Technische Universiteit Delft
Faculteit Elektrotechniek, Wiskunde en Informatica
Delft Institute of Applied Mathematics

Total least squares method for the purpose of noise reduction in low-cost MRI images

(Total least squares methode ten behoeve van ruisvermindering in afbeeldingen van een goedkopere MRI scanner)

Verslag ten behoeve van het
Delft Institute of Applied Mathematics
als onderdeel ter verkrijging

van de graad van

BACHELOR OF SCIENCE
in
TECHNISCHE WISKUNDE

door

Manon van Zon

Delft, Nederland
Augustus 2019

BSc verslag TECHNISCHE WISKUNDE

Total least squares method for the purpose of noise reduction in low-cost MRI images

(Total least squares methode ten behoeve van ruisvermindering in afbeeldingen van een goedkopere MRI scanner)

Manon van Zon

Technische Universiteit Delft

Begeleider

Dr.ir. M.B. van Gijzen

Overige commissieleden

Prof.dr.ir. A.W. Heemink

Dr.ir. M. Keijzer

Augustus, 2019

Delft

Abstract

MRI images are very useful for detecting diseases, as well as for the treatment of diseases. However, MRI scanners are usually too expensive for developing countries to purchase and maintain. Therefore, a less expensive scanner is being developed at Delft University of Technology and Leiden University Medical Center. Unfortunately, the images which are generated by this low-cost MRI scanner are contaminated by noise. The MRI images are determined by solving a system of equations of the form $A\mathbf{x} = \mathbf{y}$, where $\mathbf{x}$ is the unknown image. As this system is perturbed, regression methods can be applied in order to find the best approximate value of $\mathbf{x}$. The ordinary least squares method solves this system for perturbations in $\mathbf{y}$. Whereas in the MRI case, it appears that both A and $\mathbf{y}$ are perturbed. The total least squares method is often used in order to solve these kind of perturbed systems of equations. The aim of this thesis is to investigate the abilities of this method for the purpose of noise reduction in MRI images. It appears that the total least squares method in combination with regularization operators is able to cancel out a certain amount of noise from the images. However, no significant advantages compared to the ordinary least squares method are found.

Preface

In the past couple of months, I have been working on this thesis in order to fulfil the requirements of the Bachelor programme Applied Mathematics at Delft University of Technology. During my study, I gained a large interest into providing technical solutions for medical issues. The underlying project of this thesis aims to save children who are suffering from hydrocephalus. Because of this, I enjoyed it very much to be to able to work on this thesis. Furthermore, working on this thesis has not only shown me what admirable things can be derived from mathematics. It also showed me a collaboration between the fields I am most interested in. Therefore, I would like to thank my supervisor Martin van Gijzen for allowing me to work on this project and the time he made available for me. Furthermore, I would like to thank Prof.dr.ir. A.W. Heemink and Dr.ir. M. Keijzer for their time and willingness to be a committee member.

Manon van Zon,
Delft, August 2019

Contents

1 Introduction

It is impossible to imagine our health care without MRI scanners, as they are a very effective tool for diagnosing lots of different diseases. Unfortunately, MRI scanners are very expensive and as a consequence, MRI diagnostics are unavailable for lots of people in developing countries. In order to make a difference, several researchers at Delft University of Technology and Leiden University Medical Center are developing a new kind of MRI scanner, as shown in Figure 1. This new scanner should be affordable and maintainable in third world countries. The ultimate goal is to make the MRI scanner usable for treatment of children with hydrocephalus (also known as water on the brain), from which numerous are suffering in developing countries. However, some problems interfere with the proper use of the low-cost MRI scanner. One of the main issues is the large amount of noise that contaminates the images. In general, regression methods are used to reduce noise. Therefore, in this thesis we look into a regression method called total least squares. The aim is to find out if this method can improve the quality of images generated by the low-cost MRI scanner, this gives rise to the followings research questions:

- How can the total least squares solution be computed?

- Is the total least squares method of interest to our problem?

The remainder of this thesis is structured as follows. In Section 2 we derive the mathematical model to describe how MRI images are generated. To this end, we present some physical background information on the workings of an MRI. As the images which result from the mathematical model are perturbed, regression methods are discussed in Section 3 in order to reduce noise in general. In Section 4 we explain how the regression methods can be applied to MRI images and the obtained results are shown in Section 5. Finally, in Section 6 we will formulate answers to the research questions and give recommendations for future research.

Figure 1: The low-cost MRI scanner. (Photo by F. Auperle.)

2 MRI

The aim of this section is to derive the mathematical model to describe how MRI images are generated. Section 2.1 provides some physical background that is needed to understand the functioning of an MRI scanner. Next, in Section 2.2, we describe how the introduced physical properties are applied in MRI. In Section 2.3 we show how the functioning of the MRI scanner leads to a measurable signal. This signal can be used to generate a 2D image of a slice of the object to be imaged, which we explain in Section 2.4. Finally, in Section 2.5 we derive the mathematical model from the obtained signal which is used for image reconstruction. This Section is based on [8] and [3].

2.1 Background theory

Magnetic fields and magnetisation lay at the foundation of the functioning of an MRI scanner. First, we introduce magnetic fields. A magnetic field is an invisible physical phenomenon which is caused by charged particles in motion. For example, by electricity flowing through a coil or by a magnet. Furthermore, it is important to note that a magnetic field exerts a measurable force on charged particles, which we will use later in this section. Magnetic fields are often denoted by a vector $\vec{B}$ which describes its direction and magnitude in Tesla (T). Once a charged particle enters a magnetic field it becomes magnetised, i.e., its behaviour is changed by the magnetic field. Different particles react differently to a magnetic field. This behaviour is described by magnetisation $\vec{M}$ in $(\mathrm{A{\cdot}m^{-1}})$.

Next, we describe an important property of atoms. Atoms consist of a nucleus and electrons orbiting around the nucleus. The nucleus of an atom consists of neutrons and at least one proton, whose total number is approximately the atomic weight. Atoms with odd atomic weight are known to possess a non-zero spin, which can be interpreted as the rotation of a particle about its own axis. If a spinning particle is partially charged, it will generate its own magnetic field.

Once a spinning particle enters an external magnetic field, which is the case during an MRI scan, several things happen. First, the spin aligns either parallel or anti-parallel with the direction of the external magnetic field. Each spinning particle is most likely to align parallel to the field, since this is the lower energy state. Secondly, the spinning particle starts precessing about the axis of the external magnetic field, as shown in Figure 2. Precession is the change in the direction of the axis of a spinning particle.

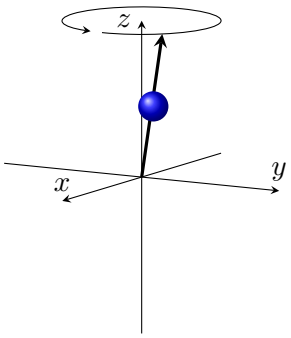


Figure 2: Precession of a spinning particle around the z-axis.

The frequency ω (in Hz) of this precession is proportional to the strength of the magnetic field. This relation is described by the Larmor frequency

$$\omega = \gamma|\vec{B}|, \tag{2.1}$$

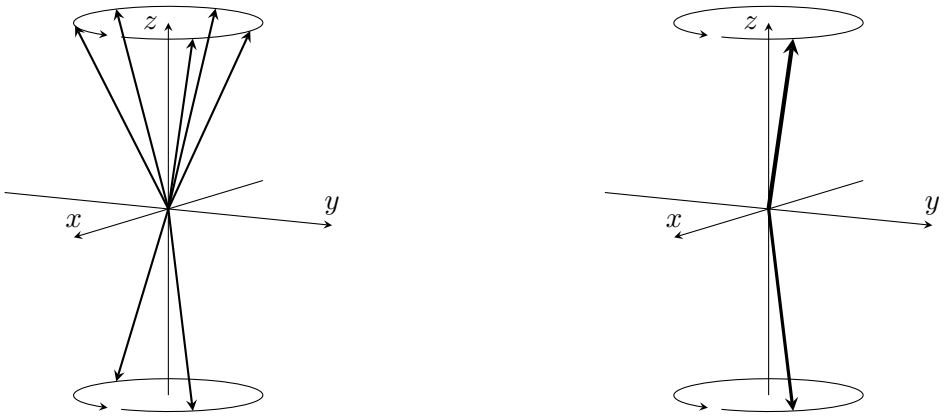
where γ is the gyromagnetic ratio in rad $\mathrm{s}^{-1}\mathrm{T}^{-1}$, which is given for different particles.

2.2 Theory behind MRI scanners

Humans consist of numerous hydrogen atoms, these atoms are very useful for generating MRI images. The nucleus of a hydrogen atom consists of a single proton and hence has odd atomic weight. Therefore the hydrogen atom, also referred to as proton, possesses a spin. Since protons are charged and spinning particles, they each generate a microscopic magnetic field. As the direction of these magnetic fields is random without the presence of an external magnetic field, the net magnetization inside the human body is approximately zero. However, this changes when someone is subjected to a magnetic field, which is the case during an MRI scan. MRI scanners are designed to generate a strong static homogeneous magnetic field $\vec{B}_0(\mathbf{r})$, where $\mathbf{r} = (x, y, z)$. This field is applied in the z-direction, meaning $\vec{B}_0(\mathbf{r}) = B_0\vec{k}$, with $\vec{k}$ the unit vector in the z-direction. Due to this applied field, the protons inside the body align either parallel or anti-parallel to the field and a net magnetization arises. The net magnetization is in the z-direction, as can be observed from Figure 3a and is called the longitudinal magnetization. Furthermore, the protons will precess about the field's axis at the following Larmor frequency

$$\omega_0 = \gamma B_0. \tag{2.2}$$

Note that the protons do not precess in phase relative to each other.



(a) Spins are precessing out of phase, hence only a net longitudinal magnetization exists.

(b) Spins are precessing in phase, resulting in a net transverse magnetization.

Figure 3: The direction and phase of spins result in a net magnetization.

It is in the interest of MRI to also generate a transverse magnetization, i.e., in the (x, y) plane. In particular, changes in magnetization are used to generate MRI images. The transverse magnetization is obtained using a Radio Frequency (RF) system. The RF system generates another magnetic field, which we call $\vec{B}_1(\mathbf{r}, t)$. This field is switched on and off during an MRI scan and therefore depends on the time t. The RF system consists of a coil that emits waves in the radio frequency range. The energy provided by these waves, makes the protons precess in phase. Consequently, a transverse magnetization arises, as can be noted from Figure 3b. Furthermore, the radio frequency waves cause some of the spinning protons to flip into the higher energy state. As a result, the longitudinal magnetization decays. After this so-called RF excitation period, the longitudinal magnetization will regrow, whereas the transverse magnetization dissolves. Since this changing magnetization is important for generating MRI images, we will observe the changing behaviour of

the magnetization $\vec{M}(\mathbf{r},t)$ that is caused by adding the $\vec{B}_1(\mathbf{r},t)$ field. Let

$$\vec{M}(\mathbf{r},t) = M_x(\mathbf{r},t)\vec{i} + M_y(\mathbf{r},t)\vec{j} + M_z(\mathbf{r},t)\vec{k}, \tag{2.3}$$

where $\vec{i},\vec{j},\vec{k}$ denote the unit vectors in the x,y and z direction, respectively. Then the transverse magnetization can be defined as follows

$$M_{xy}(\mathbf{r},t) = M_x(\mathbf{r},t) + i \cdot M_y(\mathbf{r},t), \tag{2.4}$$

where i is the complex constant. The change of magnetization in time in the presence of an applied magnetic field $\vec{B}(\mathbf{r},t)$ is described by the Bloch equations. The Bloch equation for MRI is of the following form

$$\frac{\mathrm{d}\vec{M}(\mathbf{r},t)}{\mathrm{d}t} = \vec{M}(\mathbf{r},t) \times \gamma\vec{B}(\mathbf{r},t) - \frac{M_x(\mathbf{r},t)\vec{i} + M_y(\mathbf{r},t)\vec{j}}{T_2} - \frac{(M_z(\mathbf{r},t) - M_0)\vec{k}}{T_1}, \tag{2.5}$$

where M_0 is the equilibrium value of $\vec{M}(\mathbf{r},t)$ when only the magnetic field $\vec{B_0}(\mathbf{r})$ is present. T_1 and T_2 are relaxation times, i.e., T_1 represents the time in seconds that is needed to recover 63% of the longitudinal magnetization. Whereas T_2 is the time in seconds that is needed for a 63% decay of the transverse magnetization. Once the RF excitation period is completed, the precessing protons will gradually start precessing out of phase again. As a result the transverse magnetization decreases exponentially with a factor e^{-t/T_2}. Let $t = 0$ be the time when the RF excitation period is ended, then the transverse magnetization for $t > 0$ is given by

$$M_{xy}(\mathbf{r},t) = M_{xy}(\mathbf{r},0)e^{-t/T_2(\mathbf{r})}e^{-i(\gamma B_z(\mathbf{r})+\omega_0)t}. \tag{2.6}$$

Here $M_{xy}(\mathbf{r},0)$ is what forms the MRI image and $B_z(\mathbf{r})$ denotes the longitudinal component of the present magnetic field $\vec{B}$. A detailed description of this derivation is included in Appendix A.1.

2.3 Signal obtained by MRI scanners

Next, one can use this change in magnetization to convert it into a measurable signal. This conversion is carried out by another coil of the RF system. According to Faraday's law of induction, a changing magnetic field will create an electromotive force in a nearby coil. From this law it follows that the induced voltage $V(t)$ in the RF coil is given by

$$V(t) = -\frac{\partial}{\partial t}\int_{\text{object}} c(\mathbf{r})\left(\vec{B}(\mathbf{r}) \cdot \vec{M}(\mathbf{r},t)\right)\,\mathrm{d}\mathbf{r}, \tag{2.7}$$

with $c(\mathbf{r})$ the coil response pattern. $\vec{B}(\mathbf{r}) = \vec{B}_x(\mathbf{r})\vec{i} + \vec{B}_y(\mathbf{r})\vec{j} + \vec{B}_z(\mathbf{r})\vec{k}$ denotes the magnitude and direction of the magnetic field at position $\mathbf{r}$. As a result, the obtained signal is as follows

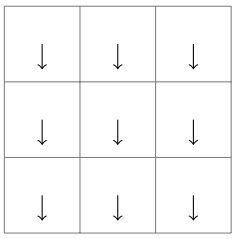
$$S(t) = \int_{\text{object}} c(\mathbf{r})\overline{\vec{B}_{xy}(\mathbf{r})}M_{xy}(\mathbf{r},0)e^{-t/T_2(\mathbf{r})}e^{-i\gamma B_z(\mathbf{r})t}\,\mathrm{d}\mathbf{r}. \tag{2.8}$$

Here, $\vec{B}_{xy}(\mathbf{r})$ is the magnetic field in the transverse plane. For a detailed description of this derivation, see Appendix A.2.

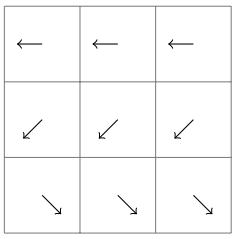
2.4 Generating 2D images

In order to generate 2D images from the received signals, MRI scanners contain a gradient system. This system consists of three orthogonal coils, which are used to generate a magnetic field whose z component varies linearly at different positions. When we want to create an image in the (x,y)

plane, we first have to select a slice of the object to be imaged. This is carried out using the gradient coil in the z-direction. In this selected slice, the experienced gradient field is homogeneous. Since frequency is proportional to the strength of the magnetic field, all spins in the selected slice precess at the same rate, as in Figure 4a. This makes it impossible to distinguish their signals. However, the gradients in the other directions are used to solve this problem. While switching on the gradient field in the x-direction, a linearly varying magnetic field from left to right, as in Figure 4, arises. As a result, the spins that experience a stronger magnetic field will precess at a higher frequency. Using the gradient field that is applied in the y-direction, a linearly varying magnetic field from bottom to top arises. Hence, it causes spins at different rows of the slice, as sketched in Figure 4, to precess at a different rate. Opposed to the x-gradient, the y-gradient is quickly turned off. As a result, the y-gradient makes spins in distinct rows to precess out of phase, which is shown in Figure 4b. In conclusion, all spins in the selected slice are precessing at different frequencies and have phase shifts. Then, methods known as phase and frequency encoding can be used to localize the signals.



(a) The direction of $M_{xy}(\mathbf{r}, t)$ caused by the spins on different (x, y) positions in the selected slice.



(b) The influence on the left image after the y-gradient has quickly been switched on and off.

Figure 4: The influence of the y-gradient on a selected slice of the object to be imaged.

2.5 Mathematical model

From the signal obtained in Equation (2.8), the goal is to estimate the unknown continuous function $x(\mathbf{r}) := M_{xy}(\mathbf{r}, 0)$, which represents the MRI image. We can approximate $M_{xy}(\mathbf{r}, 0)$ using a finite series expansion. Thus,

$$x(\mathbf{r}) = \sum_{j=1}^{N} x_j(\mathbf{r}) b(\mathbf{r} - \mathbf{r}_j), \tag{2.9}$$

where b is the rect$(\cdot)$ basis function, $\mathbf{r}_j$ denotes the center of the translated jth basis function and N is the number of pixels. Substituting Equation (2.9) into Equation (2.8), we obtain the following equation for every ith sample of the signal at time t_i

$$S(t_i) = \sum_{j=1}^{N} a_{ij} x_j(\mathbf{r}), \tag{2.10}$$

for $i \geq 1$ and

$$a_{ij} = \int_{\text{object}} b(\mathbf{r} - \mathbf{r}_j) c(\mathbf{r}) \overline{\vec{B}_{xy}(\mathbf{r})} e^{-t_i / T_2^*(\mathbf{r})} e^{-i\gamma B_z(\mathbf{r}) t_i} \, \mathrm{d}\mathbf{r}. \tag{2.11}$$

Here, $T_2^*(\mathbf{r})$ represents T_2 in the vicinity of field inhomogeneities. Since basis functions are usually highly localized, we can use 'center of pixel' approximations, hence a_{ij} can be denoted as follows

$$a_{ij} = c(\mathbf{r}_j) \overline{\vec{B}_{xy}(\mathbf{r}_j)} e^{-t_i / T_2^*(\mathbf{r}_j)} e^{-i\gamma B_z(\mathbf{r}_j) t_i}. \tag{2.12}$$

The measured signals of MRI scans, as described by Equation (2.10), are contaminated by noise. Thus the obtained signal y_i at the ith sample is given by

$$y_i = S(t_i) + g_i. \tag{2.13}$$

Here, g_i represents the Gaussian noise which is present at the ith sample. This leads to the following perturbed system of equations

$$\mathbf{y} = A\mathbf{x} + \mathbf{g}, \tag{2.14}$$

where the matrix entries of A are given by Equation (2.12). In addition, we assume the coil response pattern to be constant for a particular pixel. Also, the magnetic field is considered to be homogeneous for a certain pixel. Furthermore, note that T_2^*, which is in the order of thousands of milliseconds, is very long compared to t_i. Hence, the most interesting part of the MRI signal is given by

$$a_{ij} \propto e^{-i\gamma B_z(\mathbf{r}_j)t_i}. \tag{2.15}$$

3 Regression analysis

In the previous section we obtained the following system of equations

$$A\mathbf{x} = \mathbf{y}, \tag{3.1}$$

with $A \in \mathbb{C}^{m\times n},\ m > n$ the coefficient matrix, $\mathbf{x} \in \mathbb{C}^n$ is the vector that contains the unknown pixel values and $\mathbf{y} \in \mathbb{C}^m$ contains the measured signals. In practice, it is difficult to find $\mathbf{x}$ as the system is perturbed, i.e., A and $\mathbf{y}$ include noise. As a result, it is no longer possible to determine $\mathbf{x}$ with certainty, hence we approximate $\mathbf{x}$. In order to find an approximation, regression methods can be applied. In Section 3.1 we briefly discuss the ordinary least squares method. Next, in Section 3.2 we introduce the total least squares method and show how it can be used to compute a solution to a system as described above. In Section 3.3 it is shown how the power method provides an alternative way of computing the total least squares solution. Finally, in Sections 3.4 and 3.5 some examples about the difference between the least squares and total least squares methods and their performances are shown. Sections 3.1, 3.2 and 3.4 are based on [11] and [9].

3.1 Least squares

The least squares (LS) method assumes that perturbations in a system as described by Equation (3.1) are caused by errors in $\mathbf{y}$ only. Hence, the variables in the coefficient matrix A are considered to be free of error. Assuming that A has full column rank, i.e., all columns of A are linearly independent, the unique least squares solution $\mathbf{x}_{\text{ls}}$ is given by

$$\mathbf{x}_{\text{ls}} = (A^*A)^{-1}A^*\mathbf{y}, \tag{3.2}$$

where A^* denotes the conjugate transpose of A. The LS method can be reformulated as it aims to determine a correction $\Delta\mathbf{y}$ to $\mathbf{y}$ such that the system $A\mathbf{x} = \mathbf{y} + \Delta\mathbf{y}$ has an exact solution and $\Delta\mathbf{y}$ is as small as possible. Precisely, the least squares solution is obtained by minimizing the following problem

$$\min_{\Delta\mathbf{y}\in\mathbb{C}^m} ||\Delta\mathbf{y}||_2, \quad \text{subject to } \mathbf{y} + \Delta\mathbf{y} \in R(A). \tag{3.3}$$

Here, $||\cdot||_2$ denotes the 2 norm and $R(A)$ is the column space of A. When a minimizing $\Delta\mathbf{y}$ is found, every $\mathbf{x}$ that satisfies $A\mathbf{x} = \mathbf{y} + \Delta\mathbf{y}$ is called an LS solution.

3.2 Total least squares

Opposed to LS, the total least squares (TLS) method assumes that besides $\mathbf{y}$, the matrix A is perturbed as well. Both the errors on A and $\mathbf{y}$ are assumed to be identically distributed. Hence, it treats errors in both A and $\mathbf{y}$ equally. In order to find the best approximate solution to Equation (3.1), TLS searches for ΔA and $\Delta\mathbf{y}$ such that $(A + \Delta A)\mathbf{x} = \mathbf{y} + \Delta\mathbf{y}$ has an exact solution. TLS optimizes the following problem

$$\min_{\Delta A\in\mathbb{C}^{m\times n},\ \Delta\mathbf{y}\in\mathbb{C}^m} ||\ [\Delta A \ \ \Delta\mathbf{y}]\ ||_F \quad \text{subject to } (\mathbf{y} + \Delta\mathbf{y}) \in R(A + \Delta A). \tag{3.4}$$

Here, $||A||_F$ is the Frobenius norm of an arbitrary matrix A, which is defined as $\sqrt{\text{trace}(A^*A)}$, where $\text{trace}(A^*A)$ is the sum of the elements on the main diagonal of A^*A. Once minimizing ΔA and $\Delta\mathbf{y}$ are found, then any $\mathbf{x}$ satisfying $(A + \Delta A)\mathbf{x} = \mathbf{y} + \Delta\mathbf{y}$ is called a TLS solution. In the remainder of this subsection, we derive a basic TLS solution to the perturbed system. In order to do so, we use the singular value decomposition.

3.2.1 Singular value decomposition

The singular value decomposition (SVD) [4] is often used to determine a total least squares solution. One of the advantages of this decomposing method, is that it can be applied to any $m \times n$ matrix A. However, computing the SVD of large matrices becomes computationally expensive. The SVD factorizes the matrix A of rank r into the following product of matrices

$$A = U\Sigma V^*, \qquad \text{with } \Sigma = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}.$$

Here, $U = [u_1, ..., u_m] \in \mathbb{C}^{m \times m}$ and $V^* = [v_1^*, ..., v_n^*] \in \mathbb{C}^{n \times n}$ are both unitary matrices, i.e., $UU^* = U^*U = I$ and $VV^* = V^*V = I$. The columns of U are called the left singular vectors of A and the columns of V are the right singular vectors of A. Furthermore, $\Sigma_r \in \mathbb{R}^{m \times n}$ is a diagonal matrix that contains the singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$ of A.

The SVD and eigen decomposition of a matrix $A \in \mathbb{C}^{m \times n}$ are closely related. Let $U\Sigma V^*$ be the SVD of A, then

$$\begin{aligned} A^*A &= V\Sigma^T U^* U \Sigma V^* \\ &= V\Sigma^T \Sigma V^* \\ &= V\Sigma^2 V^*. \end{aligned}$$

Since V^* is a unitary matrix, i.e., $V^* = V^{-1}$, the notation above is equivalent to the eigen decomposition of A^*A, where the elements on the diagonal of Σ^2 are the eigenvalues. Hence, it follows that the right singular vectors of A are the eigenvectors of A^*A. Furthermore, the singular values of A are the square root of the eigenvalues of A^*A.

3.2.2 Solution

Next we can use the SVD to compute the TLS solution. First, note that Equation (3.1) can be rewritten as follows

$$\begin{bmatrix} A & \mathbf{y} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ -1 \end{bmatrix} = \mathbf{0}. \tag{3.5}$$

For ease of notation, define $C = \begin{bmatrix} A & \mathbf{y} \end{bmatrix}$ where the columns of A and $\mathbf{y}$ are concatenated horizontally. Then, applying the singular value decomposition to C, results in $C = U\Sigma V^*$. Next, suppose that A is of full column rank, i.e., A has n linearly independent columns. If no exact solution to Equation (3.1) exists, $\mathbf{y}$ is not in the column space of A. Consequently, the matrix C is of row rank $n + 1$. In order to find a solution to the system, we need to reduce the rank of the rows of C to n. The following theorem adjusted to complex matrices states that the best rank n approximation can be found, using only the n largest singular values and their corresponding singular vectors.

Theorem Eckart-Young-Mirsky. Let the SVD of $C \in \mathbb{C}^{m \times n}$, $m > n$ be given by $C = \sum_{i=1}^{r} \sigma_i u_i v_i^*$, with r = rank$(C)$. If $k < r$ and $C_k = \sum_{i=1}^{k} \sigma_i u_i v_i^*$, then

$$\min_{\text{rank}(D)=k} ||C - D||_2 = ||C - C_k||_2 = \sigma_{k+1}.$$

Proof
This proof makes use of [10] and [2]. Let $D \in \mathbb{C}^{m \times n}$ be an arbitrary matrix with rank$(D) = k$. We have to show that

$$||C - D||_2 \geq ||C - C_k||_2.$$

As D is of rank k, it follows that $\dim \text{Null}(D) = n-k$. Moreover, the subspace of $\mathbb{C}^n$ that consists of the $k+1$ right singular vectors of C corresponding to the $k+1$ largest singular values, $v_1, \ldots, v_{k+1}$, is of rank $k+1$. Meaning that

$$\dim \text{Null}(D) + \dim([v_1, ..., v_{k+1}]) = n - k + k + 1 > n.$$

Since all vectors from both of these subspaces are part of $\mathbb{C}^n$ and the sum of their dimensions is bigger than n, they must have a non-trivial intersection. Hence, there must be an $\mathbf{x} \in \text{Null}(D) \cap R([v_1, ..., v_{k+1}])$. More specifically, there must be an $\mathbf{x}$ of unit length, i.e., $||\mathbf{x}||_2 = 1$. Suppose $\mathbf{x}$ is not of unit length, in this case it holds that $\mathbf{x}/||\mathbf{x}|| \in \text{Null}(D)$ and $\mathbf{x}/||\mathbf{x}|| \in R([v_1, ..., v_{k+1}])$. Thus $\mathbf{x}$ can be scaled to a vector of unit length. Hence, we get the following

$$\begin{aligned}
||C - D||_2^2 &\geq ||(C - D)\mathbf{x}||_2^2 \\
&= ||C\mathbf{x}||_2^2 \\
&= (C\mathbf{x})^* C\mathbf{x} \\
&= \mathbf{x}^* V \Sigma U^* U \Sigma V^* \mathbf{x} \\
&= (V^*\mathbf{x})^* \Sigma^2 (V^*\mathbf{x}) \\
&= \sum_{i=1}^{k+1} \sigma_i^2 ||v_i^* \cdot \mathbf{x}||_1^2 \\
&\geq \sigma_{k+1}^2.
\end{aligned}$$

In the last step, we used that the singular values are denoted in descending order. Furthermore, the 2 norm of a matrix is defined to be its largest singular value. As a result, we get

$$||C - C_k||_2^2 = ||\sum_{i=1}^{n} u_i \sigma_i v_i^* - \sum_{i=1}^{k} u_i \sigma_i v_i^*||_2^2 = ||\sum_{i=k+1}^{n} u_i \sigma_i v_i^*||_2^2 = \sigma_{k+1}^2,$$

since σ_{k+1} denotes the largest resulting singular value of $\sum_{i=k+1}^{n} u_i \sigma_i v_i^*$. We obtain,

$$||C - D||_2 \geq ||C - C_k||_2 = \sigma_{k+1}.$$

□

As a result, we find that the best rank n approximation of C is given by $\sum_{i=1}^{n} u_i \sigma_i v_i^*$. Multiplying this lower rank n approximation of C by the vector v_{n+1} yields

$$\sum_{i=1}^{n} u_i \sigma_i v_i^* v_{n+1} = \mathbf{0},$$

where we used that V is a unitary matrix, i.e., $v_i \cdot v_j = 0$ if $i \neq j$. Hence, $[A \quad \mathbf{y}]v_{n+1} = \mathbf{0}$. In order to obtain the TLS solution as formulated in Equation (3.5), the vector v_{n+1} has to be scaled. Consequently, we find that the total least squares solution $\mathbf{x}_{\text{tls}}$ is given by

$$\mathbf{x}_{\text{tls}} = -\frac{1}{v_{n+1,n+1}} \begin{bmatrix} v_{n+1,1} \\ v_{n+1,2} \\ \vdots \\ v_{n+1,n+1} \end{bmatrix}, \tag{3.6}$$

if $v_{n+1,n+1} \neq 0$. Here $v_{n+1,i}$ denotes the ith entry of the right singular vector that corresponds to singular value σ_{n+1}.

3.3 Power method

It follows from Equation (3.6) that the TLS solution can be found using only the right singular vector of C that corresponds to its smallest singular value σ_{n+1}. This observation can be used in order to compute the TLS solution differently and thereby simplify the needed calculations. Hence, the TLS solution can be computed by the power method [1]. The power method computes the (in absolute value) largest eigenvalue $\lambda_{\max}$ of a diagonalizable matrix and the corresponding eigenvector without decomposing the matrix.

Since the eigenvectors and singular vectors of a matrix are closely related, the power method can also be used to find the right singular vector of C that corresponds to its largest singular value. However, we are interested in the right singular vector of C that corresponds to the smallest singular value. Equivalently, we want to find the eigenvector of C^*C that corresponds to its smallest eigenvalue. In order to do find this eigenvector of C^*C, we have to make some slight alterations. A scalar μ must be found such that $\lambda_{\max} - \mu \leq 0$. Then the method, in this case known as the shifted power method, converges to the eigenvector of $C^*C - \mu I$ which corresponds to the (in absolute value) largest eigenvalue of $C^*C - \mu I$. Next, we show that this can be used to obtain the desired eigenvector of C^*C. Suppose $\mathbf{v}$ is an eigenvector of $C^*C - \mu I$ corresponding to eigenvalue λ, then

$$\begin{aligned}(C^*C - \mu I)\mathbf{v} &= \lambda \mathbf{v} \\ C^*C\mathbf{v} - \mu\mathbf{v} &= \lambda\mathbf{v} \\ C^*C\mathbf{v} &= (\lambda + \mu)\mathbf{v}.\end{aligned}$$

Hence, C^*C and $C^*C - \mu I$ have the same eigenvectors, but different eigenvalues. The scalar μ can be found using the Frobenius norm of C as follows

$$\begin{aligned}\mu &:= ||C||_F^2 \\ &= \text{trace}(C^*C) \\ &= \text{trace}(V\Sigma^2 V^*) \\ &= \text{trace}(\Sigma^2 V^* V) \\ &= \text{trace}(\Sigma^2) \\ &= \sum_{i=1}^{n} \sigma_i^2 \\ &\geq \sigma_{\max}^2,\end{aligned}$$

where $\sigma_{\max}$ denotes the largest singular value of C and therefore $\sigma_{\max}^2$ equals the largest eigenvalue $\lambda_{\max}$ of C^*C.

3.3.1 Algorithm

The power method is an iterative algorithm. We will shortly describe how the algorithm computes the eigenvector of an arbitrary matrix A corresponding to the in absolute value largest eigenvalue. The first step of the algorithm is to select a starting vector $\mathbf{q}_0$, which might be chosen at random. At the following step, the matrix A is multiplied by this vector and then normalized in order to obtain the vector $\mathbf{q}_1$. This process is repeated until the algorithm has converged to the desired eigenvector that corresponds to the eigenvalue of largest magnitude. Hence, the vector obtained at the kth iteration step is as follows:

$$\mathbf{q}_{k+1} = \frac{A\mathbf{q_k}}{||A\mathbf{q_k}||},$$

with $k \geq 0$.

3.3.2 Convergence

Next, we will analyse the convergence of the power method. First, suppose that $|\lambda_1| > |\lambda_2| \geq ... \geq |\lambda_m|$ are the eigenvalues of an arbitrary matrix A. Suppose that the corresponding eigenvectors of A form a basis for $\mathbb{C}^m$. Then, the initial guessed vector $\mathbf{q}_0$ is a linear combination of the corresponding eigenvectors $\mathbf{v}_1, ..., \mathbf{v}_m$. This assumption holds for the matrix C^*C as described above, as it is a diagonalizable matrix. Hence,

$$\mathbf{q}_0 = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + ... + c_m\mathbf{v}_m, \qquad c_i \in \mathbb{R}. \tag{3.7}$$

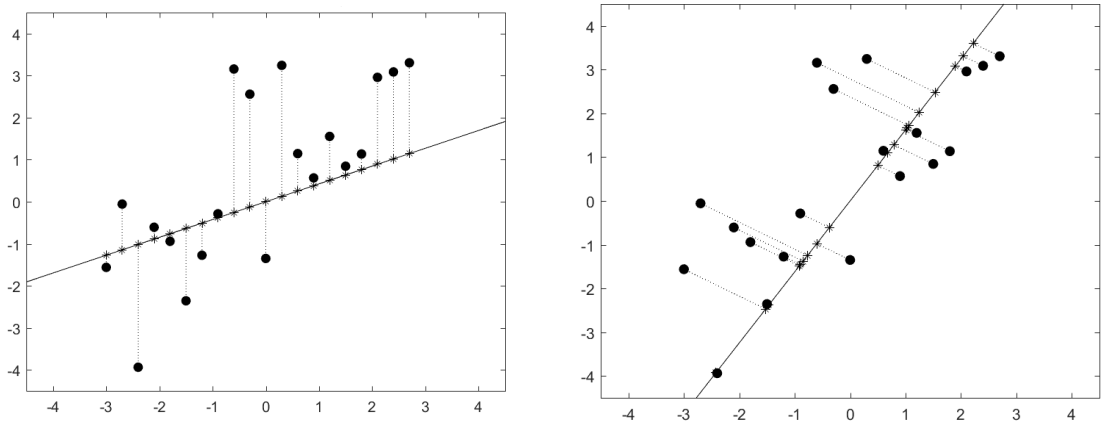
To obtain convergence, $\mathbf{q}_0$ should have a component in the direction of the eigenvector corresponding to the largest eigenvalue in magnitude. Hence, we assume that $c_1 \neq 0$. Then

$$\begin{aligned}
\frac{A^k\mathbf{q}_0}{||A^k\mathbf{q}_0||} &= \frac{A^k(c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + ... + c_m\mathbf{v}_m)}{||A^k(c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + ... + c_m\mathbf{v}_m)||} \\
&= \frac{c_1A^k\mathbf{v}_1 + c_2A^k\mathbf{v}_2 + ... + c_mA^k\mathbf{v}_m}{||c_1A^k\mathbf{v}_1 + c_2A^k\mathbf{v}_2 + ... + c_mA^k\mathbf{v}_m||} \\
&= \frac{c_1\lambda_1^k\mathbf{v}_1 + c_2\lambda_2^k\mathbf{v}_2 + ... + c_m\lambda_m^k\mathbf{v}_m}{||c_1\lambda_1^k\mathbf{v}_1 + c_2\lambda_2^k\mathbf{v}_2 + ... + c_m\lambda_m^k\mathbf{v}_m||} \\
&= \frac{\lambda_1^k\left(c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^k\mathbf{v}_2 + ... + c_m\left(\frac{\lambda_m}{\lambda_1}\right)^k\mathbf{v}_m\right)}{||\lambda_1^k\left(c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^k\mathbf{v}_2 + ... + c_m\left(\frac{\lambda_m}{\lambda_1}\right)^k\mathbf{v}_m\right)||} \\
&= \pm\frac{c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^k\mathbf{v}_2 + ... + c_m\left(\frac{\lambda_m}{\lambda_1}\right)^k\mathbf{v}_m}{||c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^k\mathbf{v}_2 + ... + c_m\left(\frac{\lambda_m}{\lambda_1}\right)^k\mathbf{v}_m||}.
\end{aligned}$$

Since $|\lambda_1|$ is a dominant eigenvalue by assumption, this expression goes to $\pm\frac{c1}{|c1|}\mathbf{v_1}$, where $\mathbf{v}_1$ is normalized, as k tends to infinity. Hence, the power method converges to a multiple of the eigenvector that corresponds to the largest eigenvalue in magnitude. When other eigenvalues are in absolute value close to $|\lambda_1|$, the power method will converge more slowly.

3.4 Difference between least squares and total least squares

To gain some insight into the differences between LS and TLS, we will consider an example which is shown in Figure 5. In this example, we are dealing with an overdetermined system of equations, i.e. there are more equations than unknowns. Since no exact solution to this system exists, the aim is to find a straight line that best fits the twenty perturbed data points. In this example, the line is passing through the origin, thus the slope of the curve is the only parameter to be estimated. The least squares method determines the best approximation by correcting only the y-coordinate of each point, which can be seen in Figure 5a. Whereas Figure 5b shows that the total least squares method aims to find the best curve by adjusting both the x-, and y-coordinates. Every $*$ in Figure 5 denotes what the corrected value of the perturbed data point is.



(a) Least squares solution.

(b) Total least squares solution.

Figure 5: Least squares and total least squares solutions of a perturbed system of equations.

From this example, it can be observed that the LS and TLS errors are characterized in different ways. Leading to fairly different results in this case. The least squares error is defined by the sum of the squared vertical distances between each measured data point and its corresponding approximated data point on the line. Whereas the TLS error minimizes the sum of the squared orthogonal distances between a data point and its corrected value.

3.5 Examples

Next, we present some examples for which we determine the LS and TLS solutions. The aim of these examples is to show what method derives the best solution compared to the exact solution. Both LS and TLS will be applied to an overdetermined system of equations as described by Equation (3.1). However, in this case there exists an exact solution to the system, which is known beforehand. Subsequently, normally distributed noise with zero mean and variable variance will be added to the system. The noise is added to either the measurement vector $\mathbf{y}$ or both $\mathbf{y}$ and the coefficient matrix A. Next, LS and TLS are applied to the perturbed system in order to find the LS and TLS solution, respectively. After every test, the results show what method gave the best approximation to the exact solution. Here, the best approximation is defined to be the one with smallest Euclidean distance to the exact solution. This set-up will be repeated 200000 times for each example. The final results show how often each method gave the best solution (in percentages). In all examples, a cut-off value of 10^{-10} is used, meaning if the errors differ at most 10^{-10}, the solutions are considered to be equal.

3.5.1 Polynomial regression

This first example is similar to the example we looked at in Section 3.4. Again, the aim is to fit a polynomial to the perturbed system of equations. In this case, the system is as follows,

$$\begin{bmatrix} 1 & a_1 & a_1^2 & \dots & a_1^n \\ 1 & a_2 & a_2^2 & \dots & a_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & a_m & a_m^2 & \dots & a_m^n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}, \tag{3.8}$$

where $m > n$. Solving this system is equivalent to fitting a n-th degree polynomial. This can be described as follows

$$y_i = \sum_{j=0}^{n} a_i^j \cdot x_j,$$

for $i \in \{1, ..., m\}$.

In the example underneath, the aim is to fit a first degree polynomial $y = \alpha + \beta x$ to a set of perturbed data points. In order to do so, the two parameters α and β, have to be estimated. The system we use in the example is as follows

$$\begin{bmatrix} 1 & -2.0 \\ 1 & -1.8 \\ \vdots & \vdots \\ 1 & 1.8 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} -1.3 \\ -1.1 \\ \vdots \\ 2.5 \end{bmatrix}, \tag{3.9}$$

with the exact solution $[\alpha, \beta]^T = [0.7, 1.0]^T$. The code of this example is given in Appendix B.2. Next we add Gaussian noise to the measurements in the vector $\mathbf{y}$ on the right-hand side of Equation (3.9) with different values of the standard deviation $s_{\mathbf{y}}$. The data in the matrix A are free of error. The results that are obtained after applying LS and TLS to this system are shown in Table 1.

Table 1: The results after applying LS and TLS to fit a first degree polynomial to a perturbed system of equations. The first column shows the value of the standard deviation $s_{\mathbf{y}}$ of the normally distributed noise which is applied to the measurements in $\mathbf{y}$. The second and third column show how often the LS and TLS method gave the best solution in percentages, respectively.

$s_{\mathbf{y}}$	LS (in %)	TLS (in %)
1.00	88.7	11.3
0.50	71.0	29.0
0.10	54.3	45.7
0.05	52.1	47.9
0.01	50.1	49.9

It is clear that for larger values of the $s_{\mathbf{y}}$, the least squares method performs better than the total least squares method. As only vertical perturbations are added to $\mathbf{y}$, which in turn will be minimized by the least squares method, this result is as expected. For smaller values of $s_{\mathbf{y}}$, both methods become almost equally likely to give the best solution. Next, also the data matrix A will be perturbed with normally distributed noise with zero mean and different values of the standard deviation $s_{\mathbf{A}}$. The aim is still to fit a first degree polynomial as explained above. The obtained results are shown in Table 2.

Table 2: The results after applying LS and TLS to fit a first degree polynomial to a perturbed system of equations. The first and second column show the value of the standard deviations s_A and $s_{\mathbf{y}}$ of the normally distributed noise which is applied to the measurements in A and $\mathbf{y}$, respectively. The third and fourth column show how often the LS and TLS method gave the best solution in percentages, respectively.

s_A	$s_{\mathbf{y}}$	LS (in %)	TLS (in %)
1.00	1.00	38.4	61.6
0.50	0.50	32.6	67.4
0.10	0.10	45.6	54.4
0.05	0.05	47.8	52.2
0.01	0.01	49.5	50.5

The results are different when the data matrix A contains normally distributed noise as well, compared to the case where only $\mathbf{y}$ is perturbed. In this case, we see that for larger values of the standard deviation of the noise, TLS performs better than LS, as expected. For smaller values of s_A and $s_{\mathbf{y}}$, the two methods give similar results, as in the first scenario.

3.5.2 Exponential regression

Next, we consider an example which corresponds to the case of images generated by an MRI scanner. As observed in Equation (2.15), the obtained signal is a combination of sines and cosines corresponding to different frequencies f_i at different points in time t_j, with $i \in \{1, ..., m\}$ and $j \in \{1, ..., n\}$. Therefore it might be interesting to have a look at an example of the following form, where the entries of the coefficient matrix A are $e^{2\pi i f_i t_j}$. Hence, the overdetermined system takes the following form

$$\begin{bmatrix} e^{2\pi i f_1 t_1} & e^{2\pi i f_1 t_2} & \dots & e^{2\pi i f_1 t_n} \\ e^{2\pi i f_2 t_1} & e^{2\pi i f_2 t_2} & \dots & e^{2\pi i f_2 t_n} \\ \vdots & \vdots & \ddots & \vdots \\ e^{2\pi i f_m t_1} & e^{2\pi i f_m t_2} & \dots & e^{2\pi i f_m t_n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}, \tag{3.10}$$

with $m > n$. For this example, we used $\mathbf{f} = [0, ..., 6]^T$ and $\mathbf{t} = [0, \frac{1}{175}, \frac{2}{175}, ..., 1]^T$. The exact solution is chosen as $\mathbf{x} = [2, 1.5, 4, 2.5, 3.5, 2, 1]^T$, which is used to determine $\mathbf{y}$. The code which is used for this example is given in Appendix B.3. The results after applying both LS and TLS to this system when only the measured data in $\mathbf{y}$ are perturbed with Gaussian noise and different values of the standard deviation $s_{\mathbf{y}}$, are shown in Table 3.

Table 3: The results after applying LS and TLS to the system described in Equation (3.10). The first column shows the value of the standard deviation $s_{\mathbf{y}}$ of the normally distributed noise which is applied to the measurements in $\mathbf{y}$. The second and third column show how often the LS and TLS method gave the best solution in percentages, respectively.

$s_{\mathbf{y}}$	LS (in %)	TLS (in %)
1.00	97.5	2.5
0.50	83.7	16.3
0.10	57.5	42.5
0.05	54.0	46.0
0.01	50.9	49.1

As in the previous example, LS performs better than TLS for larger values of $s_{\mathbf{y}}$. The smaller the

standard deviation of the noise becomes, the more equally the methods perform. Next, also the data matrix A will be perturbed with Gaussian noise with zero mean and different values of the standard deviation $s_{\mathbf{A}}$. The aim is still to find the best solution $\mathbf{x}$ to the system which is described in Equation (3.10). The obtained results after applying both LS and TLS are shown in Table 4.

Table 4: The results after applying LS and TLS to the system described in Equation (3.10). The first and second column show the value of the standard deviations s_A and $s_{\mathbf{y}}$ of the normally distributed noise which is applied to the measurements in A and $\mathbf{y}$, respectively. The third and fourth column show how often the LS and TLS method gave the best solution in percentages, respectively.

s_A	$s_{\mathbf{y}}$	LS (in %)	TLS (in %)
1.00	1.00	23.4	76.6
0.50	0.50	0.0	100.0
0.10	0.10	15.0	85.0
0.05	0.05	30.1	69.9
0.01	0.01	45.7	54.3

From the results it follows that TLS performs better than LS, especially for larger values of $s_{\mathbf{A}}$ and $s_{\mathbf{y}}$. When the standard deviation of the noise becomes small, there is almost no noise, hence both methods are almost equally likely to give the best result.

3.5.3 Exponential regression with specific noise

In the previous example, Gaussian noise has been added to the data matrix A. However, it is more likely that a specific part of the entries of A contain noise. For example, the frequencies may contain noise. Therefore, this example focuses on errors in f_i, as described in Equation (3.10), instead of errors on the data matrix itself. Applying both LS and TLS to this perturbed system for different values of the standard deviation $s_{\mathbf{f}}$ of the noise applied to $\mathbf{f}$, leads to the results as shown in Table 5.

Table 5: The results after applying LS and TLS to the system which is described in Equation (3.10). The first and second column show the value of the standard deviations $s_{\mathbf{f}}$ and $s_{\mathbf{y}}$ of the normally distributed noise which is applied to the measurements in $\mathbf{f}$ and $\mathbf{y}$, respectively. The third and fourth column show how often the LS and TLS method gave the best solution in percentages, respectively.

s_f	$s_{\mathbf{y}}$	LS (in %)	TLS (in %)
1.00	1.00	100.0	0.0
0.50	0.50	100.0	0.0
0.10	0.10	94.2	5.8
0.05	0.05	82.5	17.5
0.01	0.01	58.0	41.9

The least squares method clearly outperforms the total least squares method. As in previous examples, the added noise is normally distributed. However, in this case, the normally distributed noise is added to a component of an exponential power. Hence, the noise on the data matrix A is not Gaussian any more, whereas the noise on the measurements in $\mathbf{y}$ still are. As both sides of the equation are no longer perturbed the same way, this may partially explain the unexpected results.

4 Least squares and total least squares applied to MRI data

The examples we considered in Section 3.5 were included to show differences in the LS and TLS solution of a perturbed and overdetermined set of equations. However, these examples do not necessarily represent the case where actual MRI data is used. Therefore, the aim of this section is to show how TLS can be used to potentially improve the quality of MRI images, where data from the LUMC MRI scanner is used. In Section 4.1 we describe how the images are obtained in this case. Next, in Section 4.2 we introduce regularization, which is used to cancel out noise while computing the LS and TLS solution. In Section 4.3 and Section 4.4 it is explained how the corresponding LS and TLS solutions can be derived.

4.1 Fast Fourier transform

MRI images are often obtained using a Fourier transform [12], i.e., applying a Fourier transform to the measured signals leads to the desired image. Computing a Fourier transform can be carried out by multiplying the measured signals with a discrete Fourier transform (DFT) matrix, which is a (scaled) unitary matrix of full column rank. However, this can be a very time-consuming process. A more efficient way of computing the Fourier transform of a signal, is by using the fast Fourier transform (FFT). This algorithm decomposes the DFT matrix in order to compute the Fourier transform and is therefore very efficient. The image, i.e., $\mathbf{x}$ is determined using a inverse fast Fourier transform (IFFT) such that $\mathbf{x} = \text{IFFT}(\mathbf{y})$, where $\mathbf{y}$ contains the measured signals. As these signals are perturbed, the obtained image contains noise as well.

4.2 Regularization

In an attempt to cancel out as much noise as possible from the obtained image $\mathbf{x}$, a regularization operator L can be added to the problem. Regularization allows for adding prior knowledge about the solution. For example, when it is known that neighbouring pixel values barely differ, a first order difference matrix can be used as a regularization operator. The least squares method as presented by Equation (3.3), becomes the following problem when adding regularization [5]

$$\min_{\Delta\mathbf{y}\in\mathbb{C}^m} ||\Delta\mathbf{y}||_2 + \tau||L\mathbf{x}||_2, \quad \text{subject to } \mathbf{y}+\Delta\mathbf{y} \in R(A). \tag{4.1}$$

Here, τ is a regularization parameter, which can be used to either limit or amplify the influence of the regularization operator L. Choosing a useful regularization parameter is not trivial, in [6] a method is described which can be used in an attempt to choose a good regularization parameter. Moreover, Equation (4.1) can also be presented as

$$\min_{\Delta\mathbf{y}\in\mathbb{C}^m} ||\Delta\mathbf{y}||_2, \quad \text{subject to } \mathbf{y}+\Delta\mathbf{y} \in R(A) \text{ and } ||L\mathbf{x}||_2 \leq \delta, \tag{4.2}$$

where δ is a positive constant. The corresponding normal equation is given by

$$(A^*A + \tau^2 L^*L)\mathbf{x} = A^*\mathbf{y}. \tag{4.3}$$

Similarly, the total least squares method as presented by Equation (3.4), becomes the following problem when adding regularization

$$\min_{\Delta A\in\mathbb{C}^{m\times n},\ \Delta\mathbf{y}\in\mathbb{C}^m} || \begin{bmatrix}\Delta A & \Delta\mathbf{y}\end{bmatrix} ||_F, \quad \text{subject to } \mathbf{y}+\Delta\mathbf{y} \in R(A+\Delta A) \text{ and } ||L\mathbf{x}||_2 \leq \delta. \tag{4.4}$$

Different choices can be made for the operator L. We use two regularization operators, D_x and D_y. Here, D_x is a first order difference matrix, i.e., a matrix with 1 on the main diagonal and -1 on

the super diagonal. Hence, D_x allocates pixels next to each other the same value. Analogously, the first order difference matrix D_y is added in order to assign pixels above and below each other the same value. Thus, D_y is a matrix with 1 on the main diagonal and -1 on the diagonal that starts p positions to the right of the main diagonal. Here, p denotes the number of pixels that a row of the image contains. Hence, as can be observed from Equation (4.3), the aim is to determine the best approximate solution $\mathbf{x}$ such that

$$\begin{bmatrix} \mathrm{FFT}(\cdot) \\ \tau \mathrm{D}_x \\ \tau \mathrm{D}_y \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}. \tag{4.5}$$

4.3 Regularized least squares

The corresponding normal equation for the system described in Equation (4.5) is as follows

$$(I + \tau^2 \mathrm{D}_x^* \mathrm{D}_x + \tau^2 \mathrm{D} y^* \mathrm{D}_y)\mathbf{x} = \mathrm{IFFT}(\mathbf{y}), \tag{4.6}$$

note that $\mathrm{IFFT}(\mathbf{y})$ is the original obtained image, which is contaminated by noise.

4.4 Regularized total least squares

In order to find the TLS solution, we again have to determine a shift μ to make sure that the power method converges to the smallest singular value and the corresponding right singular vector of

$$C := \begin{bmatrix} \mathrm{FFT}(\cdot) & \mathbf{y} \\ \tau \mathrm{D}_x & \mathbf{0} \\ \tau \mathrm{D}_y & \mathbf{0} \end{bmatrix}. \tag{4.7}$$

As the notation of C is not fully represented by matrix and vector notation, we cannot use the Frobenius norm of C in order to determine the shift μ. However, we can determine C^*C and use Gershgorin's circle theorem in order to find bounds for the largest eigenvalue $\lambda_{\max}$ of C^*C. For notational convenience, let $F(\cdot) = \mathrm{FFT}(\cdot)$ and $F^*(\cdot) = \mathrm{IFFT}(\cdot)$, then

$$C^*C = \begin{bmatrix} F^*(\cdot) & \tau \mathrm{D}_x^* & \tau \mathrm{D}_y^* \\ \mathbf{y}^* & \mathbf{0}^T & \mathbf{0}^T \end{bmatrix} \begin{bmatrix} F(\cdot) & \mathbf{y} \\ \tau \mathrm{D}_x & \mathbf{0} \\ \tau \mathrm{D}_y & \mathbf{0} \end{bmatrix} \tag{4.8}$$

$$= \begin{bmatrix} I + \tau^2 \mathrm{D}_x^* \mathrm{D}_x + \tau^2 \mathrm{D}_y^* \mathrm{D}_y & F^*(\mathbf{y}) \\ (F^*(\mathbf{y}))^* & \mathbf{y}^*\mathbf{y} \end{bmatrix}. \tag{4.9}$$

Using Gershgorin's theorem, we find the following bound for the eigenvalues λ for the top part of C^*C,

$$|\lambda - (1 + 2\tau^2 + 2\tau^2)| \leq 2\tau^2 + 2\tau^2 + \max |F^*(\mathbf{y})|.$$

For the bottom part of C^*C we find that

$$|\lambda - \mathbf{y}^*\mathbf{y}| \leq \sum_{i=1}^{n^2+1} |(F^*(\mathbf{y}))_i^*|,$$

where $n^2 + 1$ is the number of columns of C^*C. Hence, we find that

$$\lambda_{\max} \leq \mu := \max\{1 + 8\tau^2 + \max |F^*(\mathbf{y})|, \quad \mathbf{y}^*\mathbf{y} + \sum_{i=1}^{n^2+1} |(F^*(\mathbf{y}))_i^*|\}. \tag{4.10}$$

5 Results

In this section we consider three different test problems. Each of these cases provide an image which is generated by the low-cost MRI scanner from Leiden University Medical Center. We will apply both the regularized least squares method (RLS) and regularized total least squares method (RTLS) to these cases and discuss the obtained results. The code which is used to obtain the results is in Appendix B.4.

5.1 Test case Orange

The first test case is an image of an orange. The original image which is obtained by applying the inverse fast Fourier transform to the measured signals, is as follows.

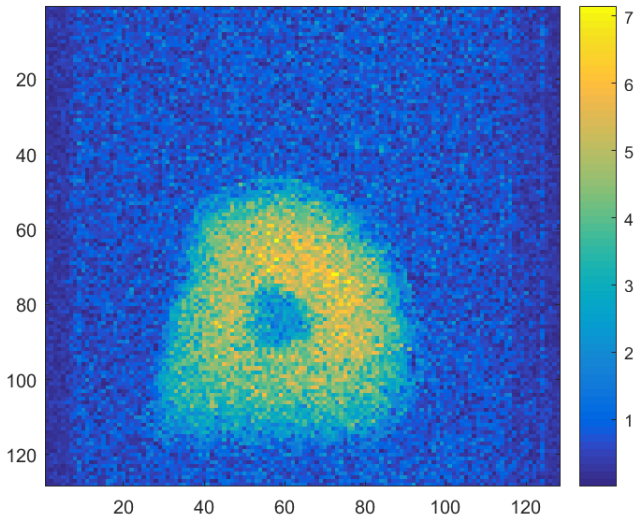
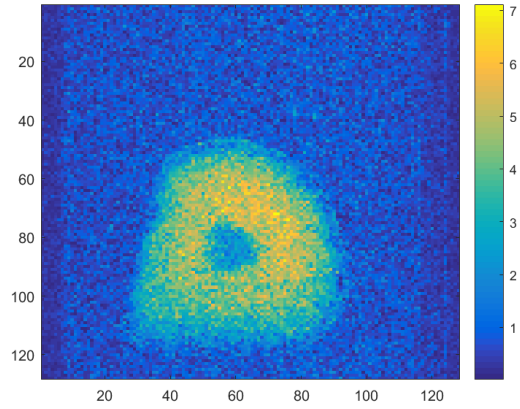


Figure 6: The original MRI image of the test case Orange, which is obtained by applying the inverse fast Fourier transform to the measured MRI signal.

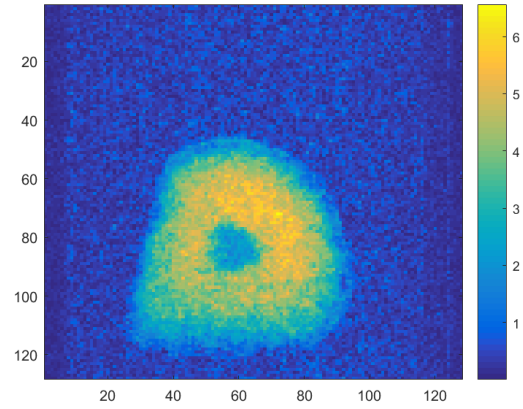
The image shows roughly the contour of the orange, while it is clear that a lot of noise is present. In an attempt to remove as much noise as possible, we will apply both RLS and RTLS.

5.1.1 Regularized least squares

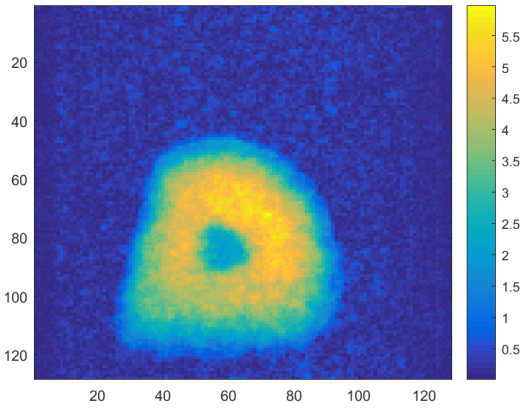
In this subsection we will observe the results when RLS is applied. Hence, the aim is to solve the system as described in Equation (4.5), where $\mathbf{y}$ contains the measured signals of the Orange test case. We will use different values of the regularization parameter τ, which are chosen for demonstration purposes. The obtained results are shown in Figure 7.



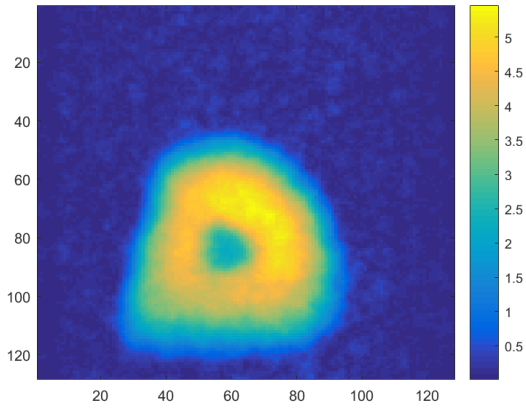
(a) $\tau = 0.1$.



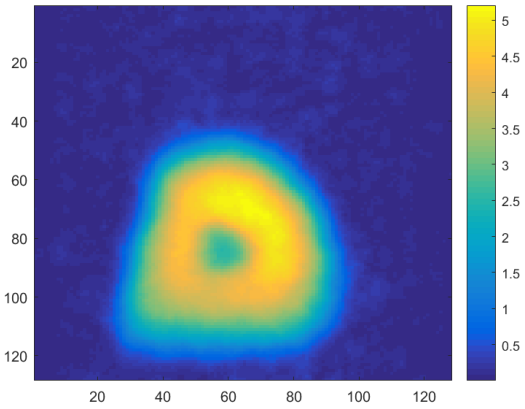
(b) $\tau = 0.5$.



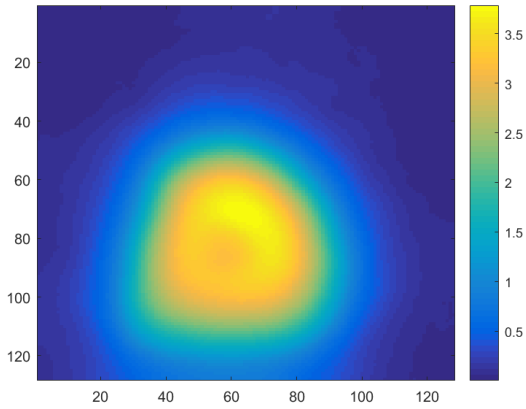
(c) $\tau = 1$.



(d) $\tau = 2$.



(e) $\tau = 3$.



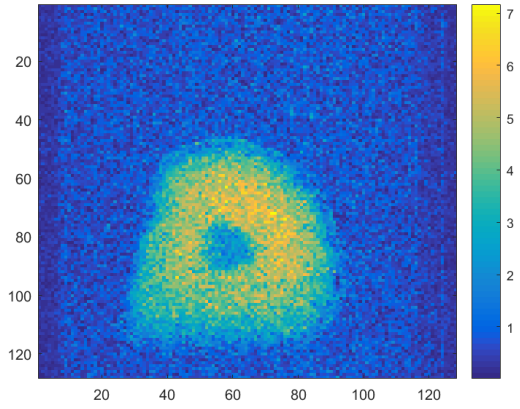
(f) $\tau = 10$.

Figure 7: Results after applying the regularized least squares method to the test problem Orange for different values of the regularization parameter τ.

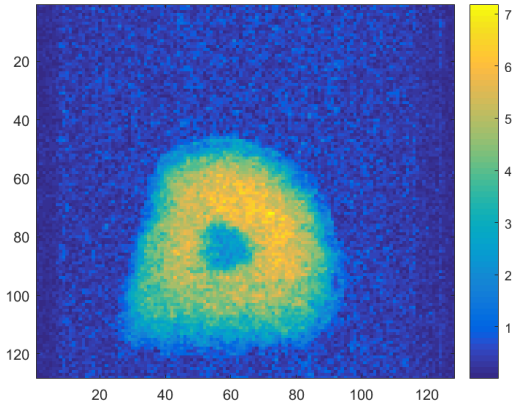
From the results, it can be observed that for larger values of τ, the images are less contaminated by noise. On the other hand, the images become blurry as the values of neighbouring pixels get closer.

5.1.2 Regularized total least squares

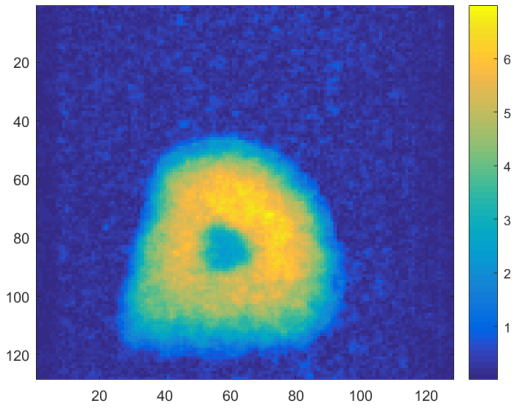
When using the same values for the regularization parameter τ as with RLS, the following RTLS solutions are obtained for the test problem Orange.



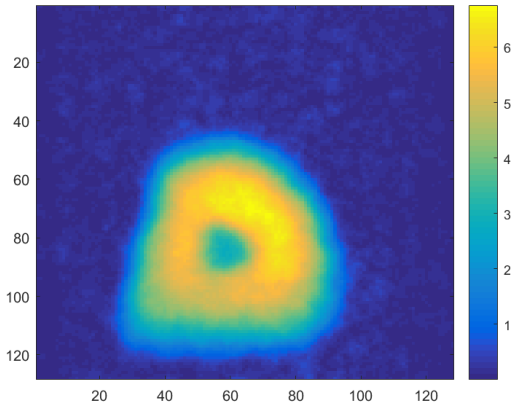
(a) $\tau = 0.1$.



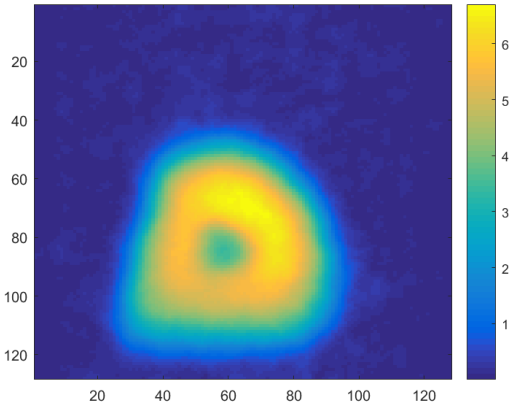
(b) $\tau = 0.5$.



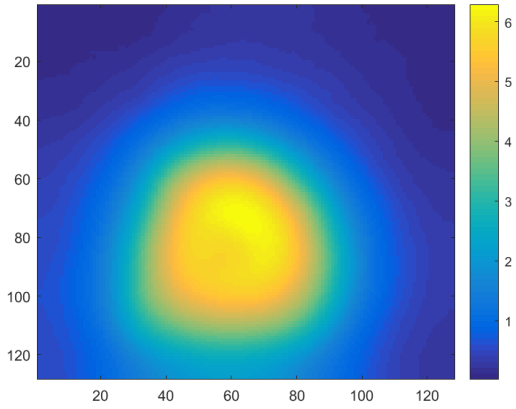
(c) $\tau = 1$.



(d) $\tau = 2$.



(e) $\tau = 3$.



(f) $\tau = 10$.

Figure 8: Results after applying the regularized total least squares method to the test problem Orange for different values of the regularization parameter τ.

The RLS and RTLS solutions seem very similar up to a scaling constant for almost all values of τ. However, when $\tau = 10$, differences between the solutions can be observed. Both RLS and RTLS show images that are too smooth to observe the orange properly. But the RTLS solution seems even smoother than the RLS solution. In order to explain these results, we derive a different formulation of the (regularized) TLS solution, as shown in [11].

Suppose $\sigma_{\min}$ is the smallest singular value of C which corresponds to the right singular vector $\mathbf{v}$, where C is as described in Equation (4.7). Then it follows that $\sigma_{\min}^2$ is an eigenvalue of C^*C corresponding to the eigenvector $\mathbf{v}$. As shown in Section 3.2.2, $\mathbf{v}$ is a scaled version of the TLS solution $\mathbf{x}_{\text{tls}}$. On the other hand, $\begin{bmatrix}\mathbf{x}_{\text{tls}}\\ -1\end{bmatrix}$ is an eigenvector of C^*C corresponding to eigenvalue $\sigma_{\min}^2$. Hence, we can formulate the following eigen decomposition of C^*C

$$\begin{bmatrix} I + \tau^2 \mathrm{D}_x^* \mathrm{D}_x + \tau^2 \mathrm{D}_y^* \mathrm{D}_y & F^*(\mathbf{y}) \\ (F^*(\mathbf{y}))^* & \mathbf{y}^*\mathbf{y} \end{bmatrix} \begin{bmatrix}\mathbf{x}_{\text{tls}}\\ -1\end{bmatrix} = \sigma_{\min}^2 \begin{bmatrix}\mathbf{x}_{\text{tls}}\\ -1\end{bmatrix}. \tag{5.1}$$

The top half of this equation shows that

$$(I + \tau^2 \mathrm{D}_x^* \mathrm{D}_x + \tau^2 \mathrm{D}_y^* \mathrm{D}_y)\mathbf{x}_{\text{tls}} - F^*(\mathbf{y}) = \sigma_{\min}^2 \mathbf{x}_{\text{tls}}, \tag{5.2}$$

$$((1 - \sigma_{\min}^2)I + \tau^2 \mathrm{D}_x^* \mathrm{D}_x + \tau^2 \mathrm{D}_y^* \mathrm{D}_y)\mathbf{x}_{\text{tls}} = F^*(\mathbf{y}). \tag{5.3}$$

Hence an alternate expression for the RTLS solution is given by

$$\mathbf{x}_{\text{tls}} = ((1 - \sigma_{\min}^2)I + \tau^2 \mathrm{D}_x^* \mathrm{D}_x + \tau^2 \mathrm{D}_y^* \mathrm{D}_y)^{-1} F^*(\mathbf{y}), \tag{5.4}$$

or equivalently,

$$\left(I + \frac{\tau^2}{(1 - \sigma_{\min}^2)} \mathrm{D}_x^* \mathrm{D}_x + \frac{\tau^2}{(1 - \sigma_{\min}^2)} \mathrm{D}_y^* \mathrm{D}_y\right) \mathbf{x}_{\text{tls}} = \frac{F^*(\mathbf{y})}{1 - \sigma_{\min}^2}, \tag{5.5}$$

for $\sigma_{\min}^2 \neq 1$. Since the obtained image is scaled, $F^*(\mathbf{y})/(1 - \sigma_{\min}^2)$ is optically the same as $F^*(\mathbf{y})$. From Equation (4.6) we know that the RLS solution $\mathbf{x}_{\text{ls}}$ corresponding to the same system of equations is given by

$$\mathbf{x}_{\text{ls}} = (I + \tau^2 \mathrm{D}_x^* \mathrm{D}_x + \tau^2 \mathrm{D}_y^* \mathrm{D}_y)^{-1} F^*(\mathbf{y}). \tag{5.6}$$

Hence, the RLS and RTLS solutions are equivalent for $\sigma^2_{\min} = 0$.

The value of $\sigma^2_{\min}$ can be determined by the Rayleigh quotient, using the obtained eigenvector $\mathbf{v}$ of $C^*C - \mu I$ which is computed by the shifted power method. Here, μ denotes the shift as determined in Equation (4.10). Note that $\mathbf{v}$ is also an eigenvector of C^*C as explained in Section 3.3. Hence, the largest eigenvalue λ of $C^*C - \mu I$ is given by

$$\lambda = \frac{\mathbf{v}^*(C^*C - \mu I)\mathbf{v}}{\mathbf{v}^*\mathbf{v}}. \tag{5.7}$$

Then the corresponding eigenvalue of C^*C is given by $\sigma^2_{\min} = \lambda + \mu$.

From Equations (5.5) and (5.6), it follows that the RLS and RTLS solutions differ by a scaling constant and regularization parameter. The optical differences are only caused by the difference in regularization parameter, i.e., the RLS solution would visually give the same result as the RTLS solution when the regularization parameters $\tau/\sqrt{1 - \sigma^2_{\min}}$ and τ are used for RLS and RTLS, respectively. As we have used the same values of the regularization parameter in the case above, the RLS and RTLS solutions will only visibly differ for large values of τ or values of $\sigma^2_{\min}$ which are very close to 1. The values of $\sigma^2_{\min}$, the smallest eigenvalue of C^*C, for different values of τ have been computed by the Rayleigh quotient and are shown in Table 6.

Table 6: The value of the smallest eigenvalue $\sigma^2_{\min}$ of C^*C for different values of the regularization parameter τ.

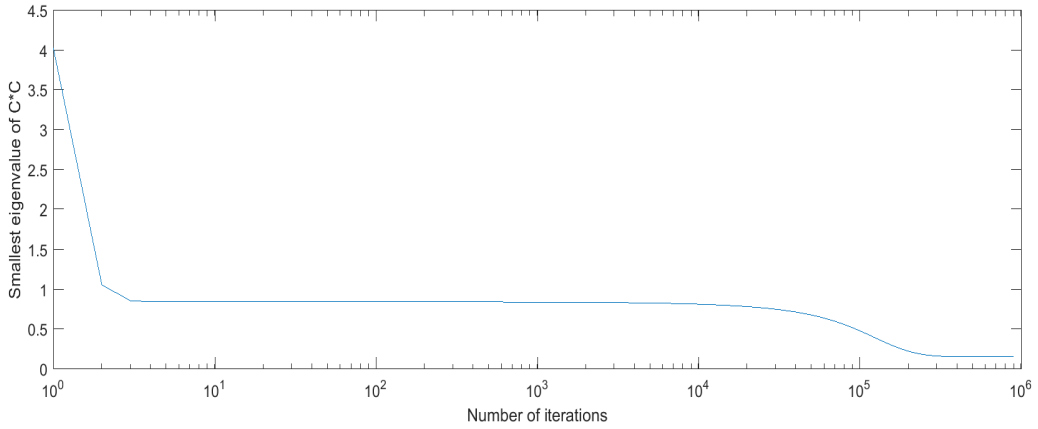
τ	$\sigma^2_{\min}$
0.1	0.0073
0.5	0.0933
1.0	0.1535
2.0	0.2015
3.0	0.2362
10.0	0.5006

These values of $\sigma^2_{\min}$ explain the obtained results. For example, when $\tau = 0.1$, the term $\mathrm{D}_x^*\mathrm{D}_x + \mathrm{D}_y^*\mathrm{D}_y$, as described in Equation (5.6) is multiplied by 0.1^2 in order to obtain the RLS solution. Whereas this same value of τ leads to multiplication of $\left(\frac{0.1}{\sqrt{1-0.0073}}\right)^2 \approx 0.1004^2$ in the RTLS case, as can be observed in Equation (5.5). Hence, the solutions barely differ from an optical point of view. Conversely, the value $\tau = 10$ for the RLS solution corresponds with a multiplication of $\mathrm{D}_x^*\mathrm{D}_x + \mathrm{D}_y^*\mathrm{D}_y$ by 10^2. For RTLS, this results in multiplication by a factor $\left(\frac{10}{\sqrt{1-0.5006}}\right)^2 \approx 19.9760^2$, this relatively big difference is visible in the images. Observing all obtained values of $\sigma^2_{\min}$ in Table 6, the RLS and RTLS solutions are further apart for larger values of τ, which corresponds to the results as shown in Figures 7 and 8.

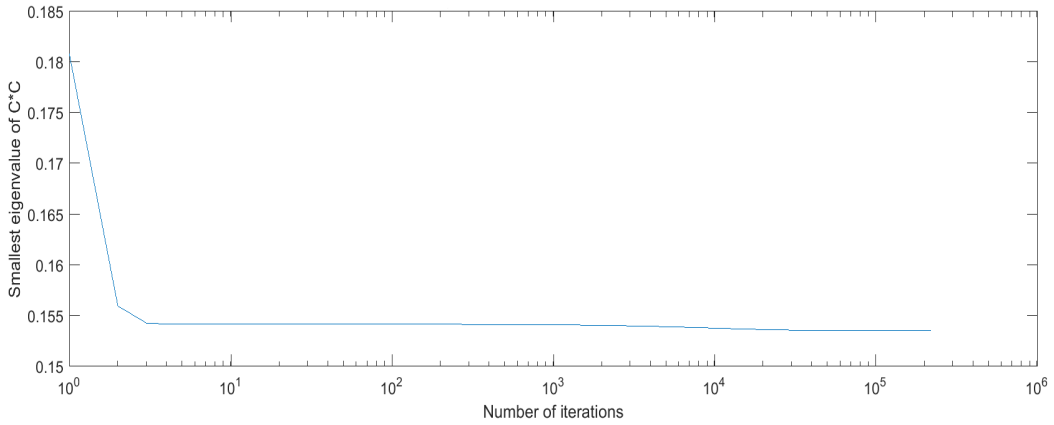
5.1.3 Convergence

The shifted power method as described in Sections 3.3 and 4.4 is used to compute the RTLS solution. As explained, any arbitrary starting vector can be used in order to find the solution, as long as the vector has a component in the direction of the largest eigenvector. Hence, we first used a starting vector with every entry equal to 1. As this starting vector is chosen arbitrarily, it might converge relatively slowly to the solution. Hence, another starting vector might provide a faster convergence rate. Since the results in Figure 7 and Figure 8 are very similar, the RLS

solution could be used in an attempt to obtain a better initial vector. Hence, the first entries of this vector correspond to the RLS solution and the last component is set equal to minus one, as the starting vector needs one extra entry compared to the RLS solution. In this case, the shifted power method stops iterating if two consecutively RTLS solutions differ less than 10^{-6}. In Figure 9 the convergence of the shifted power method to $\sigma^2_{\min}$, the desired eigenvalue of C^*C, is shown for both starting vectors when $\tau = 1$.



(a) Convergence of the shifted power method to $\sigma^2_{\min}$ while using the starting vector where every entry is equal to 1.



(b) Convergence of the shifted power method to $\sigma^2_{\min}$ while using the RLS solution as the first entries of the starting vector and the last entry is equal to -1.

Figure 9: Convergence of the shifted power method to the smallest eigenvalue $\sigma^2_{\min}$ of C^*C when $\tau = 1$ for different starting vectors.

Both starting vectors eventually lead to the smallest eigenvalue of approximately 0.1535. As can be observed from Figures 9a and 9b, the first starting vector needs hundreds of thousands of iterations before the desired eigenvalue is found, whereas the adjusted RLS starting vector only needs a couple of iterations to get very close to the smallest eigenvalue. Hence, the first starting vector experiences a much slower convergence rate than the RLS starting vector.

5.2 Test case Tubes

The next test problem is an image of five tubes. The image which is originally obtained by applying the inverse fast Fourier transform to the MRI signal, is as follows.

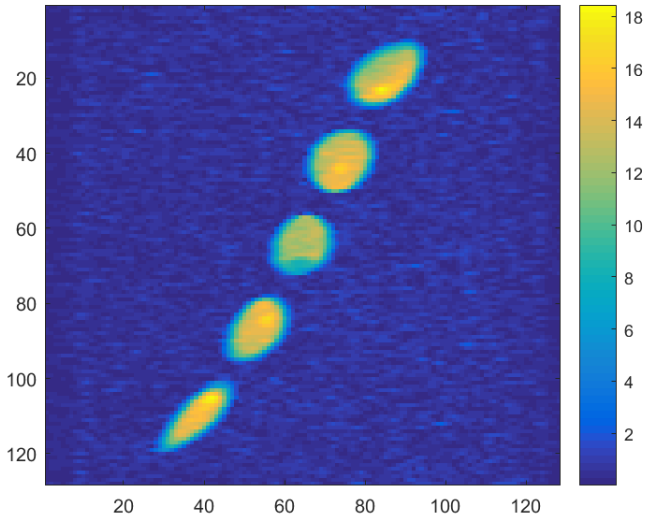
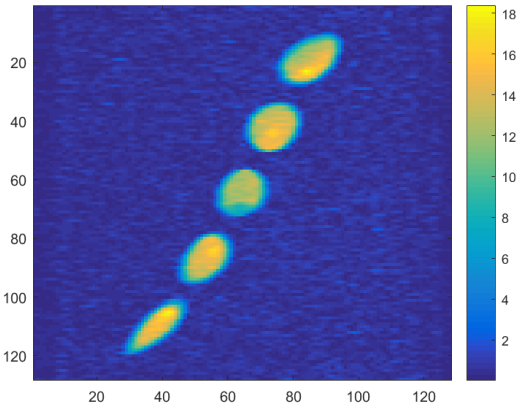


Figure 10: The original MRI image for the test problem Tubes, which is obtained by applying the inverse fast Fourier transform to the measured MRI signal.

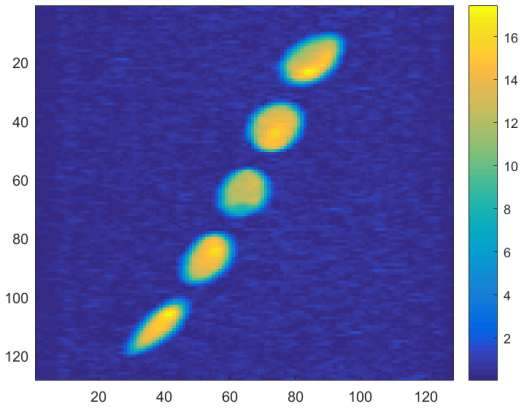
In comparison with Figure 6 of the previous test example, this image seems originally less contaminated by noise.

5.2.1 Regularized least squares

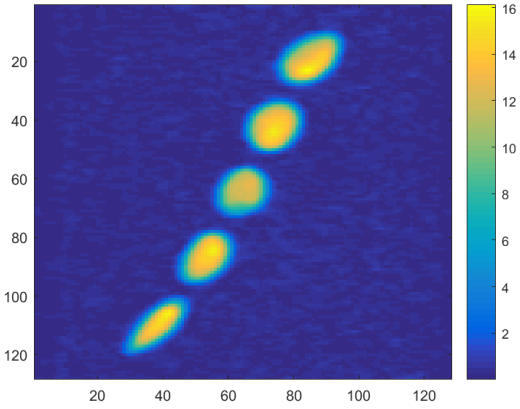
Next, we apply regularized least squares to the MRI signal for the same values of the regularization parameter τ as in the previous test case. The results are shown below.



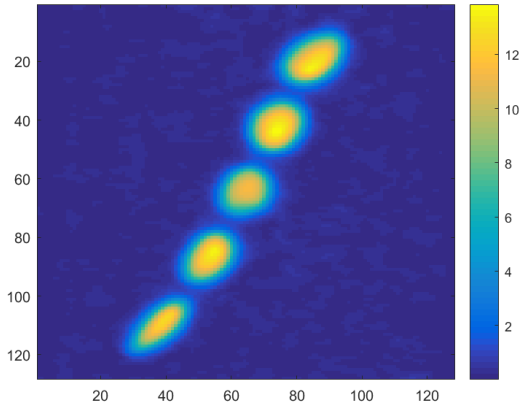
(a) $\tau = 0.1$.



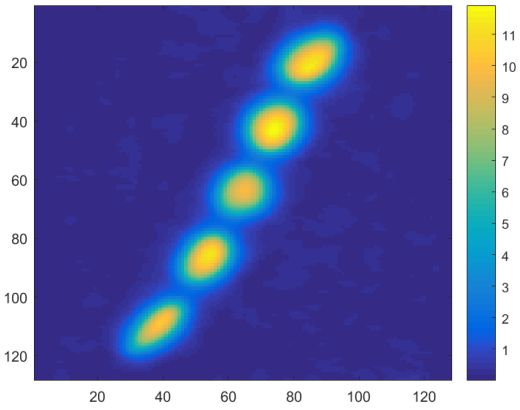
(b) $\tau = 0.5$.



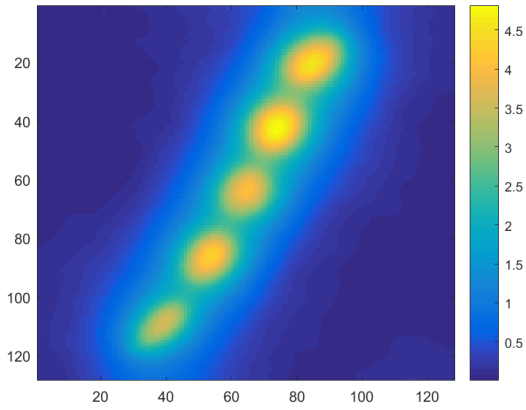
(c) $\tau = 1$.



(d) $\tau = 2$.



(e) $\tau = 3$.



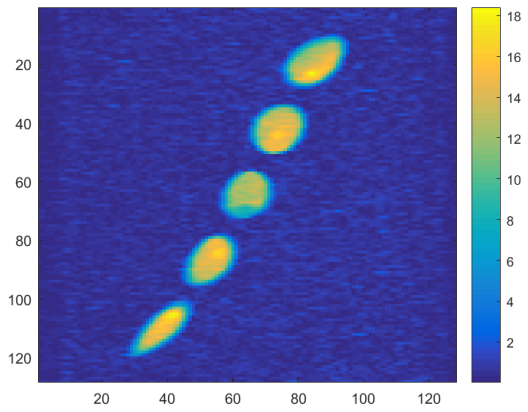
(f) $\tau = 10$.

Figure 11: Results after applying the regularized least squares method to the test problem Tubes with different regularization parameter values τ.

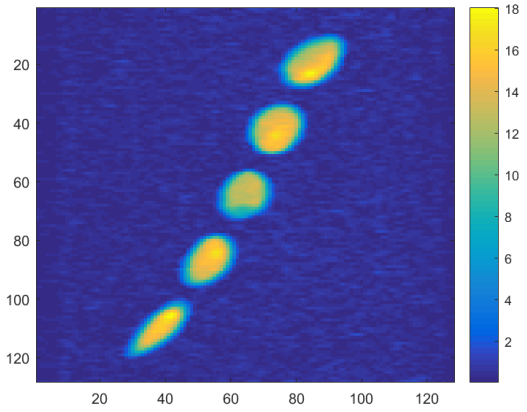
Similar to the previous test case, we find that more noise is removed for larger values of τ. However, the image becomes too smooth while using these larger values.

5.2.2 Regularized total least squares

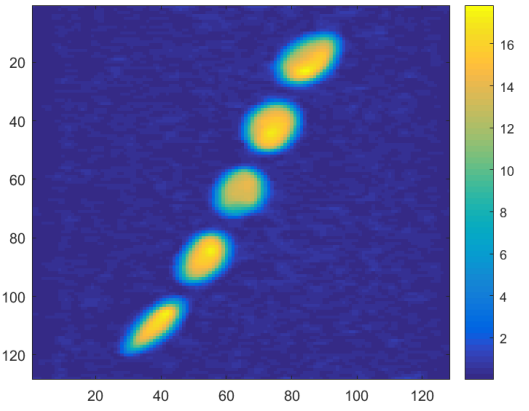
The following results are obtained by applying regularized total least squares to the MRI signal. Again, we choose the same values of the regularization parameter τ.



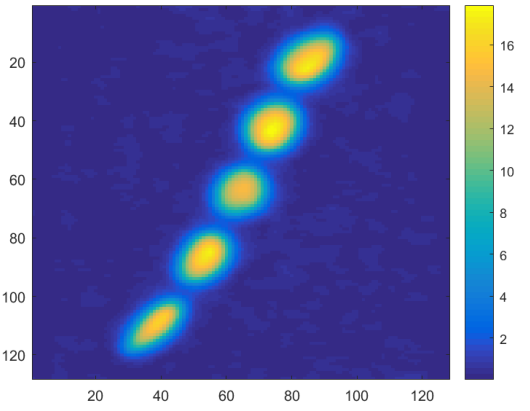
(a) $\tau = 0.1$.



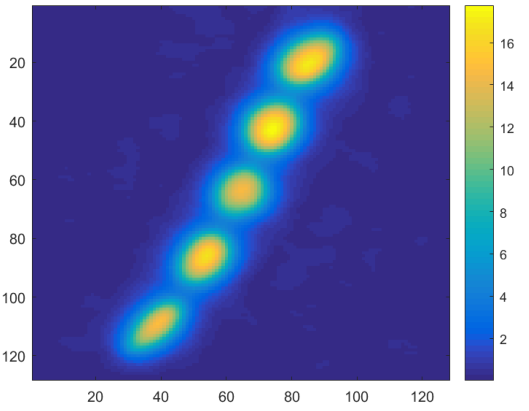
(b) $\tau = 0.5$.



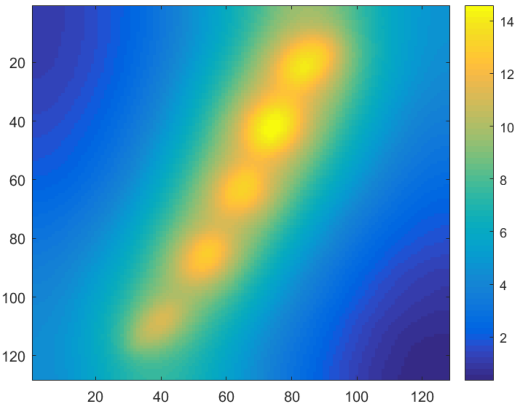
(c) $\tau = 1$.



(d) $\tau = 2$.



(e) $\tau = 3$.



(f) $\tau = 10$.

Figure 12: Results after applying the regularized total least squares method to the test problem Tubes with different regularization parameter values τ.

The differences in the RLS and RTLS solutions are comparable to the differences in results of the Orange test problem. Again, the images obtained by RLS and RTLS only seem to differ up to a constant for smaller values of τ. When $\tau = 3$, a small difference in the amount of noise can be noticed and the image which is obtained by RTLS is smoother. Also, when $\tau = 10$, the solutions clearly differ. It can be observed that the RTLS solution is blurrier than the RLS solution. These results can again be explained by the smallest eigenvalue $\sigma^2_{\min}$ of the matrix C^*C. The values of $\sigma^2_{\min}$ for different values of τ are shown in Table 7.

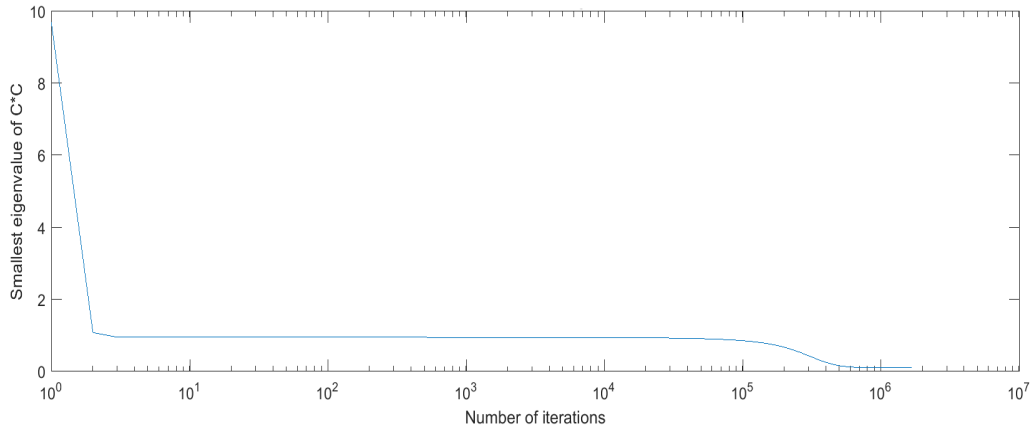
Table 7: The value of the smallest eigenvalue $\sigma^2_{\min}$ of C^*C for different values of the regularization parameter τ.

τ	$\sigma^2_{\min}$
0.1	0.0019
0.5	0.0367
1.0	0.1024
2.0	0.2604
3.0	0.4233
10.0	0.8846

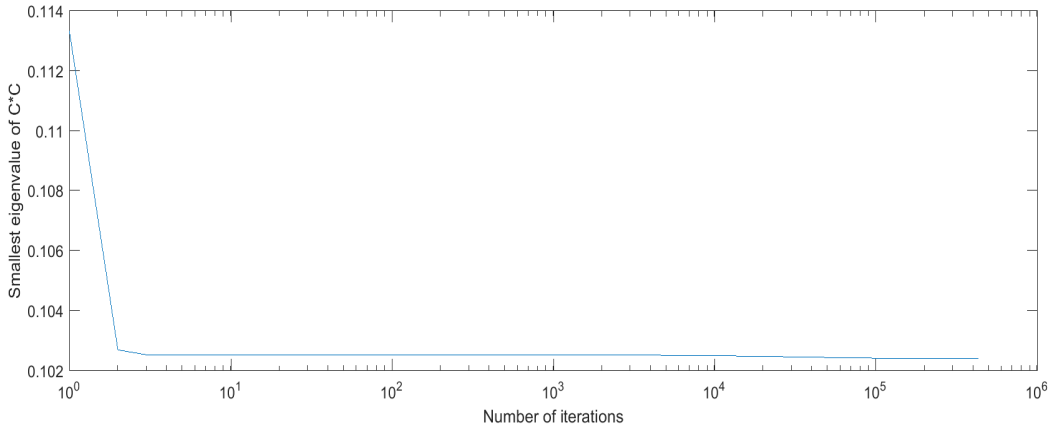
For larger values of τ, it can be observed that $\sigma^2_{\min}$ gets closer to 1. As a result, the RLS and RTLS solutions will (visually) be further apart for larger values of τ, as can be noted from Equations (5.5) and (5.6).

5.2.3 Convergence

Next, we will observe the convergence of the shifted power method to the desired eigenvalue $\sigma^2_{\min}$ of C^*C for $\tau = 1$. We use two different starting vectors, the same as in the previous test case. Hence, we first used the starting vector where every entry is equal to one. Secondly, the starting vector which corresponds to the RLS solution and where the last entry is equal to -1 is used. The shifted power method stops iterating if two consecutively RTLS solutions differ less than 10^{-6}. In Figure 13 the convergence for these two starting vectors is shown.



(a) Convergence of the shifted power method to $\sigma^2_{\min}$ while using the starting vector where every entry is equal to 1.



(b) Convergence of the shifted power method to $\sigma^2_{\min}$ while using the RLS solution as the first entries of the starting vector and where the last entry equal to -1.

Figure 13: Convergence of the shifted power method to the smallest eigenvalue $\sigma^2_{\min}$ of C^*C when $\tau = 1$ for different starting vectors.

Both starting vectors converge to the smallest eigenvalue of approximately 0.1024. As can be observed from Figures 13a and 13b, the starting vector that contains 1 at each entry needs lots of iterations. The RLS vector however only needs a couple of iterations to get very close to the smallest eigenvalue. Hence, the first starting vector experiences a much slower convergence rate than the adjusted RLS starting vector.

5.3 Test case Shepp Logan

The final test problem is the Shepp Logan image. The image that is originally obtained by applying the inverse fast Fourier transform to the obtained MRI signal is as follows.

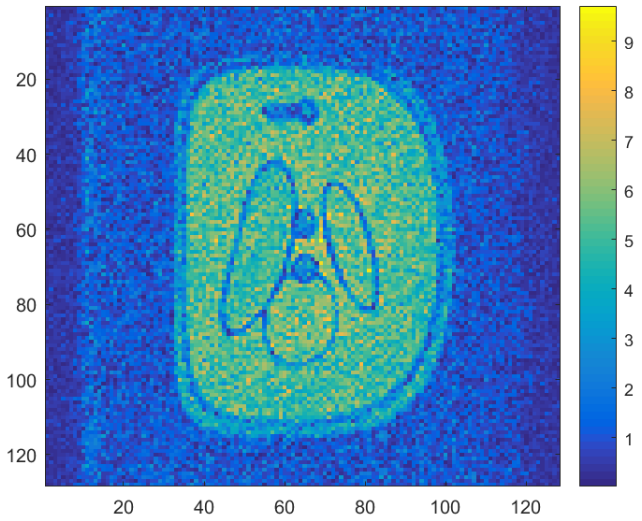


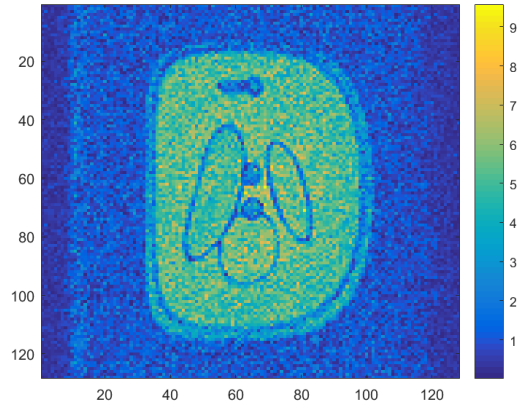
Figure 14: The original MRI image for the test problem Shepp Logan, which is obtained by applying the inverse fast Fourier transform to the measured MRI signal.

This image seems to be equally contaminated by noise compared to the image in Figure 6, which

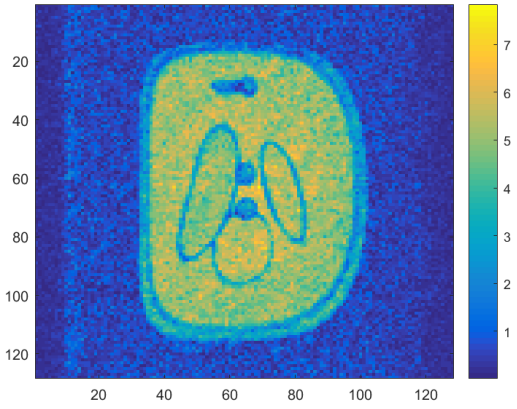
was obtained by the Orange test case. However, the contour of the object in this image is more clearly. Next, we will apply RLS and RTLS to this test case for different values of the regularization parameter τ and observe the results.

5.3.1 Regularized least squares

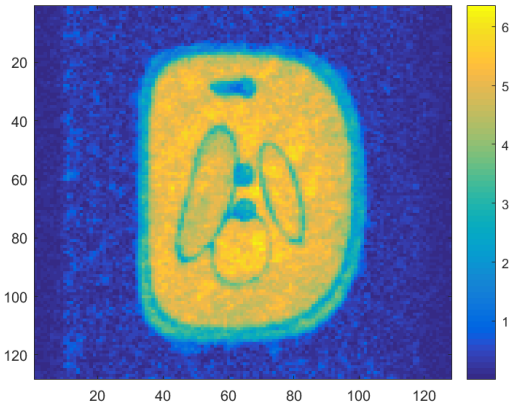
First, we apply RLS to the measured signal. The obtained results for different values of the regularization parameter τ are as follows.



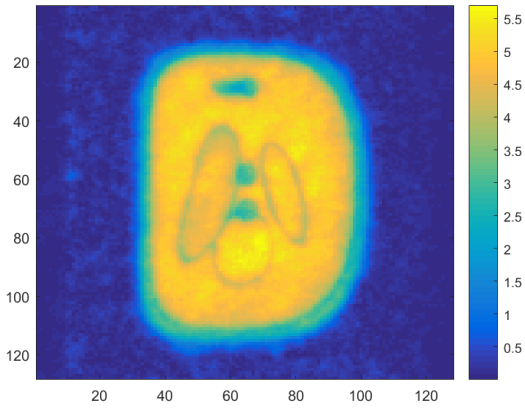
(a) $\tau = 0.1$.



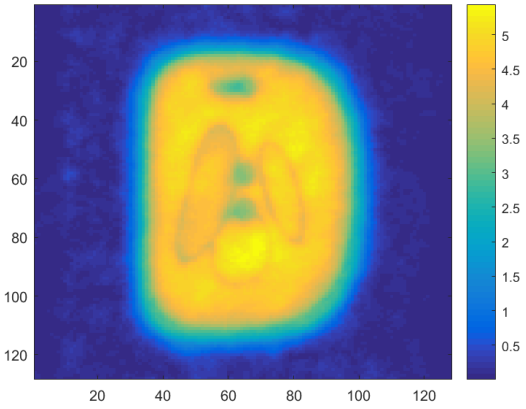
(b) $\tau = 0.5$.



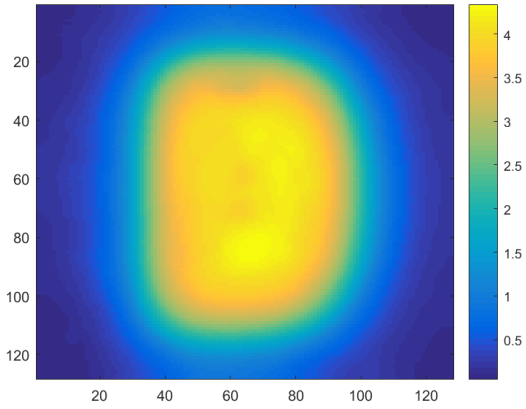
(c) $\tau = 1$.



(d) $\tau = 2$.



(e) $\tau = 3$.



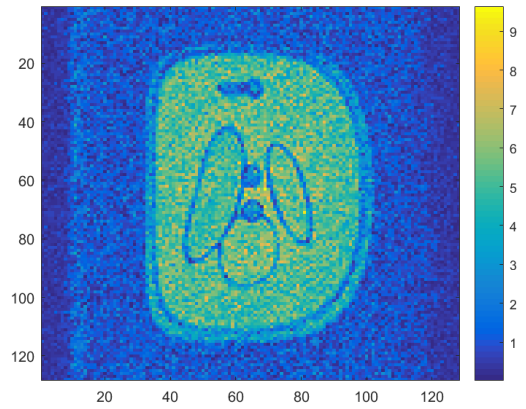
(f) $\tau = 10$.

Figure 15: Results after applying the regularized least squares method to the test problem Shepp Logan with different regularization parameter values τ.

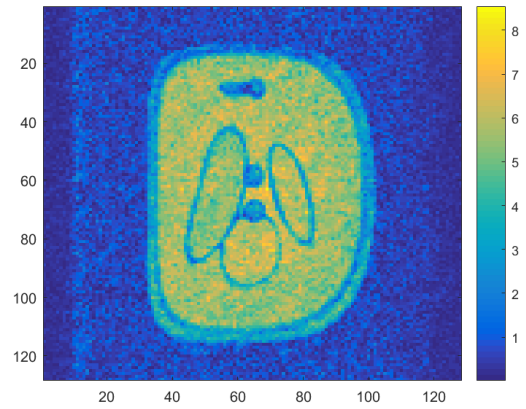
While using the RLS method, a lot of noise can be cancelled out. On the contrary, edges and structures become less clear for larger values of τ.

5.3.2 Regularized total least squares

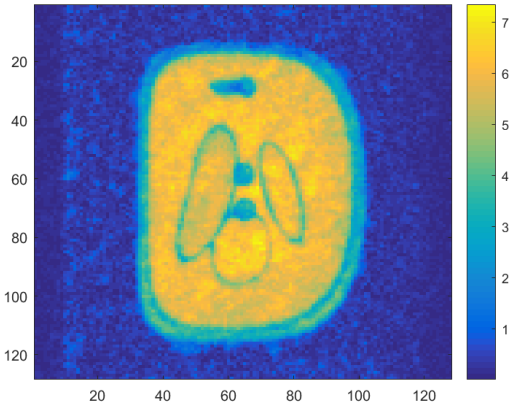
Next, RTLS is applied to the measured signal. The obtained results are shown in Figure 16.



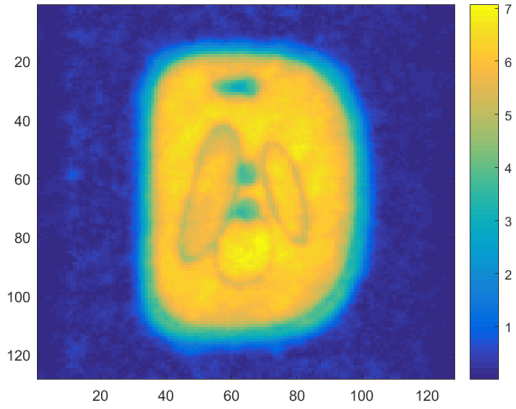
(a) $\tau = 0.1$.



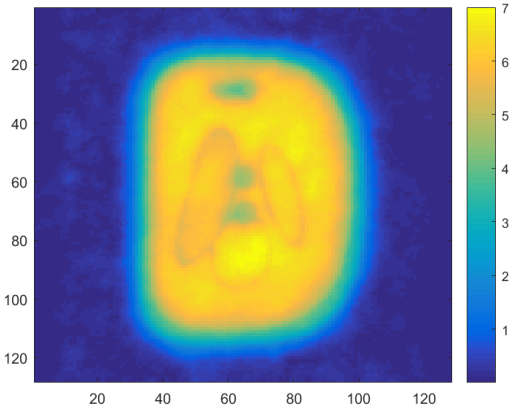
(b) $\tau = 0.5$.



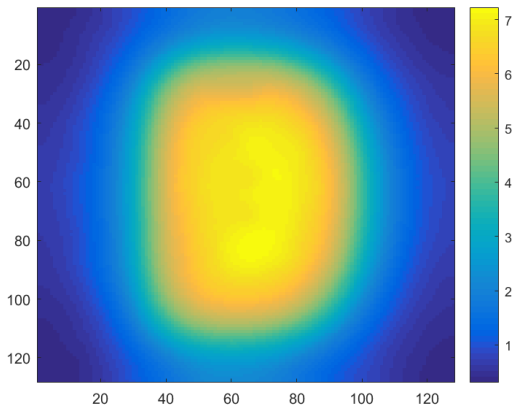
(c) $\tau = 1$.



(d) $\tau = 2$.



(e) $\tau = 3$.



(f) $\tau = 10$.

Figure 16: Results after applying the regularized total least squares method to the test problem Shepp Logan with different regularization parameter values τ.

The differences in the results which are obtained by RLS and RTLS correspond to the differences in results which were found in the previous test cases. Again, the images obtained by RLS and RTLS only seem to differ up to a constant for smaller values of τ. When $\tau = 3$, a small difference in the amount of noise can be noticed and the RTLS image is smoother. Also, the RTLS solution which is obtained when $\tau = 10$, is smoother than the RLS solution. These results can, as in the previous test cases, be explained by the value of the smallest eigenvalue $\sigma^2_{\min}$ of the matrix C^*C. The values for $\sigma^2_{\min}$ for different values of τ are shown in Table 8.

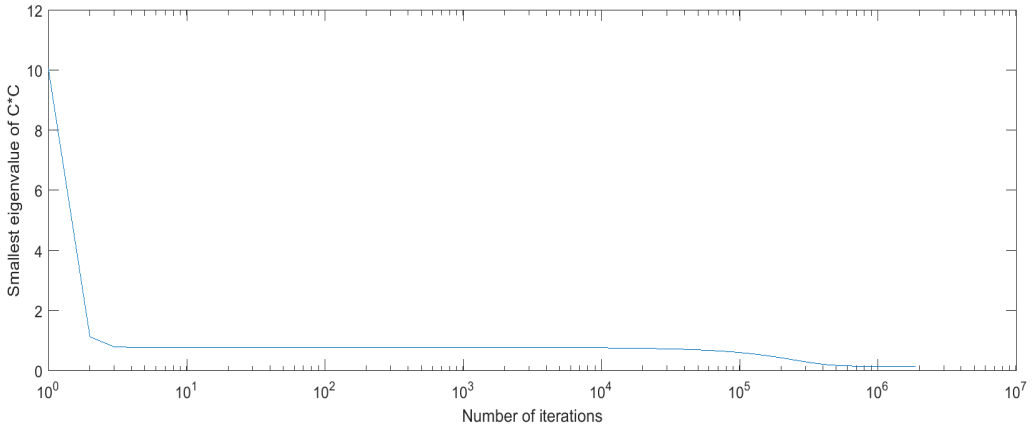
Table 8: The value of the smallest eigenvalue σ^2_{min} of C^*C for different values of the regularization parameter τ.

τ	σ^2_{min}
0.1	0.0070
0.5	0.0914
1.0	0.1534
2.0	0.2035
3.0	0.2366
10.0	0.4669

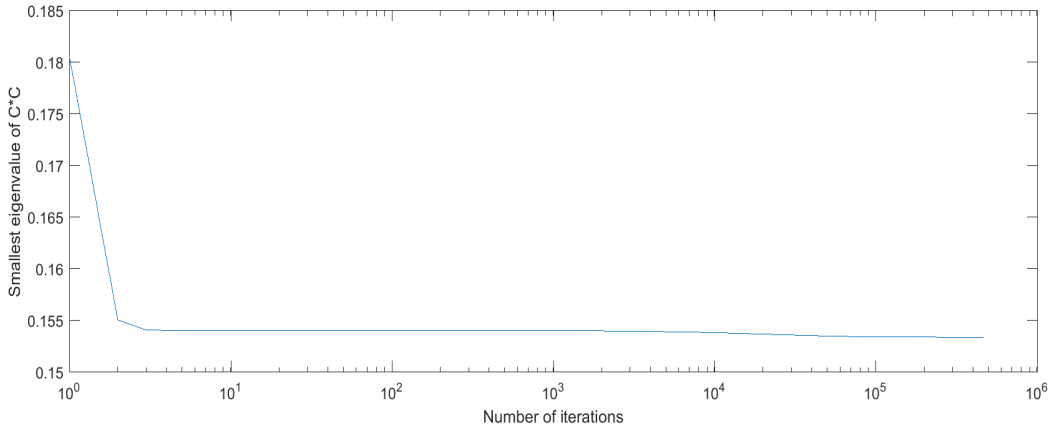
As the value of τ increases, the value of σ^2_{min} gets closer to 1. Therefore, the images which are obtained by RLS and RTLS will differ more when larger values of τ are used, as can be observed from Equations (5.5) and (5.6).

5.3.3 Convergence

Finally, we will observe the convergence of the shifted power method to desired eigenvalue σ^2_{min} of C^*C when $\tau = 1$. The used starting vectors are the same as in the previous test cases. Again, the shifted power method stops iterating if two consecutively RTLS solutions differ less than 10^{-6}. In Figure 17a, the convergence of the initial vector where each entry equals one can be observed. Next, Figure 17b shows the convergence of the adjusted RLS starting vector.



(a) Convergence of the shifted power method to σ^2_{min} while using the starting vector where every entry is equal to 1.



(b) Convergence of the shifted power method to $\sigma^2_{\min}$ while using the RLS solution as the first entries of the starting vector and the last entry equal to -1.

Figure 17: Convergence of the shifted power method to the smallest eigenvalue $\sigma^2_{\min}$ of C^*C for different starting vectors.

Both starting vectors converge to a smallest eigenvalue of approximately 0.1534. As can be observed, the starting vector that contains 1 at each entry requires over a million iterations until the desired eigenvalue is found. Whereas the RLS starting vector needs much less iterations in order to get very close to the smallest eigenvalue. Hence, the first starting vector experiences a much slower convergence rate than the RLS starting vector. This result is similar to the result of the previous test cases. Therefore, the RTLS method seems to converge very slowly to the solution when an arbitrary starting vector is used.

6 Conclusion

In this section we summarise the most important conclusions, formulate answers to our research questions and give recommendations for future research. We start by answering the first research question:

How can the total least squares solution be computed?

In Section 3.2, we observed that the total least squares (TLS) solution of a perturbed system of equations $A\mathbf{x} = \mathbf{y}$, can be computed using the singular value decomposition (SVD). The SVD should be applied to the matrix that contains both the columns of the A and the measurement vector $\mathbf{y}$. It follows that the right singular vector that corresponds to the smallest singular value, equals the TLS solution up to a scaling constant. However, computing the SVD of a very large matrix becomes computationally expensive. Hence, we use the shifted power method to compute the desired singular vector and thereby the TLS solution.

Is the total least squares method of interest to our problem?

As observed in Section 5, the (regularized) total least squares method provides solutions which are very similar to the (regularized) least squares solution. The visual difference between both solutions is only caused by a regularization parameter. Hence, it cannot be concluded what method leads to better results in this case. However, the regularized least squares method is computationally cheaper. As a result, no significant advantages of the total least squares method are found compared to the least squares method in this case.

Finally, we have some suggestions for future research. First, in Section 3.5, we observed the LS and TLS performances to several perturbed systems of equations. The results in Tables 2 and 4 show a sudden increase in the TLS performance for a standard deviation of 0.50 of the Gaussian noise. Furthermore, the test which is generated in Section 3.5.3 led to remarkable results. LS clearly outperformed TLS, while noise was present on both the data matrix and measurement vector. In this case, the noise on the data matrix and measurement vector were both derived from different distributions. Hence, it might be interesting to investigate the performance of TLS in different circumstances concerning the distributional form of the error and the deviation of the noise.

Secondly, as observed in the Section 5, the convergence rate of the (shifted) power method is very slow for starting vectors which are chosen arbitrarily. Hence, other methods might provide faster convergence rates. As the matrix we used to calculate the TLS solution is Hermitian, Lanczos iteration [7] might provide faster convergence to the TLS solution.

Finally, by aggregating pixels, better noise reduction could be achieved at the cost of lowering the resolution of the image. For example, four neighbouring pixels could be reduced to one by taking their average values. Hence, combining this method with TLS, might provide images which are less contaminated by noise.

References

[1] R. Bronson and G. B. Costa. *Linear algebra: an introduction*. Academic Press, 2007.

[2] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[3] J. A. Fessler. Model-based image reconstruction for mri. *IEEE Signal Processing Magazine*, 27(4):81–89, 2010.

[4] G. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 2(2):205–224, 1965.

[5] G. H. Golub, P. C. Hansen, and D. P. O'Leary. Tikhonov regularization and total least squares. *SIAM Journal on Matrix Analysis and Applications*, 21(1):185–194, 1999.

[6] P. C. Hansen. Analysis of discrete ill-posed problems by means of the l-curve. *SIAM review*, 34(4):561–580, 1992.

[7] C. Lanczos. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA, 1950.

[8] Z.-P. Liang and P. C. Lauterbur. *Principles of magnetic resonance imaging: a signal processing perspective*. SPIE Optical Engineering Press, 2000.

[9] I. Markovsky and S. Van Huffel. Overview of total least-squares methods. *Signal processing*, 87(10):2283–2302, 2007.

[10] L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *The quarterly journal of mathematics*, 11(1):50–59, 1960.

[11] S. Van Huffel and J. Vandewalle. *The total least squares problem: computational aspects and analysis*, volume 9. Siam, 1991.

[12] C. Van Loan. *Computational frameworks for the fast Fourier transform*, volume 10. Siam, 1992.

A MRI theory

A.1 Magnetization

As the protons in the magnetic field are precessing, the transverse magnetization rotates according to the Larmor frequency as described in Equation (2.2). This makes it difficult to observe changes in magnetization. To simplify, we introduce an x' and y' axis which both rotate at the same frequency ω_0 as the transverse magnetization. Hence, the direction of both axes change over time. The magnetization from this so called rotating frame of reference, $\vec{M}_{\text{rot}}(\mathbf{r}, t)$, can be defined as follows

$$\vec{M}_{\text{rot}}(\mathbf{r}, t) = M_{x'}(\mathbf{r}, t)\vec{i'} + M_{y'}(\mathbf{r}, t)\vec{j'} + M_{z'}(\mathbf{r}, t)\vec{k'}, \tag{A.1}$$

where $\vec{i'}, \vec{j'}, \vec{k'}$ are defined by

$$\begin{cases} \vec{i'} &= \cos(\omega_0 t)\vec{i} - \sin(\omega_0 t)\vec{j} \\ \vec{j'} &= -\sin(\omega_0 t)\vec{i} + \cos(\omega_0 t)\vec{j} \\ \vec{k'} &= \vec{k} \end{cases}. \tag{A.2}$$

According to Equations (2.4), (A.1) and (A.2), the relation between the rotating and laboratory frame of reference of the transverse magnetization is given by

$$M_{x'y'}(\mathbf{r}, t) = M_{xy}(\mathbf{r}, t)e^{i\omega_0 t}. \tag{A.3}$$

Note that the magnetization in the laboratory frame of reference uses the ordinary x and y axes. Now define

$$\frac{\mathrm{d}\vec{M}}{\mathrm{d}t} = \frac{\mathrm{d}M_x}{\mathrm{d}t}\vec{i} + \frac{\mathrm{d}M_y}{\mathrm{d}t}\vec{j} + \frac{\mathrm{d}M_z}{\mathrm{d}t}\vec{k}, \tag{A.4}$$

$$\frac{\partial \vec{M}_{\text{rot}}}{\partial t} = \frac{\mathrm{d}M'_x}{\mathrm{d}t}\vec{i'} + \frac{\mathrm{d}M'_y}{\mathrm{d}t}\vec{j'} + \frac{\mathrm{d}M'_z}{\mathrm{d}t}\vec{k'}. \tag{A.5}$$

We obtain the following formula, as derived in [8],

$$\frac{\mathrm{d}\vec{M}}{\mathrm{d}t} = \frac{\partial \vec{M}_{\text{rot}}}{\partial t} + \vec{\omega} \times \vec{M}_{\text{rot}}, \tag{A.6}$$

where $\vec{\omega} = -\omega\vec{k}$. Furthermore, $\frac{\mathrm{d}\vec{M}}{\mathrm{d}t}$ describes the rate at which the magnetization changes in the laboratory frame of reference and $\frac{\partial \vec{M}_{\text{rot}}}{\partial t}$ describes this change from a rotating frame of reference. Let $\vec{B}_{\text{rot}}$ be the magnetic field which is experienced from a rotating frame of reference. Combining Equations (2.5) and (A.6) under the assumption that T_1 and $T_2 \to \infty$, we find that

$$\frac{\partial \vec{M}_{\text{rot}}}{\partial t} = \vec{M}_{\text{rot}} \times (\gamma \vec{B}_{\text{rot}} + \vec{\omega}). \tag{A.7}$$

This describes the changing magnetization in time from a rotating frame of reference. From Equations (A.3) and (A.7) it follows that the changing magnetization in the transverse plane right after the RF excitation period is given by the next equation,

$$\frac{\partial}{\partial t} M_{x'y'}(\mathbf{r}, t) = -i\gamma B_z(\mathbf{r}) M_{x'y'}(\mathbf{r}, t), \tag{A.8}$$

where $B_z(\mathbf{r})$ is the part of the (rotating) magnetic field that points into the z-direction. Let $t = 0$ be the time when the RF excitation is completed. One can observe the transverse magnetization for $t > 0$ by solving the above first order linear differential equation. This gives the following formula

$$M_{x'y'}(\mathbf{r}, t) = M_{x'y'}(\mathbf{r}, 0)e^{-i\gamma B_z(\mathbf{r})t}. \tag{A.9}$$

Once the RF excitation period is completed, it no longer holds that $T_1, T_2 \to \infty$. Furthermore, the precessing protons will gradually start precessing out of phase again. As a result the transverse magnetization decreases exponentially with a factor e^{-t/T_2}. From Equation (A.3) it follows that in the laboratory frame of reference the transverse magnetization is given by

$$M_{xy}(\mathbf{r}, t) = M_{xy}(\mathbf{r}, 0)e^{-t/T_2(\mathbf{r})}e^{-i(\gamma B_z(\mathbf{r})+\omega_0)t}. \tag{A.10}$$

A.2 MRI signal

The relaxation time $T1$, as described in Equation (2.5), is generally much longer than $T2$. As a result, $M_z(\mathbf{r}, t)$ is a slowly varying function compared to $M_x(\mathbf{r}, t)$ and $M_y(\mathbf{r}, t)$. Hence, the z component of the magnetization in Equation (2.7) can be neglected and the equation can be rewritten as follows

$$V(t) = -\int_{\text{object}} c(\mathbf{r}) \left(B_x(\mathbf{r}) \frac{\partial M_x(\mathbf{r}, t)}{\partial t} + B_y(\mathbf{r}) \frac{\partial M_y(\mathbf{r}, t)}{\partial t} \right) \, \mathrm{d}\mathbf{r}. \tag{A.11}$$

The components of the magnetic field in the x- and y- direction, B_x and B_y can be written as follows

$$B_x = |\vec{B}_{xy}(\mathbf{r})| \cos(\phi_r) \quad \text{and} \quad B_y = |\vec{B}_{xy}(\mathbf{r})| \sin(\phi_r),$$

with $\phi_r \in [0, 2\pi)$ the reception phase angle. For example, if $\phi_r = 0$, the direction of the magnetic field at position $\mathbf{r}$ is in the x- direction and at $\phi_r = \pi/2$ the field is in the y-direction.

Let $\gamma B_z(\mathbf{r}) = \Delta\omega(\mathbf{r})$ and define $\omega(\mathbf{r}) := \omega_0 + \Delta\omega(\mathbf{r})$. Using Equation (A.10), $M_x(\mathbf{r}, t)$ and $M_y(\mathbf{r}, t)$ can be rewritten as follows

$$M_x(\mathbf{r}, t) = |M_{xy}(\mathbf{r}, 0)|e^{-t/T_2(\mathbf{r})} \cos[-\omega(\mathbf{r})t + \phi_e(\mathbf{r})], \tag{A.12}$$

$$M_y(\mathbf{r}, t) = |M_{xy}(\mathbf{r}, 0)|e^{-t/T_2(\mathbf{r})} \sin[-\omega(\mathbf{r})t + \phi_e(\mathbf{r})]. \tag{A.13}$$

with $\phi_e \in [0, 2\pi)$ the initial phase shift introduced by RF excitation. ϕ_e shows analogous to ϕ_r the direction of the magnetization. Usually $\omega(\mathbf{r}) \gg 1/T_2(\mathbf{r})$, using this assumption and the equations above, we can observe that the time-derivatives of $M_x(\mathbf{r}, t)$ and $M_y(\mathbf{r}, t)$ are approximated by

$$\frac{\partial M_x(\mathbf{r}, t)}{\partial t} = \omega(\mathbf{r})|M_{xy}(\mathbf{r}, 0)|e^{-t/T_2(\mathbf{r})} \sin[-\omega(\mathbf{r})t + \phi_e(\mathbf{r})], \tag{A.14}$$

$$\frac{\partial M_y(\mathbf{r}, t)}{\partial t} = -\omega(\mathbf{r})|M_{xy}(\mathbf{r}, 0)|e^{-t/T_2(\mathbf{r})} \cos[-\omega(\mathbf{r})t + \phi_e(\mathbf{r})]. \tag{A.15}$$

Substituting the above derivations in Equation (A.11), we find that the detected voltage signal is given by

$$V(t) = \int_{\text{object}} c(\mathbf{r})\omega(\mathbf{r})|\vec{B}_{xy}(\mathbf{r})||M_{xy}(\mathbf{r}, 0)|e^{-t/T_2(\mathbf{r})} \cos[-\omega(\mathbf{r})t + \phi_e(\mathbf{r}) - \phi_r(\mathbf{r}) + \frac{\pi}{2}] \, \mathrm{d}\mathbf{r}. \tag{A.16}$$

This voltage is the high-frequency signal that is received by the RF system. Next, the signal $V(t)$ will be moved into a low-frequency band by signal demodulation. In order to do this, the received signal will first be multiplied by two different reference signals, that is $2\cos(\omega_0 t)$ and $2\sin(\omega_0 t)$. After this, a technique known as low-pass-filtering can be applied in order to create a low-frequency signal. As a result, the demodulated signal $S(t)$ is obtained. Multiplying $V(t)$ by the first reference signal yields

$$\begin{aligned} 2\cos(\omega_0 t)V(t) = &\int_{\text{object}} c(\mathbf{r})\omega(\mathbf{r})|\vec{B}_{xy}(\mathbf{r})||M_{xy}(\mathbf{r}, 0)|e^{-t/T_2(\mathbf{r})} \cos[-\omega(\mathbf{r})t - \omega_0 t + \phi_e(\mathbf{r}) - \phi_r(\mathbf{r}) + \frac{\pi}{2}] \, \mathrm{d}\mathbf{r} \\ &+ \int_{\text{object}} c(\mathbf{r})\omega(\mathbf{r})|\vec{B}_{xy}(\mathbf{r})||M_{xy}(\mathbf{r}, 0)|e^{-t/T_2(\mathbf{r})} \cos[-\omega(\mathbf{r})t + \omega_0 t + \phi_e(\mathbf{r}) - \phi_r(\mathbf{r}) + \frac{\pi}{2}] \, \mathrm{d}\mathbf{r}. \end{aligned}$$

Next, low-pass-filtering can be applied in order to create a low-frequency signal. Low-pass-filtering leads to the removal of the first term. Hence,

$$V_1(t) = \int_{\text{object}} c(\mathbf{r})\omega(\mathbf{r})|\vec{B}_{xy}(\mathbf{r})||M_{xy}(\mathbf{r},0)|e^{-t/T_2(\mathbf{r})}\cos[-\omega(\mathbf{r})t+\omega_0 t+\phi_e(\mathbf{r})-\phi_r(\mathbf{r})+\frac{\pi}{2}]\ \mathrm{d}\mathbf{r}. \quad \text{(A.17)}$$

$V_1(t)$ is now a low frequency signal. By using the reference signal $2\sin(\omega_0 t)$, a similar low frequency signal $V_2(t)$ is found. Hence,

$$V_2(t) = \int_{\text{object}} c(\mathbf{r})\omega(\mathbf{r})|\vec{B}_{xy}(\mathbf{r})||M_{xy}(\mathbf{r},0)|e^{-t/T_2(\mathbf{r})}\sin[-\omega(\mathbf{r})t+\omega_0 t+\phi_e(\mathbf{r})-\phi_r(\mathbf{r})+\frac{\pi}{2}]\ \mathrm{d}\mathbf{r}. \quad \text{(A.18)}$$

Next, the obtained signal can be written as $S(t) = V_1(t) + iV_2(t)$, where i denotes the complex number. As a result, we obtain the following signal

$$S(t) = \int_{\text{object}} c(\mathbf{r})\omega(\mathbf{r})|\vec{B}_{xy}(\mathbf{r})||M_{xy}(\mathbf{r},0)|e^{-t/T_2(\mathbf{r})}e^{-i[\Delta\omega(\mathbf{r})t-\phi_e(\mathbf{r})+\phi_r(\mathbf{r})-\frac{\pi}{2}]}\ \mathrm{d}\mathbf{r}. \quad \text{(A.19)}$$

First, we assume that $\Delta\omega(\mathbf{r}) \ll \omega_0$. Then omitting the scaling factor $\omega_0 e^{\frac{\pi}{2}i}$ and using the following observations

$$|\vec{B}_{xy}(\mathbf{r})|e^{-i\phi_r(\mathbf{r})} = \overline{\vec{B}_{xy}(\mathbf{r})} \quad \text{and} \quad |M_{xy}(\mathbf{r},0)|e^{i\phi_e(\mathbf{r})} = M_{xy}(\mathbf{r},0),$$

we obtain the signal as described in Equation (2.8).

B Matlab codes

B.1 Function: tls

```
function x = tls(A,b)
    C = [full(A) b];
    [U,S,V] = svd(C,0);

    [m,n] = size(A);
    [m,r] = size(b);

    V12 = V(1:n, n+1:end);
    V22 = V(n+1:end, n+1:end);

    if det(V22) ~= 0
        x = -V12*(V22\eye(size(V22)));
    end
end
```

B.2 Polynomial regression

This code makes use of the function tls(A,b) which is given in Appendix B.1.

```
t = 0.2*[-10:9].';
A = [ones(length(t),1), t];
xsol = [0.7,1].';
b = A*xsol;

numberOfExperiments = 200000;
r = [1,0.5,0.1,0.05,0.01];

countxls = zeros(length(r),1);
countxtls = zeros(length(r),1);

for i=1:length(r)
    for j = 1:numberOfExperiments
        bn = b + r(i)*randn(size(b));
        An = A + r(i)*randn(size(A));

        xlsA = An\bn;
        xtlsA = tls(An,bn);

        errxlsA = norm(xlsA - xsol);
        errxtlsA = norm(xtlsA -xsol);

        if errxlsA > (errxtlsA + 10^(-10))
             countxtls(i) = countxtls(i) + 1;
        end

        if errxtlsA > (errxlsA + 10^(-10))
             countxls(i) = countxls(i) + 1;
```

```
        end
    end
end

countxls = (countxls/numberOfExperiments)*100;
countxtls = (countxtls/numberOfExperiments)*100;
```

B.3 Exponential regression

This code makes use of the function tls(A,b) which is given in Appendix B.1.

```
numberOfExperiments = 200000;
r = [1,0.5,0.1,0.05,0.01];

countxlsA = zeros(length(r),1);
countxtlsA = zeros(length(r),1);
countxlsf = zeros(length(r),1);
countxtlsf = zeros(length(r),1);

t = linspace(0,1,175);
f = [0:1:6];

for n=1:length(r)
    for m = 1:numberOfExperiments

    fn = f + r(n)*randn(size(f));

        for i=1:length(t)
            for j=1:length(f)
                A(i,j) = exp(2*pi*1i*f(j)*t(i));
                Af(i,j) = exp(2*pi*1i*fn(j)*t(i));
            end
        end
        xsol = [2,1.5,4,2.5,3.5,2,1].';
        b  = A*xsol;

        bn = b + r(n)*(1+1i)*randn(size(b));
        An = A + r(n)*(1+1i)*randn(size(A));

        xlsA = An\bn;
        xlsf = Af\bn;
        xtlsA = tls(An,bn);
        xtlsf = tls(Af,bn);

        errxlsA = norm(xlsA - xsol);
        errxtlsA = norm(xtlsA -xsol);

        if errxlsA > (errxtlsA + 10^(-10))
            countxtlsA(n) = countxtlsA(n) + 1;
        end
```

```
        if errxtlsA > (errxlsA + 10^(-10))
            countxlsA(n) = countxlsA(n) + 1;
        end

        errxlsf = norm(xlsf - xsol);
        errxtlsf = norm(xtlsf -xsol);
        if errxlsf > (errxtlsf + 10^(-10))
            countxtlsf(n) = countxtlsf(n) + 1;
        end
        if errxtlsf > (errxlsf + 10^(-10))
            countxlsf(n) = countxlsf(n) + 1;
        end
    end
end
```

B.4 Power method

```
load(b);
x = ifft2c( b );

n = length(b);
lambda = 1; %regularisatie parameter

Dx = spalloc(n^2,n^2,3*n);
Dy = spalloc(n^2,n^2,3*n);
for j = 1:n
    for i = 1:n
        k = i + (j-1)*n;
        if i < n
            Dx(k,k) = 1;
            Dx(k,k+1) = -1;
        end
        if j < n
            Dy(k,k) = 1;
            Dy(k,k+n) = -1;
        end
   end
end

xls = (speye(n^2) + lambda^2*Dx'*Dx + lambda^2*Dy'*Dy)\reshape(ifft2c(b),n^2,1);
xls = reshape(xls,n,n);

p = ifft2c(b);
p = reshape(p,n^2,1);

%Gershgorin:
shift = max(1+8*lambda^2 + max((abs(p))),
                              sum(abs(p)) + reshape(b,n^2,1)'*reshape(b,n^2,1));

%startvector power method [qouda; qoudb]
```

```
qouda = reshape(xls,n^2,1);
qoudb = -1;

%qouda = ones(n^2,1);
%qoudb = 1;

marge = 10^(-6);
verschil = 1 + marge;
maxIter = 2000000;
eigval = zeros(10^7,1);
while verschil > marge && (iter < maxIter)
    qa = qouda + lambda^2*Dx'*(Dx*qouda) + lambda^2*Dy'*(Dy*qouda) + p*qoudb;
    qb = p'*qouda +  reshape(b,n^2,1)'*reshape(b,n^2,1)*qoudb;
    qa = qa - shift*qouda;
    qb = qb - shift*qoudb;
    normq = norm([qa;qb]);
    qa = qa/normq;
    qb = qb/normq;

    verschil = (norm(((-1/qb)*qa) - ((-1/qoudb)*qouda)));

    eigval1 =[qouda;qoudb]'*([qa;qb]*normq);
    eigval2 = [qouda;qoudb]'*[qouda;qoudb];
    eigval(iter + 1) = eigval1/eigval2;

    qouda = qa;
    qoudb = qb;
end

xtls = (-1/qb)*qa;
xtls = reshape(xtls,n,n);
```