth',2,'FontSize',12);%Edit figure styling.
end
```

4 Application of Tomography to Materials Science

In materials science and engineering X-ray projections (particularly X-ray diffractions) are performed to investigate a large variety of intricate material properties. If samples are at least 10 times wider than they are tall, X-rays can only be taken 35 degrees either side of the vertical direction; tomography can be implemented to examine such a problem.

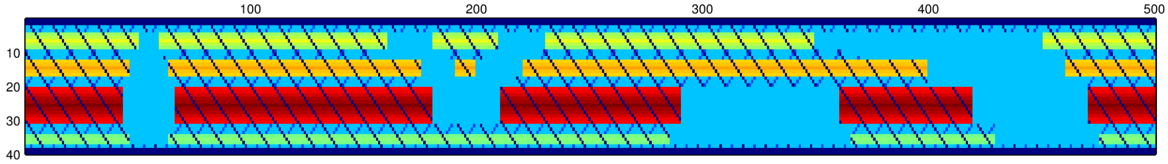


Figure 9: Sample of mesh tube with fragmented core.

To explore such a problem, consider as the test sample a mesh tube with a fragmented core - as depicted in Figure 9.

Listing 8: Creating mesh tube with fragmented core sample and producing Figure 9.

```
sample=zeros(40,500); %base of sample.
for i=1:40 %This draws the bulk of the mesh grid.
for j=0:10:460
sample(i,501-(i+j))=-20; %Background mesh grid.
sample(i,i+j)=-50; %Foreground mesh grid.
end
end %Following 3 for loops draw the remaining mesh grid in each corner.
for i=1:30
sample(i,31-i)=-20; %Background mesh grid.
sample(i+10,501-i)=-20; %Background mesh grid.
sample(i+10,i)=-50; %Foreground mesh grid.
sample(i,i+470)=-50; %Foreground mesh grid.
end
for i=1:20
sample(i,21-i)=-20; %Background mesh grid.
sample(i+20,501-i)=-20; %Background mesh grid.
sample(i+20,i)=-50; %Foreground mesh grid.
sample(i,i+480)=-50; %Foreground mesh grid.
end
for i=1:10
sample(i,11-i)=-20; %Background mesh grid.
sample(i+30,501-i)=-20; %Background mesh grid.
sample(i+30,i)=-50; %Foreground mesh grid.
sample(i,i+490)=-50; %Foreground mesh grid.
end

for i=1:2 %Top and bottom tube borders.
sample(i,:)=-50*ones(1,500);
sample(38+i,:)=sample(3-i,:);
end
for i=5:7 %%Core tube 1, rows 5 to 9.
```

```
sample(i,:)=6*i*ones(1,500); %Top half of core.
end
for i=1:2 %Bottom half of core.
sample(7+i,:)=sample(7-i,:);
end

for i=13:15 %Core tube 2, rows 13 to 17.
sample(i,:)=4*i*ones(1,500); %Top half of core.
end
for i=1:2 %Bottom half of core.
sample(15+i,:)=sample(15-i,:);
end

for i=21:26 %Core tube 3, rows 21 to 31.
sample(i,:)=4*i*ones(1,500); %Top half of core.
end
for i=1:5 %Bottom half of core.
sample(26+i,:)=sample(26-i,:);
end

for i=35:36 %Core tube 4, rows 35 to 37.
sample(i,:)=0.75*i*ones(1,500); %Top half of core.
end
for i=1:1 %Bottom half of core.
sample(36+i,:)=sample(36-i,:);
end
%Fragmentions the created by reassigning entries of sample to be 0.
```

By reapplying the Octave code in Listings 1, 2 with `Image=sample`, and defining optional parameters of `siddonS` as `nProj=35` and `nAngle=70`, the corresponding linear systems of equations (1) was established and Figures 10 and 11 are generated. For `siddonS` to allow the value of `nAngle=70`, line `36` of `siddonS` was edited, by removing the function call `ceil` from `Ang_step = ceil(180/nAngles)`. This is to allow for a non-integer when dividing value for `Ang_step`, which is required later in the algorithm.

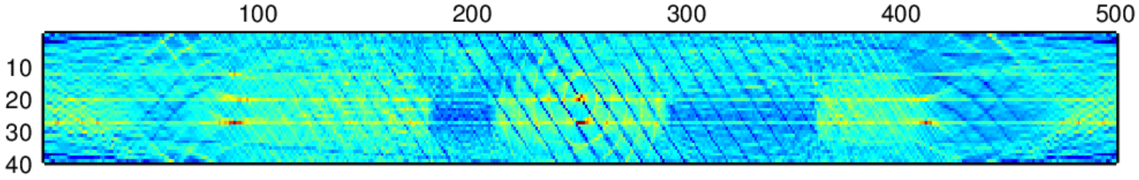
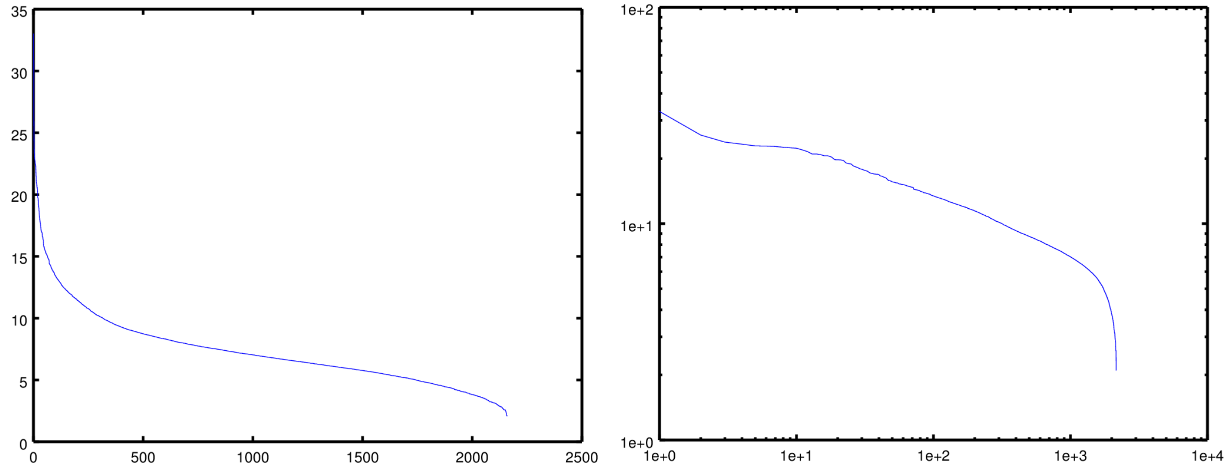


Figure 10: Reconstructed sample image.

Figure 10 shows an unsuccessful attempted at image reconstruction; it is likely the mesh grating of the tube will be read as random noise, meaning the reconstructed image will not stabilise effectively. To investigate this, as in Section 1, singular values are plotted to determine the effective rank of the rank deficient matrix A.

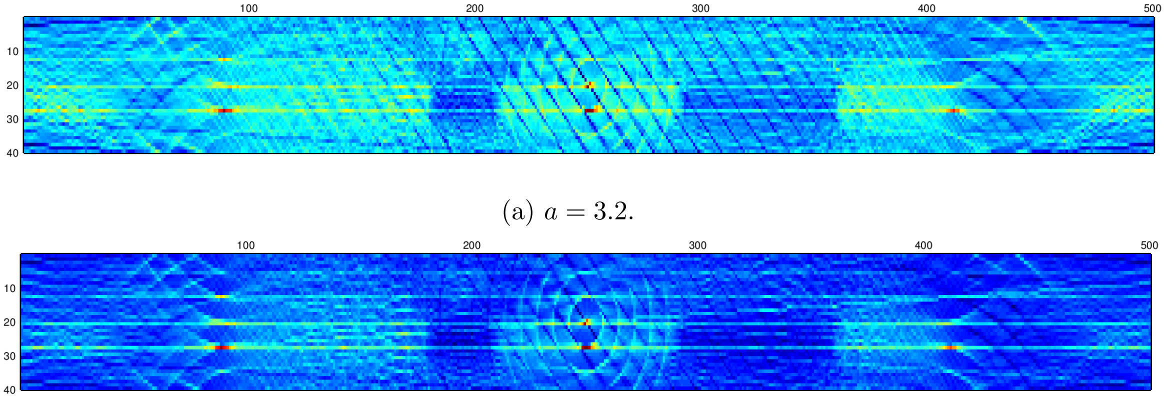


(a) Plot of singular values.

(b) Natural log-log plot of singular values.

Figure 11: Plot and natural log-log plot of singular values for materials sample.

Similarly to previous cases of effective rank approximation, insignificant singular values cannot be deduced from Figure 11(a). There is a sharp drop in singular values of the natural log-log plot at approximately 1600, meaning the matrix is extremely rank deficient, giving further indication that reconstructing the image will not be effective. Tikhonov regularisation will be applied with an added pseudo-random error of standard deviation $\sigma = 0.1$ in a hope to eliminate noise and as well as stabilise the already distorted reconstructed image.



(a) $a = 3.2$.

(b) $a = 12.8$.

Figure 12: Tikhonov regularisation for materials science sample.

Tikhonov regularisation failed to recover any significant data for all Tikhonov factor values, as evidenced by Figure 12. This is largely due to the complexity of the sample and indicates a limitation of Tikhonov regularisation when applied to such a rank deficient system.

Listing 9: Implementing Tikhonov regularisation for materials science sample.

```
bnoise=b+0.1*randn(size(b)); %Adding a fixed noise to b.
for a=[0.8 3.2 12.8];  %Various Tikhonov factor values.
Atik=A'*A+a^2*eye(size(A,2)); %Implementing Tikhonov regularisation.
btik=A'*bnoise; %Implementing Tikhonov regularisation.
xtik=Atik\btik; %Least squares solution to Tikhonov regularisation system.
figure; imagesc(reshape(xtik,40,500))
set(findall(gcf,'type','text'),'FontSize',12,'FontName','CMU Serif');
set(gca,'XAxisLocation','top','linewidth',2,'FontSize',10);%Edit figure styling.
set(gca,'DataAspectRatio',[1.5 1 1]); %Stetches y-axis for easier viewing.
end
```

A potential solution to recovering a sufficiently clear image would be to implement a more sophisticated image regularisation algorithm based on an iterative method, such as Prior-Image Induced Nonlocal Regularisation or block Algebraic Reconstruction.

References

[1] G. Herman, (2009). *Fundamentals of Computerized Tomography: Image Reconstruction from Projections (2nd ed.)*, Springer, ISBN 978-1-85233-617-2.

[2] M. Bertero, P. Boccacci, (1998), *Introduction to Inverse Problems and Imaging*, Institute of Physics Publishing, ISBN 0750304391.

[3] L. Shepp, B. Logan, (1974). *The Fourier Reconstruction of a Head Section*. IEEE Transactions on Nuclear Science. NS-21 (3): 2143.

[4] N.Trefethen, D. Bau III, (1997). *Numerical linear algebra*. Philadelphia: Society for Industrial and Applied Mathematics. ISBN 978-0-89871-361-9.

[5] A. Tikhonov, (1943). *On the Stability of Inverse Problems*. Doklady Akademii Nauk SSSR 39 (5): 195198.

[6] A. Tarantola, (2005). *Inverse Problem Theory*, Society for Industrial and Applied Mathematics, ISBN 0-89871-572-5.