Numerical Strategies for the Solution of Inverse Problems

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Geophysics
The University of British Columbia
129-2219 Main Mall
Vancouver, Canada
V6T 1Z4

Date:

Most people, if you describe a train of events to them, will tell you what the result would be. They can put those events together in their minds, and argue from them that something will come to pass. There are few people, however, who, if you told them a result, would be able to evolve from their own inner consciousness what the steps were which led up to that result. This power is what I mean when I talk of reasoning backwards.
Sherlock Holmes to Dr. Watson in: A Study in Scarlet,

Sir Arthur Conan Doyle, 1887.

Abstract

This thesis deals with the numerical solutions of linear and nonlinear inverse problems. The goal of this thesis is to review and develop new techniques for solving such problems. In so doing, the computational tools for solving inverse problems are comprehensively studied.

The thesis can be divided into two parts. In the first part, linear inverse theory is dealt with. Methods to estimate noise and efficiently invert large and full matrixes are reviewed and developed. Emphasis is given to Generalized Cross Validation (GCV) for noise estimation, and to Krylov space methods for efficient methods to invert large systems. This part is summarized by applying and comparing the methods developed on linear inverse problems which arise in gravity and tomography.

In the second part of this thesis, extensive use of the linear algebra and the noise estimation methods which were developed in the first part of the thesis is made. A review of the current methods to carry out nonlinear inverse problems is given. A test example is constructed to demonstrate that these methods may fail. Next, a new algorithm for solving nonlinear inverse problems is developed. The algorithm is based on the ability to differentiate between correlated errors which comes from the linearization, and non-correlated noise which comes from the measurement. Based on these two types of noise, a regularization procedure which has two parts is developed. The first part is made of global regularization, to deal with the measurement noise, and the second part is made from a local regularization, to deal with the nonlinearity. The thesis demonstrates that GCV can be used in order to determine the measurement noise, and the Dumped Gauss-Newton method can be used in order to deal with the local nonlinear terms. Another

aspect of nonlinear inverse theory which is developed in this work concerns approximate sensitivities. A new formulation is suggested for the approximate sensitivities and bounds are calculated using this formulation. This part is summarized by applying the techniques to the nonlinear gravity problem and to the magnetotelluric problem.

Table of Contents

List of Tables

List of Figures

Acknowledgement

Many people helped me putting up with this dissertation. However I would like to give special thanks to two people: my wife, Yonat, who supported me through these years, and my advisor, Doug Oldenburg. Doug has taught me not only about inverse theory, but also how to develop scientific thinking, and more importantly, how to keep an open mind. His help was not limited to the science; he also supported my personal goals and ideas. I learned many things from him as a human being.

This thesis is a result of inter-disciplinary research and many of the ideas came through discussions with my committee members. Jim Varah, Ken Whittall, Michael Bostock and Uri Ascher. I would like to thank Jim and Uri for pointing me the direction and giving useful ideas through personal discussion and through the SCV group meetings. Ken deserves special thanks for many interesting discussions about inverse theory, and for proofreading and correcting my thesis. People outside my committee who gave me good ideas and constructive criticism are Phil Loewen and Brian Wetton.

Other people who helped me are my fellow students and Post Doctoral fellows. I would like to thank Chen Greif, Mauricio Sacchi, Yaguo Li, Colin Farquharson and Partha Routh for their encouragement and ideas.

Special thanks deserve all the people who made the work in the Geophysics department of UBC so pleasant: the glaciology group, especially Jinj Flowers and Shawn Marshall, Dave Hildes and Yuval.

Chapter 1

Introduction

This thesis deals with the numerical treatment of ill-posed inverse problems. Such problems are solved on a daily basis in many disciplines such as geophysics, medical physics, image and signal processing and astrophysics. Most inverse problems arise in the physical world, and their solutions are used by geologists, medical doctors and others as a studying tool. The field involves people from different disciplines and it is inter-disciplinary in its nature.

A simple way to visualize an inverse problem is to imagine we are given a black box and we would like to find out its contents. We are allowed to carry out experiments and try and guess the contents of this box, however we are not allowed to open it. In inverse problems, we call the contents of the box "the model", the result of an experiment, "the data", and the experiment itself is referred to as "the forward modeling". Usually an experiment cannot provide sufficient information to determine a unique model, i.e., there could be more than one model which would produce the same data. In order to select the most reasonable model, we use the process of regularization. Regularization produces a model which satisfies some specific criteria. Since the person interpreting the results of an inverse problem is usually not the same one who made the computations, it is important to integrate between the field of application and the computational field, and to let the computational method be as flexible as possible. While this process works reasonably well for small problems, large problems demand special attention and thus many of the well-known computational techniques are not utilized. The two goals of this thesis are

first to bring existing knowledge which has been developed in numerical methods into the field of applied inverse problems and secondly, to develop new methodologies and algorithms when existing methods are insufficient.

1.1 Ill-Posed Inverse Problems

The concept of an ill-posed problem is not new. Hadamard [1923] defined a problem as being ill-posed if the solution was non-unique, or if the solution was not continuous with respect to the data, i.e., a small change in the data leads to a large change in the model. Hadamard did not deal with the numerics of ill-posed problems as he believed that the ill-posedness arose from an incorrect physical representation of the problem. Tichonov [1963] was the first to deal numerically with ill-posedness, and in so doing introduced the concept of regularization. Tichonov wanted the model to remain stable while producing a smooth approximation to the data, even if the data changed by only a small amount. In his work, regularization was introduced to stabilize the problem. Parker [1977 a] changed the way regularization is viewed, by introducing a weighted regularization in order to obtain a model which not only honours the data, but also possesses some specific desired features. His approach was used and extended by Twomey [1977], Oldenburg [1984], Menke [1984] and others.

This thesis deals with constructing discretized regularized solutions to inverse problems. This is only a part of the whole field of inverse theory. Other avenues available for obtaining information about an underlying model from observed data are appraisal (Backus and Gilbert [1968], [1970]), and inference problems. These are are not dealt with in this thesis. The concept of regularization to obtain a specific model is often called construction. Construction can be viewed in two ways: the deterministic approach, and the probabilistic approach. While the underline philosophy of these two approaches is

different, in many cases the final equations which need to be solved are similar. This thesis, in general, does not deal with the probabilistic approach. My main focus is on numerical methods used for solving the linear or nonlinear equations and the phrase "solving the inverse problem" relates to solving the regularized equations.

While the concept of regularization is well understood today, the main problem is its implementation in large problems. In recent years significant advances have been made in the field of linear inverse problems by the work of Hansen [1992 a,b], [1995], Hanke and Hansen [1993], Oldenburg *et al.* [1991], [1993], [1994], Scales [1987], Scales *et al.* [1987], [1990], [1994], Parker [1994], Parker and Whaler [1981], Nolet and Snieder [1990], and others. However there are still a number of unanswered questions, and more importantly, there is insufficient understanding as to which method should be used for a specific problem. In the field of nonlinear inverse problems there are far more advances to be made. Firstly there is a need to incorporate techniques from linear inverse theory and computational methods from optimization theory. Current algorithms for the solution of nonlinear ill-posed problems employ parameters and use heuristics which are not well explained. Therefore, there is a need to add rigor to these algorithms and to suggest better algorithms.

1.2 Motivation - Problems which are Solved

The aim of this thesis is to present practical algorithms for the solution of inverse problems and therefore it is tied to real world examples and real data sets. The thesis deals with a number of problems. Gravity is a commonly used technique in geophysics, and can be viewed as an archetypal problem, since it leads to large, dense matrix system of equations. Gravity data from a one-dimensional model is used to illustrate and compare linear algorithms in inverse problems. Two and three-dimensional gravity problems are

used in order to illustrate computational properties of large problems. A field data set is used as a test example for the three-dimensional gravity problem. The gravity problem suffers from non-uniqueness with respect to depth and therefore, in some cases, it can be made nonlinear by reformulating the problem in terms of interfaces. This formulation increases the depth resolution of the method, but compromises its linearity. The nonlinear gravity problem is used to test linear imaging methods and nonlinear methods.

A second type of problem which is solved in this thesis is tomography. In contrast to gravity, tomography problems lead to sparse systems with better resolution. Borehole tomography is a commonly used method in the area of geophysics and a field data set is used to test different methods. In medical physics, a commonly used tomography experiment is the Single Photon Emission Computed Tomography (SPECT). This thesis shows how the inversion of SPECT data can be improved by using new techniques. These techniques are tested on a data set obtained from a patient at Vancouver General Hospital. Finally in order to test techniques for nonlinear inverse problems, the magnetotelluric (MT) example is used. The MT problem is an archetypal problem in electromagnetics methods in geophysics.

In the next subsection short descriptions of these inverse problems are presented.

1.2.1 The Gravity Inverse Problem

A surface gravity survey is carried out to measure the anomalous gravitational acceleration in the vertical direction. From Newton's law we know that a unit mass positioned at $(x_i, y_i, 0)$ on the surface of the earth is attracted to a mass anomaly density, $\Delta\rho$, in position (x, y, z) inside the earth as:

$$\Delta g_i \hat{r} = \Delta g(x_i, y_i, 0)\hat{r} = \gamma \frac{\Delta\rho(x, y, z)\ dx\ dy\ dz}{(x - x_i)^2 + (y - y_i)^2 + z^2}\ \hat{r} \tag{1.1}$$

where γ is the gravitational constant and $\hat{r}$ is a unit vector pointing from the observation location, $(x_i, y_i, 0)$, to (x, y, z). Using this relation we can describe four inverse problems.

Three-Dimensional Gravity Problem

In three dimensions, the anomalous mass can be anywhere at the region D. The gravity anomaly at a point height h above the surface would then be:

$$\Delta g_i \hat{r} = \Delta g(x_i, y_i, h)\hat{r} = \gamma \int_D \frac{\Delta\rho(x, y, z)\ \hat{r}}{(x - x_i)^2 + (y - y_i)^2 + (z + h)^2}\ dx\ dy\ dz \tag{1.2}$$

However, we usually do not measure the gravity field itself, but rather only its vertical component. The data are therefore:

$$b_i = b(x_i, y_i, h) = \gamma \int_D \frac{\Delta\rho(x, y, z)\ z}{[(x - x_i)^2 + (y - y_i)^2 + (z + h)^2]^{\frac{3}{2}}}\ dx\ dy\ dz \tag{1.3}$$

The goal of this problem is to recover $\Delta\rho(x, y, z)$ from the measured data in the $x - y$ plane.

Two-Dimensional Gravity Problem

Here it is assumed that the model is only two-dimensional, i.e., $\Delta\rho = \Delta\rho(x, z)$ is independent of y. We can integrate equation 1.3 with respect to y and get:

$$b_i = b(x_i, h) = \gamma \int_D \frac{\Delta\rho(x, z)\ z}{(x - x_i)^2 + (z + h)^2} dx\ dz \tag{1.4}$$

The goal of this problem is to recover the model $\Delta\rho(x, z)$ from the measured data in the x direction.

One-Dimensional Gravity Problem

Now assume that we are at the edge of a fault ($x = 0$) and measure the gravity data on the surface ($h = 0$) as plotted in Figure 1.1. On our left there is a uniform half-space

with a known density and on our right there is a layered density which depends only on z. In this case we can divide the integral 1.4 into two quarter spaces. The integration

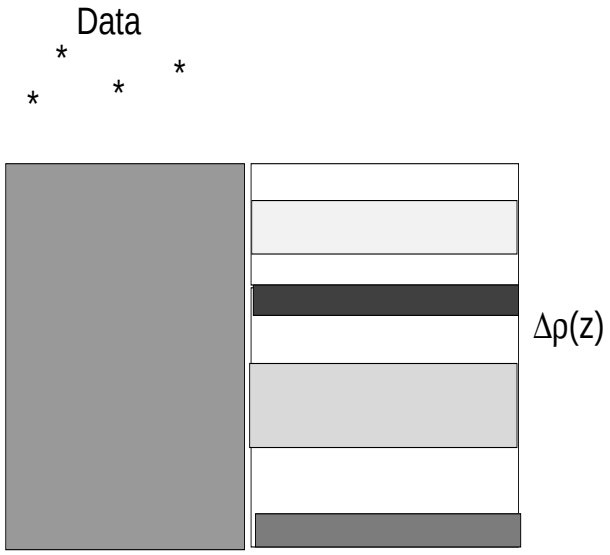

Figure 1.1: The 1-D gravity data example

over the first quarter space produces zero since its density anomaly is zero. We therefore have to carry out the integration over x:

$$b(x_i, 0) = \gamma \int_{z=0}^{\infty} \int_{x=0}^{\infty} \frac{\Delta\rho(z)\ z}{(x - x_i)^2 + z^2} dx\ dz \tag{1.5}$$

which gives

$$b(x_i, 0) = \gamma \int_{z=0}^{\infty} (\frac{\pi}{2} - atan(\frac{x_i}{z}))\Delta\rho(z)\ dz \tag{1.6}$$

The goal of this problem is to recover the one-dimensional profile $\Delta\rho(z)$ from the data in the x direction.

Nonlinear Gravity Problem

It is possible to solve the gravity inverse problem as a linear inverse problem. However, for some situations a different formulation is more suitable.

Assume that we measure the gravitational field on the earth's surface. The gravity anomaly in the z direction is related to the density of the earth according to equation 1.3. Assume now that the earth has two layers with known densities and the density contrast between them is $\Delta\rho$. The first layer has a mean depth of $h(x,y)$ which is assumed to be known. Changes in the gravitational field are due to a change in the depth of the first layer relative to h. This change is given by the function $m(x,y)$. In this case we can write:

$$b_i = \gamma \int\int_D \int_h^{h+m(x,y)} \frac{z\ \Delta\rho}{[(x-x_i)^2+(y-y_i)^2+z^2]^{\frac{3}{2}}}\ dx\ dy\ dz \tag{1.7}$$

Integrating the expression with respect to z gives:

$$b_i = \tilde{\gamma} \int\int_D \frac{1}{r_{h_i}} - \frac{1}{r_{m_i}}\ dx\ dy \tag{1.8}$$

where $\tilde{\gamma} = \Delta\rho\gamma$, $r_{h_i}^2 = (x-x_i)^2+(y-y_i)^2+h^2$ and $r_{m_i}^2 = (x-x_i)^2+(y-y_i)^2+(h+m)^2$.

The goal of the inverse problem presented in this example is to recover the surface $m(x,y)$ from the given gravity anomalies b_j. In order to do this we would also need the Fréchet derivatives of the operator. By differentiating equation 1.8 with respect to m we get:

$$\frac{\partial b_j}{\partial m} = \tilde{\gamma} \int\int_D \frac{(h+m(x,y))}{[(x-x_j)^2+(y-y_j)^2+(h+m(x,y))^2]^{\frac{3}{2}}}\ dx\ dy \tag{1.9}$$

1.2.2 The Tomography Inverse Problem

A very different type of inverse problem arises in tomography. Although the physics of geophysical borehole tomography differs from that of medical tomography, the equations are similar. The primary difference between the equations is the geometry of the

experiment. While geophysical tomography is limited in view to the boreholes and the surface, medical tomography is unlimited in its view and can observe the object from every direction.

Radio Imaging Borehole Tomography

Radio Imaging - (RIM) is a high frequency EM survey which is used to obtain information about electrical conductivity. A transmitter, in this case a vertical magnetic dipole, is deployed in one borehole and a receiver coil is in another. Ray theory is adopted and the source signal is assumed to travel along a straight line connecting the source and receiver. The amplitude of the EM wave at the receiver is given by

$$(A_r)_i = \frac{A_0}{r_i} e^{-\int_{l_i} m(x,z)dl(x,z)} \tag{1.10}$$

were A_0 is the amplitude at the source, r_i is the distance between the transmitter and the receiver, l_i is the ray path and $m(x,z)$ is an attenuation coefficient which depends upon the conductivity of the medium. Taking the logarithm of equation 1.10 yields a linear relationship between the attenuation coefficient and the data:

$$b_i = log(A_r)_i - log(A_0) + log(r_i) = -\int_{l_i} m(x,z)dl(x,z) \tag{1.11}$$

The goal of this experiment is to estimate the attenuation $m(x,z)$ from the discrete data b_i. Note that the data do not have physical dimensions.

Single Photon Emission Computed Tomography - SPECT

A very different experiment, which can nevertheless be described by very similar mathematics, is SPECT. This experiment is used on a daily basis in almost any large hospital. The experiment involves administering a dose of radioactive material to the patient. This radioactive material is attached to a molecule which targets a specific functional area,

such as the heart or the brain. Active areas will emit radiation, and thus the experiment provides information on the biochemical activity of the body.

In order to collect data, a special crystal, which detects photons, is placed in collimated bins in a plane around the patient. A schematic of the experiment is shown in Figure 1.2. The number of photons b_i which is detected at bin i, at an angle θ, is proportional

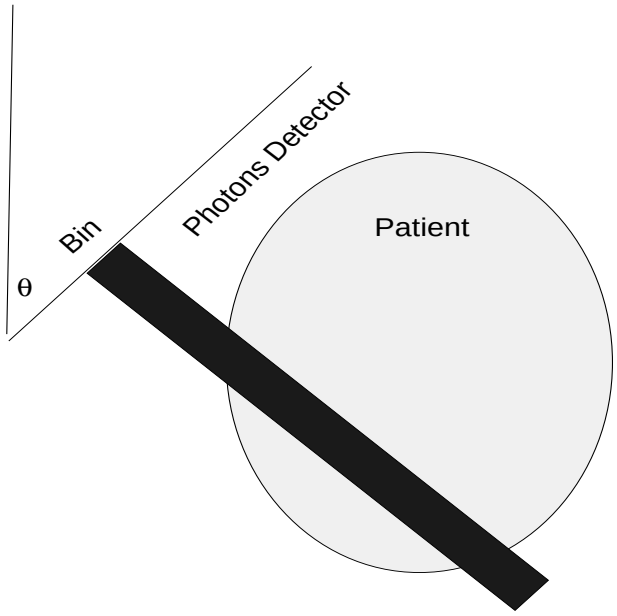

Figure 1.2: The geometry of the SPECT experiment. The data at the bin is proportional to the amount of radioactive material which is inside the black ray

to the amount of radioactive material $m(x, y)$ which is viewed by this bin. This can be written as:

$$b_i(\theta) = \int_{l_i(\theta)} m(x, y) dl(x, y) \tag{1.12}$$

The experiment is done for different θ's and finally we have the data set $b_i(\theta_j)$.

1.2.3 One-Dimensional Magnetotellurics Problem

The one-dimensional magnetotelluric (MT) problem is a well-studied nonlinear problem in electromagnetics, (see for example Parker [1994], Dosso [1990], Whittall and Oldenburg [1992]) and therefore it is very suitable as a test example. We start with Maxwell's equations in the frequency domain:

$$\nabla \times \mathcal{E} = -i\omega\mu_0 \mathcal{H} \tag{1.13}$$

$$\nabla \times \mathcal{H} = \sigma \mathcal{E}$$

where $\mathcal{E}$ and $\mathcal{H}$ are the three-component electric and magnetic field intensities, μ_0 is the magnetic permeability, which is assumed to be constant, ω is the angular frequency and σ is the electric conductivity.

Assuming that $\mathcal{E}$ and $\mathcal{H}$ are plane waves, i.e.

$$\mathcal{E} = (E, 0, 0) \qquad \mathcal{H} = (0, H, 0)$$

and that $\sigma = \sigma(z)$, we can take the curl of the first equation in 1.13 and get:

$$\frac{d^2E}{dz^2} = i\omega\mu_0\sigma(z)E \tag{1.14}$$

The boundary conditions for this equation are (see Parker [1994]):

$$E(\infty) = 0$$

$$E(0) = 1$$

This is the governing equation for the MT experiments. The data for this experiment are:

$$c_0(\omega, \sigma(z)) = -\frac{E(z=0,\omega)}{\partial_z E(z=0,\omega)} = -\frac{1}{\partial_z E(z=0,\omega)} \tag{1.15}$$

Our goal is to recover the conductivity profile $\sigma(z)$ from the complex measurements c_0.

It was proved by Parker [1977 b] that the data c_0 are Fréchet differentiable with respect to the conductivity $\sigma(z)$:

$$J(\omega, \sigma(z))(.) = (\frac{\partial c_0(\omega, \sigma(z))}{\partial \sigma}, .) = i\omega\mu_0 \int_0^\infty c^2(\sigma(z), z, \omega)(.)dz \tag{1.16}$$

where:

$$c(\sigma(z), z, \omega) = -\frac{E(z, \omega)}{\partial_z E(z = 0, \omega)}$$

1.3 Overview of this Thesis

As stated earlier, the aim of this thesis is to give numerical treatment for inverse problems. An overview of the basic principles and techniques in numerical methods is given, which is then used to motivate and build algorithms. The thesis is divided into three main parts. In the first part (Chapters 2-6) linear inverse theory is treated. Chapters 2 and 3 are a basic review of the formulation of ill-posed problems and Tichonov regularization. In Chapter 3 a short review of two methods, Generalized Cross Validation and the L-curve, are provided. A new interpretation is presented for the L-curve technique and a connection is established with the probabilistic approach to inverse problems. Chapter 4 discusses subspace methods. It provides an overview of the truncated singular value decomposition, conjugate gradient least-squares, least-squares QR as well as an explanation for the regularization properties of Krylov space methods. As part of the subspace formulation, a new multilevel method to solve ill-posed problems is developed. Another technique for solving linear problems which is reviewed is a gradient-based method. Finally, in Chapter 4, Generalized Cross Validation is extended for subspace type algorithms. This work is new and it is a building block of a strategy to solve nonlinear problems.

Chapter 5 covers the combination of subspace methods and Tichonov regularization.

This type of regularization is often referred to as hybrid type regularization. Two algorithms are developed for this purpose. First, least-squares QR is utilized for hybrid regularization, and then a second new Krylov space method is developed. This technique deals with the loss of orthogonality which is the main problem of the Krylov spaces. The last part of Chapter 5 deals with a gradient-based method for hybrid regularization which was developed by Oldenburg *et al.* [1993], [1994].

In Chapter 6, the algorithms developed in the previous chapters are tested using the examples presented in Section 1.2. At the end of this chapter a short discussion about which method should be used for the solution of linear ill-posed problems is presented.

Chapters 7-10 cover the second part of this thesis: nonlinear inverse theory. Chapter 7 serves as a review of the formulation of nonlinear inverse problems, formulation of the solution, review of minimization techniques and a review of current algorithms for the solution of nonlinear ill-posed problems. In this chapter an example is provided to show that one of the most common algorithms used for nonlinear inverse problems can fail.

Chapter 8 is dedicated to the development of computational techniques for nonlinear inverse problems. A new method for solving nonlinear inverse problems, based on Generalized Cross Validation and Damped Gauss-Newton steps is developed, along with a discussion on the calculations of such solutions when the problem is large.

Chapter 9 reviews some of the methods to approximate sensitivities using cord update, Approximate Inverse Mapping (AIM) and secant update. A new global concept of approximate sensitivities is developed. This new concept can be used to provide a bound on the solution obtained by using approximate sensitivities.

Chapter 10 is a summary of the nonlinear chapters. The algorithms developed in Chapters 7-9 are tested on two nonlinear inverse problems, the nonlinear gravity and the MT problems, which were presented in Sections 1.2.2 and 1.2.3.

Chapter 2

Formulation and Pre-Processing of the Problem

This chapter contains the basic tools for the formulation of linear inverse problems which are used later for the nonlinear case and it is mainly a review of the work of Elden [1977], [1982], [1990], Hansen [1995], Engl *et al.* [1996]. We review the discretization process and the transformation of inverse problems into a standard form. These two stages of pre-processing of the problem are important because they allow similar treatment for different problems.

2.1 Formulation of Linear Problem

Most inverse problems describe the continuous world, with an origin in Fredholm integral equations of the first kind. Let $\mathcal{H}$ be a Hilbert space, let $m(t)\in\mathcal{H}$ be the model and let $b = [b_1 ... b_N]^T$ be a vector of the measured data. This thesis deals with problems where the connection between m and b is:

$$b_i \;=\; \int_D K(s_i,t)m(t)dt \;+\; \epsilon_i \tag{2.1}$$

Where $K(s,t)$ is a smooth kernel (i.e. the kernel does not posses singularities), ϵ_i is the measurement noise assumed to be approximately Gaussian and D is the domain of integration. Our goal is to find the model m given the noisy data b.

The first question which might be asked is how to discretize the system. There are two approaches to the problem. The first is to discretize it with a number of parameters M which is smaller than N. In this case problem 2.1 becomes a well-posed least-squares

system. This approach is taken on a regular basis in medical physics but its major disadvantage is that it imposes regularization on the problem by making the solution lie in a small subspace which does not necessarily fit the problem. Since $m \in \mathcal{H}$, it is better to discretize the system as finely as possible such that the discrete system possesses some of the characteristics of the continuous system. In this work the system is discretized with a number of parameters $M > N$.

The discretization can be carried out by two main methods. First it is possible to approximate the integral 2.1 by some quadrature rule:

$$\int_D K(s_j, t) m(t) dt \approx \sum_{i=1}^{M} w_i K(s_j, t_i) m(t_i) \Delta t_i \tag{2.2}$$

which leads to the rectangular system:

$$b \;=\; A\,x \;+\; \epsilon \tag{2.3}$$

where $A_{ji} = w_i K(s_j, t_i) \Delta t_i$ and $x = m(t_i)$ is a vector in R^M.

A different method for discretization uses Galerkin methods. Let $\psi_i(s)$, $i = 1...M$ be an orthonormal set of basis functions. If the solution $m = \sum_{i=1}^{M} x_i \psi_i(s)$ then, the integral can be written as:

$$\int_D K(s_j, t) m(t) dt = \sum_{i=1}^{M} x_i \int_D K(s_j, t) \psi_i(t) dt \tag{2.4}$$

which again gives a system of equations:

$$b \;=\; A\,x \;+\; \epsilon \tag{2.5}$$

where $A_{ji} = \int_D K(s_j, t) \psi_i(t) dt$ and x is a vector of coefficients.

The choice of discretization method is problem dependent. While quadrature methods are somewhat easier since they need only the estimate of the kernel at some points, Galerkin methods may be more accurate and require fewer unknowns to obtain the same accuracy.

In both cases the matrix $A : R^M \rightarrow R^N$ is typically ill-conditioned and the data contain noise, therefore regularization is needed for the solution of the problem. In this thesis three possible types of regularization are analysed:

- Tichonov regularization, which involves the definition of a norm or a semi-norm $||Wx||$ and looking for a solution of a global objective function:

$$minimize \quad \phi(\beta, x) = ||Ax - b||^2 \; + \; \beta||Wx||^2 \tag{2.6}$$

  The minimization problem is equivalent to the solution of the least-squares system:

$$\begin{bmatrix} A \\ \sqrt{\beta}W \end{bmatrix} x = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{2.7}$$

  This type of regularization is analysed in Chapter 3.

- Subspace regularization, which involves the definition of a k dimensional subspace S_k, $k < N$ and solving the minimization problem:

$$minimize \quad \phi_d(x) = ||Ax - b||^2 \quad x \in S_k \tag{2.8}$$

  If we let $S_k = span(V_k)$ then $x = V_k z$ and the minimization problem is equivalent to the solution of the least-squares system:

$$AV_k z = b \tag{2.9}$$

  This type of regularization is analysed in Chapter 4.

- Hybrid regularization, which involves the definition of a norm or a semi-norm $||Wx||$ and a subspace S_k, $k < N$ and solving the minimization problem:

$$minimize \quad \phi(\beta, x) = ||Ax - b||^2 + \beta||Wx||^2 \quad x \in S_k \tag{2.10}$$

If we let $S_k = span(V_k)$ then $x = V_k z$ and the minimization problem is equivalent to the solution of the least-squares system:

$$\begin{bmatrix} AV_k \\ \sqrt{\beta} W V_k \end{bmatrix} z = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{2.11}$$

This type of regularization is analysed in Chapter 5.

2.2 Characteristics of Inverse Problems

Most inverse problems have common characteristics. Perhaps the most important one is the behaviour of the spectrum. Let

$$A = USV^T \tag{2.12}$$

be the singular value decomposition (SVD) of A, where $V = [v_1 ... v_M]$, is an $M \times N$ matrix which spans the active part of the model space, $U = [u_1 ... u_N]$ is an $N \times N$ matrix which spans the data space and $S = diag(\lambda_1 ... \lambda_N)$ with singular values $\lambda_1 > \lambda_2 > ... \lambda_N \geq 0$ is a diagonal matrix of size $N \times M$. Although it is impossible to prove in general, the large singular values are associated with smooth singular vectors, v, and the small singular values are associated with oscillatory vectors. This characteristic is of major importance when dealing with inverse problems. It can also be shown (Sterling and Porter [1990]) that the singular value decomposition is closely related to the singular function expansion of the operator $\int_D K(s,t)(.)dt$. In most inverse problems the model is assumed to be smooth, and this thesis deals only with this kind of model. Smooth models contain mostly vectors which are associated with large singular values. This characteristic of the model is also used extensively throughout this thesis.

2.3 Reduction to Standard Form

The Tichonov formulation of the inverse problem discussed above is regularized with an arbitrary norm. This leads to some complications in the analysis of the methods, and later on, to problems with implementation of the regularization especially in subspace and hybrid methods. It was therefore suggested to transform 2.6 into the standard form which is:

$$minimize \quad \phi_s(\beta, x) = ||Ax - b||^2 \; + \; \beta||x||^2 \tag{2.13}$$

This transformation is discussed by Elden [1977], [1982], [1990] and Engl *et al.* [1996] and we review it here with some modification. There are three possible cases: W is square and well-posed, W is rectangular and over determined and W is rectangular and under-determined.

The matrix W can represent either *a priori* knowledge about the problem (such as in Li and Oldenburg [1996]) or a general regularizer to get a specific type of solution for example a smooth one. In this work the matrix is the discretization of the continuous operator:

$$W(.) = \begin{bmatrix} \alpha_1 \nabla^2(f_1(t)(.)) \\ \alpha_2 \nabla(f_2(t)(.)) \\ \alpha_3 f_3(t)(.) \end{bmatrix} \tag{2.14}$$

where $f_i(t) \;\; i = 1, 2, 3$, are semi-positive functions. This could lead to an over-determined W. Another possibility is to choose W such that it corresponds to a differential operator of the form:

$$W(.) = \alpha_1 \nabla^2(f_1(t)(.)) + \alpha_2 \nabla(f_2(t)(.)) + \alpha_3 f_3(t)(.) \tag{2.15}$$

This could lead to a square or an underdetermined W. The underdetermined case appears if the differential operators do not include boundary conditions. In this case the operators are calculated only for the points which are inside the domain. The discretization of the

differential operators is done using central differences and, in one dimension we use the stencil $[-1, 1]$ or $[-1, 0, 1]$ for the first derivative and the stencil $[1, -2, 1]$ for the second derivative. We now discuss the three possible situations.

2.3.1 W is Square and Well-Posed

For this simple case we let $\tilde{x} = Wx$ and rewrite 2.6 as:

$$\phi(\beta, \tilde{x}) = ||AW^{-1}\tilde{x} - b||^2 \ + \ \beta||\tilde{x}||^2$$

Define $\tilde{A} = AW^{-1}$. The problem is now in its standard form with the operator $\tilde{A}$ instead of A.

2.3.2 W is Over-Determined

The over-determined case is only slightly more complicated than the square case. In this case:

$$W = \begin{bmatrix} \alpha_1 W_1 \\ \alpha_2 W_2 \\ . \\ . \\ \alpha_p W_p \end{bmatrix}$$

is a $pM \times M$ size matrix, and has a unique generalized inverse $W^\dagger$ which obeys:

$$W^\dagger W = I$$

This enables us to write the minimization problem as:

$$\phi(\beta, \tilde{x}) = ||AW^\dagger\tilde{x} - b||^2 \ + \ \beta||\tilde{x}||^2$$

where again $\tilde{x} = Wx$, and the matrix A is replaced by $AW^\dagger$. Notice that for this case, the matrix $AW^\dagger$ is $N \times pM$ and the transformed model $\tilde{x}$ is of size pM.

Another possibility for the transformation in this case is to form the matrix $Q = W^T W$ and to use the Cholesky decomposition in order to obtain an upper triangular $M \times M$ matrix $\tilde{W}$ such that $\tilde{W}^T \tilde{W} = Q$. If we take this approach then we go back to the case that W is square.

The choice between the method of transformation depends on the size of the problem and the method of solution. For small and medium sized problems it might be easy to carry out the Cholesky decomposition of Q and store the matrix $\tilde{W}$. However for very large scale applications where the matrix $W^T W$ could not be decomposed, we use iterative methods for the solution of the inverse problem, and need only the application of $W^\dagger$ on a vector, v. In this case it might be better to calculate $W^\dagger v$ when needed and not compute and store $\tilde{W}$.

2.3.3 W is Under-Determined

The case which is the most difficult is the case of W under-determined. The main reason is that if we try to set $\tilde{x} = Wx$ then x cannot be recovered by using the generalized inverse $W^\dagger$ because W has a nontrivial null space, i.e. $W^\dagger W \neq I$.

For this case let W be a $P \times M$ matrix with a pseudo-inverse $W^\dagger$ and let W_0 be an $M \times (M - P)$ size matrix which contains the null space of W, i.e. $WW_0 = 0$. Usually P is close to M and the null space, which contains only $M - P$ independent vectors, is relatively small. The solution x can be divided into two parts, one which is in the active space of W, and one which is in the null space of W:

$$x = x_W + x_0 = W_A^\dagger \tilde{x} + W_0 \tilde{x}_0$$

where $\tilde{x}$ is a P-vector and $\tilde{x}_0$ is $M - P$ vector. The matrix $W_A^\dagger$ is of size $M \times P$ which projects any P-vector to the active space of W. The goal of this decomposition is to choose $W_A^\dagger$ such that the problem is transformed into the standard form. Substituting

this decomposition in 2.7 gives:

$$\begin{bmatrix} A \\ \sqrt{\beta}W \end{bmatrix} [W_A^\dagger \tilde{x} + W_0 \tilde{x}_0] = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{2.16}$$

This is equivalent to these two systems equations:

$$AW_A^\dagger \tilde{x} + AW_0 \tilde{x}_0 = b \tag{2.17}$$

$$WW_A^\dagger \tilde{x} + WW_0 \tilde{x}_0 = WW_A^\dagger \tilde{x} = 0$$

From these two equations we can get $W_A^\dagger$ which transforms the problem into the standard form. First, $WW_A^\dagger$ should give the identity, which is similar to the over-determined and the square case. While this character of $W_A^\dagger$ takes care of active space of W in the solution it does not take care of the null space of W. Since $\tilde{x}_0$ appears only in the first equation of 2.17 it could be expressed as:

$$\tilde{x}_0 = (AW_0)^\dagger b - (AW_0)^\dagger AW_A^\dagger \tilde{x} \tag{2.18}$$

where $(AW_0)^\dagger$ is the generalized inverse of AW_0. This expression has two parts: $(AW_0)^\dagger b$ which depends only on the data, b, and $(AW_0)^\dagger AW_A^\dagger \tilde{x}$ which depends on $\tilde{x}$. If $\tilde{x}_0$ is not dependent on $\tilde{x}$ then the problem is separable and we could solve for $\tilde{x}_0$ first and then get an equation for $\tilde{x}$ which is in the standard form. This happens only if:

$$(AW_0)^\dagger AW_A^\dagger = 0$$

If $W_A^\dagger$ is equal to the generalized inverse of W, $W^\dagger$, then $WW^\dagger = I$, but $(AW_0)^\dagger AW^\dagger \neq 0$ and therefore this choice is not a good one. Elden [1982] has proposed that:

$$W_A^\dagger = W^\dagger - W_0(AW_0)^\dagger AW^\dagger \tag{2.19}$$

This choice is good since:

$$WW_A^\dagger = W(W^\dagger - W_0(AW_0)^\dagger AW^\dagger) = WW^\dagger - WW_0(AW_0)^\dagger AW^\dagger = I - 0 = I$$

and

$$(AW_0)^\dagger AW_A^\dagger = (AW_0)^\dagger A(W^\dagger - W_0(AW_0)^\dagger AW^\dagger) =$$

$$(AW_0)^\dagger AW^\dagger - (AW_0)^\dagger AW_0(AW_0)^\dagger AW^\dagger = (AW_0)^\dagger AW^\dagger - (AW_0)^\dagger AW^\dagger = 0$$

The matrix $W_A^\dagger$ is often referred to as the A weighted generalized inverse of W. Using the matrix $W_A^\dagger$, W_0, x_0 and $\tilde{x}$, the problem can be reformulated. First x_0 is found:

$$x_0 = W_0(AW_0)^\dagger b \tag{2.20}$$

then we are left with the standard form for the P vector $\tilde{x}$:

$$\begin{bmatrix} \tilde{A} \\ \sqrt{\beta} I \end{bmatrix} \tilde{x} = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{2.21}$$

where $\tilde{A} = AW_A^\dagger$ is an $N \times P$ matrix.

2.3.4 Practical Implementation

From a practical point of view it is useful to separate iterative methods, where only products of the form $\tilde{A}v$ and $\tilde{A}^T u$ are calculated, from direct methods where $\tilde{A}$ is actually calculated.

First we discuss the cases of W well-posed or over-determined. In iterative methods we calculate products of the form $AW^{-1}u$ or $AW^\dagger u$ and $(W^T)^{-1}A^T v = W^{-T}A^T v$ or $(W^\dagger)^T A^T v$. These operations are divided into two parts, first the solution of the system $Wp = u$ is calculated, then the multiplication of Ap is calculated. The transpose is done by first calculating the product $p = A^T v$ and then solving the system $W^T q = p$. If W is given by an elliptic differential operator the solution of the systems $Wp = u$ and $W^T q = v$ can be done in $\mathcal{O}(M)$ operations using multigrid methods as suggested by Dendy [1982]. The problem with multigrid solvers is that they are generally complicated for general purpose algorithms. In this thesis we have used a direct LU solution for small

scale systems and BICG, BICG-STAB or GMRES with incomplete LU preconditioning (as suggested in Saad [1996]) for the solutions of the differential systems. In general if the differential system is well-conditioned, the number of operations is of order $M^{1+\gamma}$ (with $\gamma \approx 0.3$) which is usually satisfactory even for large scale problems (30000-100000 unknowns). Solving the inverse problem typically requires at least $\mathcal{O}(kNM)$ operations (where k is some integer and hopefully $k \ll N$) and therefore the reduction to standard form does not impose major difficulty.

If we choose to use a direct method then the calculation of $\tilde{A}$ is done for each row separately. Let a_k^T be the k^{th} row of A and let $\tilde{a}_k^T$ be the k^{th} row of $\tilde{A}$. The matrix $\tilde{A} = AW^{-1}$ and therefore:

$$\tilde{A}^T = W^{-T} A^T$$

For each vector a_k and $\tilde{a}_k$ we can write:

$$W^T \tilde{a}_k = a_k \tag{2.22}$$

Thus $\tilde{a}_k$ is calculated a row at a time, and the calculation of $\tilde{A}$ takes $\mathcal{O}(M^{1+\gamma}N)$ operations.

When W is underdetermined, the product $W_A^{\dagger} v$ and $(W_A^{\dagger})^T u$ is somewhat more difficult. First we need to obtain the null space of W. Let

$$W = [W_1, W_2]$$

where W_1 is a $P \times P$ matrix and W_2 is a $P \times (M - P)$ thin matrix. The null space of W can be found by:

$$W_0 = \begin{bmatrix} W_1^{-1} W_2 \\ -I_{M-P} \end{bmatrix}$$

Using this matrix we form the $N \times (M - P)$ matrix:

$$A_0 = AW_0$$

This matrix is usually small and can be stored. Using it we calculate the component of the solution which is in the null space of W:

$$x_0 = (A_0)^\dagger b$$

and the residual

$$b_0 = b - Ax_0$$

assuming again that the operation of the generalized inverse of W on a vector is easy to obtain, the product of $W_A^\dagger v$ and $(W_A^\dagger)^T u$ is done sequentially. First we calculate the product: $s = W^\dagger v$, then calculate $s_0 = W_0((AW_0)^\dagger(As))$ and finally subtract $s - s_0$ to get the result. The transpose is carried out in the opposite order.

If we choose to work with direct methods then again the calculation of $\tilde{A}$ is done for each row separately. This is done by solving the system:

$$W^T \tilde{a}_k = a_k \tag{2.23}$$

with the A weighted generalized inverse.

In general, in order to make the reduction to standard form efficient, fast solvers to the systems $Wq = v$ and $W^T p = u$ are needed. If such solvers are not available then the reduction to standard form should be avoided. This would make Tichonov regularization more expensive and it would give an advantage to full space methods.

Chapter 3

Tichonov Regularization

In 1963 Tichonov proposed that the solution x should not fit the data exactly. The problem of finding a solution to 2.3 is transformed into a minimization problem:

$$minimize \quad \phi(\beta, x) = ||Ax - b||^2 \; + \; \beta||Wx||^2 = \phi_d \; + \; \beta\phi_m \tag{3.1}$$

The function $\phi(\beta, x)$ which is used extensively in this thesis is referred to as the global objective function, ϕ_d is the data misfit function and ϕ_m is the model objective function. The parameter β is a penalty parameter which determines how well the solution should fit the data. As β becomes large the solution fits less well to the data and as β becomes very small, the solution starts to fit noise. β is adjusted such that the solution fits the data in some optimal way. This is discussed in Sections 3.2-3.4.

The solution to the minimization problem 3.1 is achieved by differentiating with respect to x and forcing it to zero. This gives:

$$(A^T A + \beta W^T W)x = A^T b \tag{3.2}$$

The system, 3.2, is also equivalent to the least-squares system:

$$\begin{bmatrix} A \\ \sqrt{\beta}W \end{bmatrix} x = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{3.3}$$

3.1 Analysis of Tichonov Regularization

In order to understand the basic properties of regularization it is often useful to look at the spectrum of the operator and the way it is affected by the specific regularization

method. Let:

$$A = USV^T \tag{3.4}$$

be the singular value decomposition (SVD) of A described in section 2.2. Using the SVD the system 3.2 (assuming in standard form) can be written as:

$$(A^TA + \beta I)x = (VS^2V^T + \beta I)x = V(S^2 + \beta I)V^Tx = VSU^Tb$$

Multiplying both sides in V^T gives:

$$(S^2 + \beta I)V^Tx = SU^Tb$$

and

$$x = V(S^2 + \beta I)^{-1}SU^Tb = VS^{-1}(S^2 + \beta I)^{-1}S^2U^Tb$$

This can also be written as:

$$x = VS^{-1}D_TU^Tb \tag{3.5}$$

where $D = (S^2 + \beta I)^{-1}S^2$. In vector format this can be written as:

$$x = \sum_{i=1}^{N} f_T(\lambda_i)\frac{b^Tu_i}{\lambda_i}v_i \tag{3.6}$$

where f_T is the Tichonov filter function:

$$f_T(\lambda) = \frac{\lambda^2}{\lambda^2 + \beta} \tag{3.7}$$

Figure 3.1 shows the Tichonov filter for different β's. The filter penalizes vectors which are associated with $\lambda^2 \ll \beta$. Thus the role of Tichonov regularization is to damp the singular vectors which are associated with small singular values. This process is fundamental to regularization of inverse problems.

3.2 Discrepancy Principle

While the idea of regularization is clear, so far we have not discussed how much to regularize. In the Tichonov regularization case it is clear that if the penalty parameter β is too small, the model fits noise and if the regularization parameter is too large then some of the original signal is damped. If the noise level is known then the regularization parameter can be determined. This principle is often refered to as the discrepancy principle (Engl *et al.* [1996]).

Assume that the noise is Gaussian with mean 0 and standard deviation σ i.e. $\epsilon \propto N(0, \sigma)$. The true solution x_r obeys:

$$\phi_d = ||b - Ax_r||^2 = ||\epsilon||^2 = \sum_{i=1}^{N} \epsilon_i^2 = \sigma^2 \sum_{i=1}^{N} (\frac{\epsilon_i}{\sigma})^2$$

The last sum is simply a sum of N squared Gaussian random variables with mean zero and unit standard deviation. This sum is another random variable which can be described by the χ^2 distribution function. The expected value of this variable is simply N and therefore the expected value of the misfit is:

$$\phi_d{}^* = \sigma^2 N \tag{3.8}$$

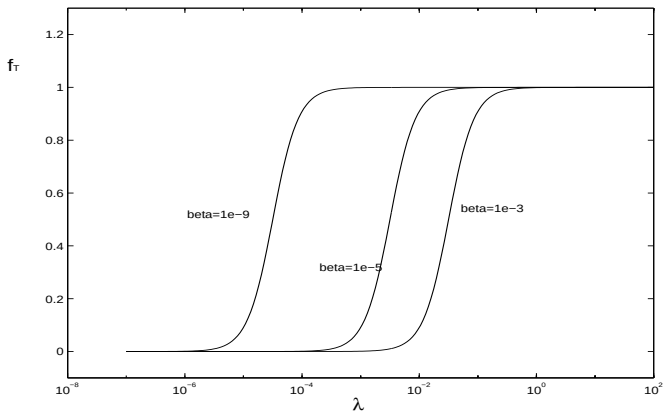

Figure 3.1: Tichonov filter function

According to the discrepancy principle the right β is found by solving:

$$||b - Ax(\beta)||^2 = ||(I - A(A^TA + \beta I)^{-1}A^T)b||^2 = N\sigma^2 \tag{3.9}$$

This is a nonlinear equation in one unknown β which can be solved in various ways. Techniques to solve this equation have been proposed by Parker [1994], Engl *et al.* [1996] and Golub and Von-Matt [1991]. The major cost of the solution of such problem is the calculation of the function, which requires the inversion or decomposition of $A^TA + \beta I$.

The discrepancy principle is also closely related to constraint minimization. In this case the inverse problem is formulated as:

$$minimize \quad ||Wx||^2 \tag{3.10}$$

$$subject\ to \quad ||Ax - b||^2 \leq T$$

Using this formulation a Lagrangian is formed:

$$\mathcal{L}(\beta, x) = \beta^{-1}(||Ax - b||^2 - T) \ + \ ||Wx||^2 \tag{3.11}$$

The Lagrangian is then minimized over x and maximized over β, and the stationary point with respect to β is a saddle point. In the linear case differentiating the equation with respect x and β gives exactly the same equations as Tichonov regularization 3.2. The Tichonov formulation is often referred to as a penalty formulation and although the final equations are identical, the philosophy of the penalty and the Lagrangian formulations are different. While the Lagrange formulation forces us to define a target misfit T the penalty formulation is more general and allows other possibilities for the choice of regularization parameter, such as the model norm or some relation between the misfit and model norm. In this thesis the formulation which is used is the penalty one because it is more general and leads to simpler algorithms from non-constrained optimization, although parallel explanations to many of the processes can be found in constrained optimization.

3.3 Generalized Cross Validation

While there is a statistical estimate of the misfit when the noise level is known, real data rarely come with standard deviations. The main reason is that geophysical experiments are costly and rarely repeated more then once. Most applications ignore this difficulty by guessing the noise level, which is, of course, not a very good idea. One of the more successful methods to deal with this problem is the Generalized Cross Validation (GCV) and we review it in this section.

The major idea of GCV is that a good model could predict new data points. We cannot go to the field and measure a new datum each time we try a new regularization parameter to verify our solution and therefore we simulate the experiment by eliminating one datum from the data set. A good solution of the reduced data set, should predict this datum fairly well even if it was not used when calculating the model. This idea is repeated for each datum and therefore the model obtained in this way is the model which can predict most of the data points even if these data points are not used.

In mathematical language, this is done by introducing the following notation. Let $x_k(\beta)$ minimize:

$$\phi_k = ||Ax - b||^2 - (a_k^T x - b_k)^2 + \beta||x||^2 \tag{3.12}$$

where a_k^T is the k^{th} row of A and b_k is the k^{th} data point. Notice that ϕ_k is the same objective function as 3.1 but with the k^{th} data point and equation missing. We can now ask, how well is the k^{th} datum predicted when it is not used? This can be measured by:

$$(a_k^T x_k(\beta) - b_k)^2$$

The Cross Validation function is defined as the sum of the squares of these differences between a predicted datum and the actual datum, for all data points:

$$CV(\beta) = \sum_{k=1}^{N} (a_k^T x_k(\beta) - b_k)^2 \tag{3.13}$$

The minimum of the CV function with respect to β represents the β for which the model would change the least if we drop one data point.

The CV as it is defined above is not very practical to compute because it involves solving N systems for different regularization parameters. A shortcut was found by Wahba [1977] who proved that:

$$CV(\beta) = \sum_{k=1}^{N} \frac{(b_k(\beta) - b_k)^2}{(1 - c_{kk})^2} \tag{3.14}$$

where $x(\beta) = (A^T A + \beta I)^{-1} A^T b$, $b_k(\beta) = a_k^T x(\beta)$ and c_{kk} is the kk element of $C = A(A^T A + \beta I)^{-1} A^T$.

The Cross Validation function has one unfavorable behaviour. If the matrix A and the data b are rotated by an orthogonal transformation to form a new rotated system, the minimum of the Cross-Validation function for this new system changes. Therefore Golub *et al.* [1979] suggested replacing the Cross-Validation by the Generalized Cross Validation which keeps its minimum under orthogonal transformations. The GCV is defined as:

$$GCV(\beta) = \frac{||b(\beta) - b||^2}{trace(I - C(\beta))^2}$$

or explicitly

$$GCV(\beta) = \frac{||(I - A(A^T A + \beta I)^{-1} A^T)b||^2}{[trace(I - A(A^T A + \beta I)^{-1} A^T)]^2} \tag{3.15}$$

The GCV function is again a function of β alone and various optimization methods can be used in order to minimize it. The major step in calculation of the GCV function involve the inversion and calculation of $(A^T A + \beta I)^{-1}$ and the estimation of the trace elements of the denominator. In a recent paper Golub and Von-Matt [1996] have developed a fast method to compute a close approximation to the GCV function for the over-determined case ($M < N$). We will use similar methodology when using hybrid methods (Chapter 5).

3.4 The L-Curve

Recently Hansen [1992 a] has developed the L-curve technique as a method to predict the regularization parameter. Although this technique does not have a formal proof, it is often used because of its simplicity. In this section we prove the connection between the L-curve and the probabilistic approach to inverse problems. According to this connection, a new choice of regularization parameter is proposed.

3.4.1 Description of the L-Curve

The L-curve is made by plotting the log of the misfit ($log(||Ax - b||^2)$) as a function of the log of the model norm ($log(||x||^2)$) which are obtained for different regularization parameters. As $\beta \rightarrow 0$ the model norm goes to infinity and the misfit is low, and as $\beta \rightarrow \infty$ the model norm goes to zero but the misfit goes to $||b||^2$. The plot has a typical L shape (Figure 3.2).

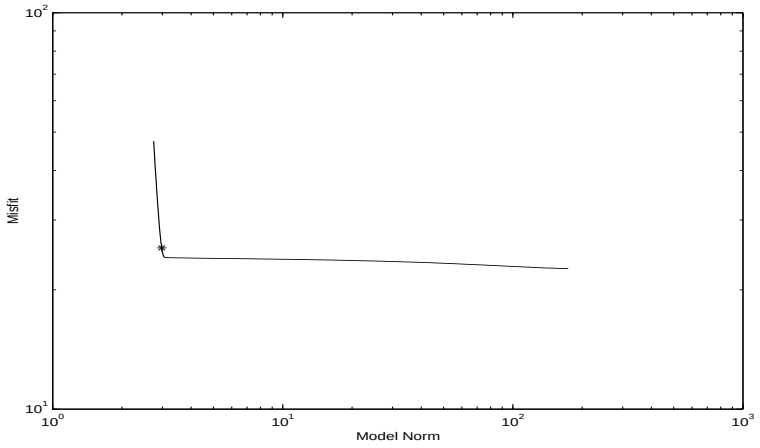

Figure 3.2: A typical L-curve of the one dimension gravity problem. The example is discussed in Chapter 6

Hansen [1992], [1995] claimed that the "best model norm" for a small misfit is obtained at the corner of the curve and therefore suggested to look at the maximum of the second

derivative of the curve. While his approach was heuristic in nature we prove that if the place which is chosen on the curve is different from the corner, the L-curve strategy agrees with the probabilistic approach.

3.4.2 The Probabilistic Approach

The probabilistic approach (Tarantola *et al.* [1982], Tarantola [1987]) looks at the probability density function of the noise ϵ

$$P(\epsilon = b - Ax) = C_b exp(-\frac{||b - Ax||^2}{2\sigma^2}) \tag{3.16}$$

where C_b is a normalization constant. In the probabilistic approach we also define the probability of the model x. It is often assumed that the model x is Gaussian with 0 mean and known constant standard deviation σ_x. The probability density function of the model is:

$$P(x) = C_x exp(-\frac{||x||^2}{2\sigma_x^2}) \tag{3.17}$$

The next stage is to look at the probability of the model given the data:

$$P(x|b) = C exp(-\frac{||x||^2}{2\sigma_x^2} - \frac{||b - Ax||^2}{2\sigma^2}) \tag{3.18}$$

where again C is a normalization constant.

The main idea in the probabilistic approach is to maximize the probability of the model given the data, which is equivalent to the minimization of:

$$minimize \quad \phi = ||b - Ax||^2 + \frac{\sigma^2}{\sigma_x^2}||x||^2 \tag{3.19}$$

We therefore see that if the distribution of the errors and the model parameters are Gaussian with **known** standard deviations σ and σ_x, then the regularization parameter β is simply:

$$\beta = \frac{\sigma^2}{\sigma_x^2} \tag{3.20}$$

This fact can be taken into consideration when searching the regularization parameter with unknown standard deviations. One could think of the following search method:

β Search Method

- 1. Choose regularization parameter β.
- 2. Minimize 3.19, get $x(\beta)$ and $\epsilon(\beta) = b - Ax(\beta)$
- 3. Calculate ${\sigma_{(est)}}^2(\epsilon(\beta)) = \epsilon(\beta)^T\epsilon(\beta)/N$ and ${\sigma_{(est)}}_x^2(x(\beta)) = x(\beta)^T x(\beta)/M$
- 4. If:

$$\beta_{(est)} = \frac{{\sigma_{(est)}}^2}{{\sigma_{(est)}}_x^2} \approx \beta$$

accept the regularization parameter.

A similar approach was suggested by Claerbout [1985]. However he did not carry out a line search, but tried to boot-strap in order to directly estimate the standard deviations. We now show that this approach is connected to the L-curve.

3.4.3 Connection Between the Probabilistic Approach and the L-Curve

In order to show the connection to the L-curve we look again at the minimization problem 3.1.

$$minimize \quad \phi = \phi_d + \beta\phi_m$$

Notice that if $x \sim N(0, \sigma_x)$ then:

$${\sigma_{(est)}}_x^2(x(\beta)) = x(\beta)^T x(\beta)/M = \phi_m/M \tag{3.21}$$

and if $\epsilon \sim N(0, \sigma)$ then

$${\sigma_{(est)}}^2(\epsilon(\beta)) = \epsilon(\beta)^T\epsilon(\beta)/N = \phi_d/N \tag{3.22}$$

If ϕ is minimized with respect to x then at the minimum:

$$d\phi = d\phi_d(x_{min}(\beta)) + \beta d\phi_m(x_{min}(\beta)) = 0$$

Therefore:

$$\frac{d\phi_d(x_{min})}{d\phi_m(x_{min})} = -\beta \tag{3.23}$$

If we combine the relation 3.20 to the probabilistic relations 3.23 and the estimated σ and σ_x from 3.22 and 3.21 we get:

$$\frac{d\phi_d}{d\phi_m} = -\frac{M\phi_d}{N\phi_m}$$

or

$$\frac{dlog(\phi_d)}{dlog(\phi_m)} = -\frac{M}{N} \tag{3.24}$$

more specifically:

$$\frac{d[log(||(I - A(A^TA + \beta I)^{-1}A^T)b||)]}{d[log||(A^TA + \beta I)^{-1}A^T)b||]} = -\frac{M}{N} \tag{3.25}$$

The L-curve is $log(\phi_d)$ as a function of $log(\phi_m)$. The point on the curve which is suggested here is the one at which the derivative satisfies 3.24. Again this equation is a nonlinear equation with one unknown parameter and can be solved by the same methods as the GCV or discrepancy principle.

Let us look at this point. If the number of model parameters is close to the number of data, then the slope is close to -1 and the result should be close to the knee of the curve. In this case this result will be close to Hansen's criteria (Hansen [1992 a]). However as the number of data grows, the slope should get closer to 0. This means that the point will lie to the right of the knee, and the regularization parameter should go to 0. In the other extreme, if the number of model parameters grows, the slope will approach infinity and the point would lie to the left of the knee, and the regularization parameter would approach ∞.

3.5 Practicalities and Limitations of Tichonov Regularization

3.5.1 Advantage of Invertible Objective Function

The operator W can be chosen arbitrarily. After it has been chosen we transform the problem into its standard form as discussed in Chapter 2. The only problematic case of W under-determined has to be handled with care. In most cases W is some form of differential operator and the null space can be calculated analytically. In general it is useful to avoid the case of W under-determined by adding a small number δ to its diagonal. The advantage of using W square is that we could use a canned solver such as the black box multigrid (Dendy [1982]) for the solution of $Wv = q$ and $W^T u = p$. If we add δ to the diagonal, the solution does not change significantly and the change can be quantified with some arithmetic. Let

$$x_1 = (A^T A + \beta W^T W)^{-1} A^T b$$

and

$$x_2 = (A^T A + \beta W^T W + \delta I)^{-1} A^T b$$

Write $B = (A^T A + \beta W^T W)$ and $s = A^T b$. The difference between the solutions is:

$$x_1 - x_2 = B^{-1}s - (B + \delta I)^{-1}s = B^{-1}(I - (I + \delta B^{-1})^{-1})s = \tag{3.26}$$

$$B^{-1}(I - I + \delta B^{-1} + \mathcal{O}(\delta^2)B^{-2})s = (\delta B^{-2} + \mathcal{O}(\delta^2)B^{-3})s$$

and therefore:

$$||x_1 - x_2|| \leq \delta ||B^{-1}x_1|| + \mathcal{O}(\delta^2)$$

Notice that the smallest singular value $\lambda_{min}(B) \geq \beta \lambda_{min}(W) = \tilde{\beta}$ and therefore:

$$||x_1 - x_2|| \leq \frac{\delta}{\tilde{\beta}} ||x_1|| + \mathcal{O}(\delta^2)$$

The difference between the models is $\mathcal{O}(\delta)$ and from a practical point of view it is simpler to work with a well-posed system. One more reason why this change is usually justified is that in most cases the choice of W is somewhat arbitrary. The reason is that W represents prior knowledge about the model which is usually not in a form of a hard constraint and in most cases is based on interpretation. Therefore changing the objective function in a slight way should not be of major concern when formulating the problem.

3.5.2 Solving the Equations

In order to practically solve the inverse problem one needs to decide on a method for the estimation of the regularization parameter (discrepancy principle, GCV or L-curve), find this regularization parameter and solve the system. All three methods require the calculation of a function of one parameter β, which involves the inversion of $(A^TA+\beta I)$, which requires $\mathcal{O}(M^3)$ operations. Usually the function has to be calculated several times and this will increase computational time even for relatively small scale problems. For this reason it might be better to decompose the matrix A, for example using the SVD of A which is $\mathcal{O}(12NM^2)$. Elden [1982] has proposed a cheaper variation. He suggested decomposing the matrix A into:

$$A = UBV^T \tag{3.27}$$

where $U^TU = UU^T = I$, $V^TV = I$ and B is bidiagonal. This decomposition is only $\mathcal{O}(\frac{4NM^2}{3})$. Using the decomposed system the problem is transformed into:

$$Bz = y \tag{3.28}$$

where $z = V^Tx$ and $y = U^Tb$. The regularized solution is then:

$$z = (B^TB + \beta I)^{-1}B^Ty \tag{3.29}$$

The calculation of $(B^TB + \beta I)^{-1}B^Ty$ requires only $\mathcal{O}(N)$ operations and therefore the search for a regularization parameter is computationally efficient. After the regularization

parameter is found, and z is calculated, the solution x is calculated by projecting z into the model space:

$$x = Vz \tag{3.30}$$

The two main bottle-necks in the Tichonov regularization are the computation of the inverse or the decomposition of A and the storage of the orthogonal matrices U and V. The solution of the system requires $\mathcal{O}(\frac{4NM^2}{3})$ operations which might be very large even for moderate M's. The storage requirement for V is $M \times N$ and for U is $N \times N$ and that could be prohibitive. In many applications such as the three dimensional gravity problem, the matrix A itself is too large to be stored and the only things which can be calculated are products of the form Av and $A^T u$. This type of application can not be handled with Tichonov-style regularization. We therefore turn to subspace methods.

Chapter 4

Subspace Methods

Most real world problems have two or three dimensions which lead to problems with a few thousand to a few million unknowns. Tichonov regularization solutions take a long time and considerable memory, and some shortcut is desired. The simplest shortcut is to discretize the problem with only a few parameters, which leads to an over-determined system. As stated in Chapter 2, this is not a good idea since coarse discretization can be viewed as forcing the solution to lie in a small subspace (chosen by the discretization) which might not fit the problem. Subspace methods can be viewed as a transformation of the problem into a small tractable subspace, and the major difference from naive under-parametrizing is that the subspace is an appropriate one for the problem.

In a more mathematical formulation we define a subspace S_k with $k \ll N < M$ and solve the minimization problem:

$$\begin{aligned} &minimize \quad ||Ax - b||^2 \\ &subject\ to \quad x \in S_k \end{aligned} \tag{4.1}$$

Problem 4.1 leads to a well-posed over-determined system for a good choice of S_k and a small enough k. In this thesis we review and explore four methods for choosing such a subspace. The most obvious subspace is that which is spanned by the singular vectors. Other methods try to obtain cheap approximations to the singular vectors. We review and explore the Krylov space, spaces from different grids and subspaces based on gradients. We then discuss the question of how many subspace vectors should be used, which is

equivalent to the choice of regularization parameter. We review the choice in the case of discrepancy principle and in the case of the L-curve, and develop the GCV principle for the selection of the subspace size.

4.1 The Truncated Singular Value Decomposition

Perhaps the most natural subspace to understand is the truncated SVD (TSVD) (Varah [1983], Van Huffel and Vandewalle [1991]). In TSVD the matrix A is decomposed into its singular-values and singular-vectors. The least-squares solution with zero misfit, can be written as:

$$x = VS^{-1}U^T b = \sum_{i=1}^{N} \frac{u_i^T b}{\lambda_i} v_i \tag{4.2}$$

As stated before, typically, the small singular values correspond to oscillatory singular vectors and cause an increase of the norm of the solution, but they do not have a big effect on the misfit. The simplest regularization is to throw away singular vectors v_i associated with small singular values. This is equivalent to defining a filter function of the form:

$$f_{TSVD}(\lambda) = \begin{cases} 0 & \lambda \leq \delta \\ 1 & \lambda > \delta \end{cases} \tag{4.3}$$

The solution x can be written as:

$$x_{TSVD} = \sum_{i=1}^{k} \frac{u_i b^T}{\lambda_i} v_i = \sum_{i=1}^{N} f_{TSVD}(\lambda) \frac{u_i b^T}{\lambda_i} v_i \tag{4.4}$$

where k is the index of the last singular value which is bigger than δ. The solution is spanned by $[v_1 ... v_k]$ which is a subspace of the whole active space of A, $[v_1 ... v_N]$. It is well known that while this solution is mathematically different from the Tichonov solution, it approximates it well.

A different way to look at this process is to define the regularization problem as a

minimization problem in the form:

$$minimize \quad ||Ax - b||^2 \tag{4.5}$$

$$subject\ to \quad x \in \mathcal{S}_k$$

where $\mathcal{S}_k$ is a subspace spanned by $V_k = [v_1 ... v_k]$.

The major advantage of TSVD is the separation of the problem to its eigen-modes, and distinguishing the important modes from the minor ones. This is a very basic principle in mathematics and physics. The major problem of the TSVD is that it costs $\mathcal{O}(12M^2N)$ operations, and therefore cannot be practically used on large scale problems. While it is expensive to calculate the whole SVD, the solution uses only k vectors and typically $k \ll N$. The main question is then: is it possible to calculate a cheap approximation to these k vectors? If such approximations are found, then a cheap solution to the system with similar properties to the TSVD can be constructed. In the next two sections we discuss such approximations.

4.2 Approximations From Krylov Space

Some of the most robust and successful algorithms to approximate the large singular vectors use the Krylov space (Lanczos [1961]). The Krylov space is the space:

$$\mathcal{K}(A, u, n) = span(A^T u, (A^T A)A^T u ... (A^T A)^{n-1} A^T u) \tag{4.6}$$

where $u \in R^N$.

In this section we review some of the Krylov space algorithms which are useful for the solution of ill-posed problems. We start with Lanczos bidiagonalization process which leads to the least-squares QR (LSQR). It has been shown by Paige and Saunders [1982] that LSQR is equivalent in infinite precision to the conjugate gradient least squares

(CGLS) method which we review next. We then analyze the regularization properties of Krylov space methods and discuss the situations in which Krylov space methods can fail.

4.2.1 Bidiagonalization and the LSQR Method

One way to obtain an approximation to the singular values and vectors of A is by Lanczos bidiagonalization. In this subsection we review the process as presented by Golub and Van Loan [1996] and explain its connection to the LSQR.

The goal of complete bidiagonalization is to generate the $N \times N$ matrix $U = [u_1 ... u_N]$, and the $M \times N$ matrix $V = [v_1 ... v_N]$ where $U^T U = I$ and $V^T V = I$, and an $N \times N$ bidiagonal matrix:

$$B = \begin{bmatrix} \alpha_1 & 0 & ... & 0 \\ \beta_1 & \alpha_2 & ... & 0 \\ .. & .. & ... & .. \\ .. & .. & ... & 0 \\ 0 & ... & \beta_{N-1} & \alpha_N \end{bmatrix}$$

such that:

$$U^T A V = B \tag{4.7}$$

or

$$A = UBV^T \leftrightarrow A^T U = VB^T \leftrightarrow AV = UB$$

Notice that U and V are not the singular vectors discussed in the previous section.

Let $\alpha_1 ... \alpha_N$ and $\beta_1 ... \beta_{N-1}$ be the diagonal elements of B, we can write:

$$Av_k = \alpha_k u_k + \beta_k u_{k+1} \quad k = 1...N-1$$

and

$$A^T u_k = \alpha_k v_k + \beta_{k-1} v_{k-1} \quad k = 2...N$$

Defining:

$$p_k = Av_k - \alpha_k u_k$$

$$r_k = A^T u_k - \beta_{k-1} v_{k-1}$$

$$\beta_0 v_0 = 0$$

we can conclude from the orthonormality of the vectors of U and V that:

$$\alpha_k = \pm||r_k||, \quad v_k = r_k/\alpha_k$$

$$\beta_k = \pm||p_k||, \quad u_{k+1} = p_k/\beta_k$$

These equations define the bidiagonalization process, given a matrix A and a starting vector $u_1 \in R^N$:

Lanczos Bidiagonalization

- Choose a vector $u_1 \in R^N$, set $v_0 = 0$.
- $\beta_1 = ||u_1||$, $u_1 = u_1/\beta_1$
- for i=1:k
  - $r = A^T u_i - \beta_{i-1} v_{i-1}$;
  - $\alpha_i = ||r||$; $\quad v_i = r/\alpha_i$;
  - $p = Av_i - \alpha_i u_i$;
  - $\beta_i = ||p||$; $\quad u_{i+1} = p/\beta_i$;

If $rank(A) = N$, then no zero α_k arise, however if $\alpha_k \rightarrow 0$, then the matrix is "almost" rank deficient. This character of the decomposition is used in the next chapter when discussing hybrid methods.

Now assume that only k steps of the bidiagonalization are carried out with a starting vector b. We have the $N\times(k+1)$ matrix $U_{k+1} = [u_1...u_{k+1}]$, where $u_1 = b$, the $M\times k$ matrix $V_k = [v_1...v_k]$ and the $(k+1)\times k$ bidiagonal matrix B_k . The matrices obey the relation:

$$AV_k = U_{k+1}B_k \tag{4.8}$$

It has been shown by Parlett [1980] and Golub and Van Loan [1996], that if the starting vector u contains the singular vectors which are associated with the large singular values, then the space spanned by the vectors in U and V contains good approximations to the space spanned by the k singular vectors of A. More over, the singular values of B_k, which are called the Ritz values, are good approximations to the singular values of A. Using this partial decomposition we can get a cheap approximate solution to the system.

Let x_k lie in the space which is spanned by V_k, then

$$x_k = V_k z$$

Substituting for x in 4.8 we can write:

$$Ax_k = AV_k z = U_{k+1}B_k z = b$$

Multiplying both sides by U_{k+1}^T and remembering that $u_1 = b$ is orthogonal to the rest of the u_i's we obtain: $U_{k+1}^T b = \beta_1 e_1$ with $e_1 = [1, 0...0]^T$. This gives the system:

$$B_k z = \beta_1 e_1 \tag{4.9}$$

B_k is only a $(k+1)\times k$ matrix and therefore the solution for z is straight forward. This type of solution requires storing the $M\times k$ matrix V_k, which for very large scale applications might be very large. Paige and Saunders [1982] found an iterative way to update the solution for every new vector v_k and a bidiagonal elements β_k and α_k which are found in each iteration. Their algorithm is called LSQR and it is one of the more stable

algorithms for the solutions of least-squares problems (Björk and Strakos [1995], Björk [1990], [1996]). Each step of the LSQR contains two parts. In the first part the vectors v_k, u_k are found through Lanczos bidiagonalization. In the second part an orthogonal transformation is used in order to update the solution x_k. The vectors u_k and v_k do not have to be stored in each iteration and therefore the storage requirements are minimal. For some uses the vectors might need to be stored (see Chapter 5) and then the algorithm is similar to the bidiagonalization algorithm.

In their paper, Paige and Saunders have shown that the LSQR is equivalent to the conjugate gradient least-squares (CGLS), which we analyze in the next section.

4.2.2 The Conjugate Gradient Least Squares Method

Conjugate gradient least-squares (CGLS) is an iterative method to solve linear equations. There are several ways to view this method. We take the approach of Golub and Van Loan [1996], and look at CGLS from a minimization point of view.

According to this approach at each iteration we try to minimize:

$$minimize \quad \phi_d = \frac{1}{2}||Ax - b||^2$$

Let us assume we are at iteration k, with a solution x_k and an A weighted residual:

$$s_{k+1} = A^T(b - Ax_k)$$

The main idea of CGLS is to update the model in the next iteration by a vector d_{k+1} multiplied by a scalar α_{k+1}:

$$x_{k+1} = x_k + \alpha_{k+1} d_{k+1}$$

The special property of this d_{k+1} is that it is A^TA conjugate to all previous d_j, $(j = 1...k)$, and therefore to x_k. By A^TA conjugate we mean that:

$$d_i^T A^T A d_j = 0 \qquad i \neq j$$

The special property of α_{k+1} is that it minimizes ϕ_d in the d_{k+1} direction:

$$minimize \quad \frac{1}{2}||A(x_k + \alpha_{k+1} d_{k+1}) - b||^2$$

It can be shown that the new direction d_{k+1} is a linear combination of the last direction d_k and the A weighted residual s_{k+1}. The algorithm can be written as follows:

CGLS Algorithm

- Initialize: $x = 0$, $d_1 = A^T b$, $r_0 = b$, $s_1 = d_1$
- for k=1,2...
  - $\alpha_k = ||s_k||^2 / ||Ad_k||^2$
  - $x = x + \alpha_k d_k$
  - $r_k = r_{k-1} - \alpha_k A d_k$
  - $s_{k+1} = A^T r_k$
  - $\beta_k = ||s_{k+1}||^2 / ||s_k||^2$
  - $d_{k+1} = s_{k+1} + \beta_k d_k$

The algorithm does not calculate the product $A^T A$ and the only things needed for the implementation are the calculation of the product of the matrix A with a vector and the product of the matrix A^T with a vector. After k iterations are performed the solution $x_k \in \mathcal{K}(A, b, k)$.

4.2.3 The Krylov Space Filter Function

The regularizing effects of Krylov space methods are well known in practice, but the properties of the Krylov filter are not well studied. In this subsection we find an expression

for the Krylov filter. First we notice that the Krylov solution is in the space $\mathcal{K}(A, b, k)$, which means that Krylov space solution is a solution to the problem:

$$minimize \quad ||Ax - b||^2 \tag{4.10}$$

subject to

$$x \in \mathcal{K}(A, b, k)$$

where k is the number of steps taken in the algorithm. The filter function can be derived from the definition of Krylov space. Since the Krylov solution is obtained from the Krylov space, the solution x_k can be written as:

$$x_k = z_1 A^T b + z_2 (A^T A) A^T b + ... + z_k (A^T A)^{(k-1)} A^T b \tag{4.11}$$

where the coefficients z minimize 4.10. Using the SVD of A and the identity:

$$(A^T A)^j = V S^{2j} V^T$$

the solution can be written as:

$$x_k = z_1 V S U^T b + z_2 V S^3 U^T b + ... + z_k V S^{2k-1} U^T b = \tag{4.12}$$

$$V(z_1 S + z_2 S^3 + ... + z_k S^{2k-1}) U^T b =$$

$$V(z_1 S^2 + z_2 S^4 + ... + z_k S^{2k}) S^{-1} U^T b = V D_k S^{-1} U^T b$$

The matrix D_k is a diagonal matrix and its diagonal elements are:

$$D_{ii} = \lambda_i^2 (z_1 + z_2 \lambda_i^2 + ... + z_k \lambda_i^{2k-2}) = \lambda_i^2 P_{k-1}(\lambda_i{}^2) \tag{4.13}$$

where P_{k-1} is a polynomial of degree $k-1$ in λ^2. Using this polynomial we can write the solution as:

$$x_k = \sum_{i=1}^{N} \lambda_i^2 P_{k-1}(\lambda_i{}^2) \frac{b^T u_i}{\lambda_i} v_i \tag{4.14}$$

This means that the Krylov filter is:

$$f_K(\lambda) = \lambda^2 P_{k-1}(\lambda^2) \tag{4.15}$$

The polynomial P_{k-1} is also known as the Ritz polynomial. The coefficients of P_{k-1} and their character can be found from the minimization problem 4.10. Substituting in 4.10 the solution x_k from 4.13 gives:

$$min||Ax - b||^2 = min||US(z_1S^2 + z_2S^4 + ... + z_kS^{2k})S^{-1}U^Tb - b||^2$$

The minimization does not change under orthogonal transformation and therefore:

$$= min||(z_1S^2 + z_2S^4 + ... + z_kS^{2k})U^Tb - U^Tb||^2$$

$$= min||(z_1S^2 + z_2S^4 + ... + z_kS^{2k} - I)U^Tb||^2$$

This minimization problem can be represented also as:

$$min|| \begin{bmatrix} b^Tu_1(z_1\lambda_1^2 + ... + z_k\lambda_1^{2k} - 1) \\ b^Tu_2(z_1\lambda_2^2 + ... + z_k\lambda_2^{2k} - 1) \\ ... \\ ... \\ b^Tu_N(z_1\lambda_N^2 + ... + z_k\lambda_N^{2k} - 1) \end{bmatrix} ||^2$$

which is equivalent to:

$$min||\Gamma(S^2\Sigma_k z - e)||^2 \tag{4.16}$$

where $\Gamma = diag(U^Tb)$, $e = [1, 1, ...1]^T$ and Σ_k is the $N \times k$ Van dermonde matrix:

$$\Sigma_k = \begin{bmatrix} 1 & \lambda_1^2 & ... & {\lambda_1}^{2(k-1)} \\ 1 & \lambda_2^2 & ... & {\lambda_2}^{2(k-1)} \\ .. & .. & ... & .. \\ .. & .. & ... & .. \\ .. & .. & ... & .. \\ 1 & \lambda_N^2 & ... & {\lambda_N}^{2(k-1)} \end{bmatrix}$$

Thus, the coefficients z are the solution of the least-squares problem:

$$\Gamma S^2 \Sigma_k z = \Gamma e \tag{4.17}$$

with the solution:

$$z = (\Gamma S^2 \Sigma_k)^\dagger \Gamma e \tag{4.18}$$

and the Krylov filter function is given by:

$$f_K(\lambda_i) = {\lambda_i}^2 P_{k-1}({\lambda_i}^2) = (S^2 \Sigma_k z)_i = (S^2 \Sigma_k (\Gamma S^2 \Sigma_k)^\dagger \Gamma e)_i \tag{4.19}$$

We now examine this filter more closely.

4.2.4 Properties of Krylov Space Filter Function

We are interested in the properties of the Krylov filter. We take a step back and look how this filter would be obtained under very different circumstances.

A very common problem is polynomial fitting (Saad [1987]). Assume we have the constant function $\psi(\lambda) = 1$ sampled at the points $\lambda_1^2 ... \lambda_N^2$ and we want to fit a polynomial in λ^2 of degree $k < N$ of the form:

$$z_1 \lambda^2 + ... + z_k \lambda^{2k}$$

to that function. This would lead to the following system:

$$S^2 \Sigma_k(\lambda^2) z = e$$

where z is a vector of size k which holds the coefficients of the polynomial and $e = [1, 1...1]^T$.

If all the equations in this least-squares problem have the same importance then we would use the generalized inverse of $S^2 \Sigma_k$, however if we want to fit some equations better

then others then we weight the equations by multiplying them by a diagonal matrix Γ and solve:

$$\Gamma S^2 \Sigma_k(\lambda^2) z = \Gamma e$$

At this stage Γ is a diagonal matrix which states how much weight we put on each equation. If we let $\Gamma = diag(U^T b)$, then the solution z of this interpolation problem is equivalent to 4.18. The problem is over-determined so the constant function $\psi(\lambda) = 1$ does not equal its polynomial approximation at the points λ_i^2. Instead at points λ_i^2 we have:

$$1 = \psi(\lambda_i) \approx (S^2 \Sigma_k (S^2 \Gamma \Sigma_k)^\dagger \Gamma e)_i$$

which is equivalent to the Krylov filter function 4.19. We therefore view the product $u_i^T b$ as the singular value weighting.

The Krylov filter depends on three things:

- The spectral content of the right hand side, b.
- The distribution of the singular values, λ.
- The iteration number, k.

Observing the spectral content of the right hand side, we can give an intuitive idea of how the Krylov filter works. Since the singular value weighting is $u_i^T b$, the approximation for the constant function 1 at points λ_i which are associated with small $u_i^T b$ is bad, but when a singular value has large weighting $u_i^T b$, it approximates the constant function 1 better. This means that the filter does not affect in a significant way vectors with large $u_i^T b$ but filters vectors which are associated with small $u_i^T b$. If the data is not noisy it contain large quantities of $u_i^T b$ for large singular values and small quantities of $u_i^T b$ for small singular values. Therefore the convergence is achieved first for the large vectors. When the data are noisy, one would still hope that the noise level is low enough so that

the amount of $u_i^T b$ for small i is still relatively larger than the amount of $u_i^T b$ for large i. However in cases that the data are noisy some of the noisy components could converge before the vectors which are associated with large singular values have converged. In such cases Krylov space methods might fail.

The dependence of the filter on the distribution of the singular values and on the iteration number is harder to explain. The Krylov filter cannot be calculated analytically for an arbitrary distribution of singular values and spectral content of the right hand side; however, we can take typical examples and look at the filter coefficients for these examples. We could then use these examples to identify which types of problems Krylov methods solve well and in which problems it might fail. In order to construct the examples we need to assume something about the spectral content of our right hand side. To make the analysis simple, we take a "worst case scenario" and assume that the spectral content is:

$$U^T b = [1, 1...1]^T = e$$

which means that we have as much noisy components as we have signal. We now observe the filter for different distributions of the singular values and different iteration numbers.

Effect of Clustered Singular Values

The first example is a very common one. We assume that the singular values $1...j$ are of the order of one and the rest are a few orders of magnitude lower. In order to analyse this case we build such an artificial distribution, then, calculate the corresponding Krylov filter for an artificial spectrum. The results are shown in Figure 4.1. The results demonstrate that this type of singular value distribution is very stable for the Krylov method. It is also possible to give some theoretical justification for the reasons that Krylov methods work so well when the singular values are clustered. Assume that the eigenvalues $1...j$

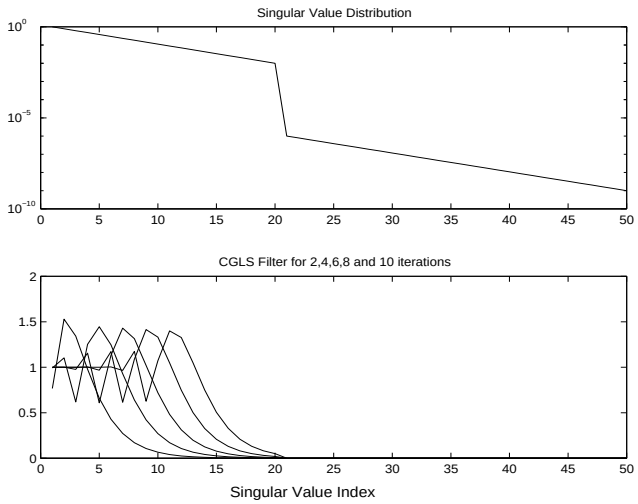

Figure 4.1: CGLS filter for step-like spectrum for 2,4,6,8 and 10 iterations

are of order of 1 and the rest are of order δ. We can then partition the matrix Σ_k into:

$$\Sigma_k = \begin{bmatrix} \Sigma_1 \\ \Sigma_2 \end{bmatrix}$$

where Σ_1 contains the singular values $1...j$ and Σ_2 contains the rest which are $\mathcal{O}(\delta)$, The Krylov filter then is the solution to the problem:

$$\begin{bmatrix} {\Lambda_1}^2 & 0 \\ 0 & \delta^2 {\Lambda_2}^2 \end{bmatrix} \begin{bmatrix} \Sigma_1 \\ \Sigma_2 \end{bmatrix} z = e$$

where Λ_1 contains the large singular-values and Λ_2 contains the small singular-values. This problem is solved by:

$$\begin{bmatrix} {\Lambda_1}^2 \Sigma_1 \\ \delta^2 {\Lambda_2}^2 \Sigma_2 \end{bmatrix}^T \begin{bmatrix} {\Lambda_1}^2 \Sigma_1 \\ \delta^2 {\Lambda_2}^2 \Sigma_2 \end{bmatrix} z = \begin{bmatrix} {\Lambda_1}^2 \Sigma_1 \\ \delta^2 {\Lambda_2}^2 \Sigma_2 \end{bmatrix}^T e$$

In order to calculate z we need to invert the square matrix on the left hand side which is:

$${\Sigma_1}^T {\Lambda_1}^4 \Sigma_1 + \delta^4 {\Sigma_2}^T {\Lambda_2}^4 \Sigma_2$$

Assuming that ${\Sigma_1}^T {\Lambda_1}^4 \Sigma_1$ is invertible:

$$({\Sigma_1}^T {\Lambda_1}^4 \Sigma_1 + \delta^4 {\Sigma_2}^T {\Lambda_2}^4 \Sigma_2)^{-1} = ({\Sigma_1}^T {\Lambda_1}^4 \Sigma_1)^{-1} (I + \delta^4 ({\Sigma_1}^T {\Lambda_1}^4 \Sigma_1)^{-1} {\Sigma_2}^T {\Lambda_2}^4 \Sigma_2)^{-1} \approx$$

$$(\Sigma_1{}^T \Lambda_1{}^4 \Sigma_1)^{-1}(I - \delta^4(\Sigma_1{}^T \Lambda_1{}^4 \Sigma_1)^{-1}\Sigma_2{}^T \Lambda_2{}^4 \Sigma_2) = (\Sigma_1{}^T \Lambda_1{}^4 \Sigma_1)^{-1} + \mathcal{O}(\delta^4)$$

The solution z can then be written as:

$$z = z(\Lambda_1) + \mathcal{O}(\delta^4)$$

where $z(\Lambda_1)$ is the solution which is achieved only with the large singular values. The solution in this case depends strongly on the large singular values and only to a very small extent on the small singular values.

Effect of Continuous Decay of the Singular Values

In some cases the singular values decay slowly. These cases are not as stable as the clustered one. In Figures 4.2 and 4.3 we observe the behavior for $\lambda = e^{-s}$ and for $\lambda = s^{-1}$. Notice that for the exponential decay the filter is still stable, however for $\lambda = s^{-1}$, the filter does not weed out the small singular values and therefore Krylov space methods might break down for this case especially if the data are very noisy.

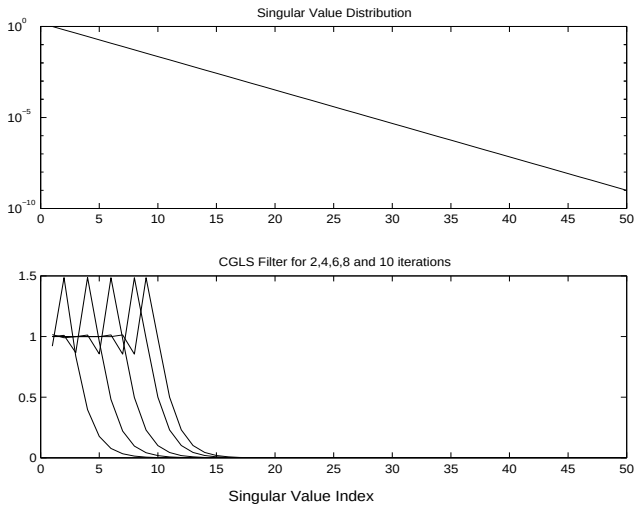

Figure 4.2: CGLS filter for slowly decaying spectrum for 2,4,6,8 and 10 iterations.

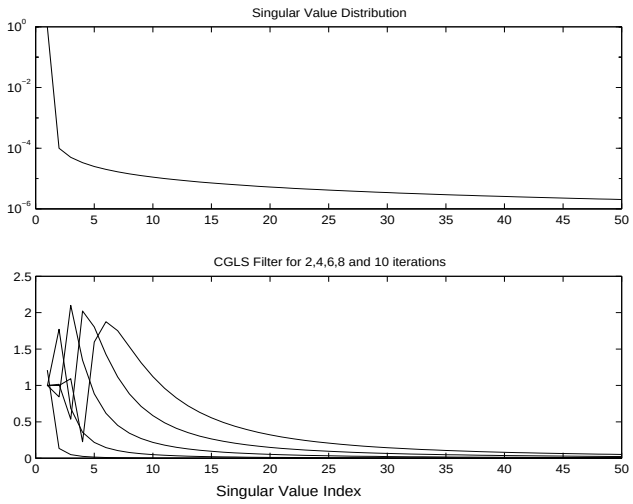

Figure 4.3: CGLS filter for very slowly decaying spectrum for 2,4,6,8 and 10 iterations.

4.2.5 Some other Properties of Krylov Space Solutions

One more aspect of the regularization is the fact that the norm of the solution increases as we solve the system to a lower misfit. This is evidently the case in Tichonov regularization and TSVD. It can be shown (Hestenes and Stiefel [1952], Hestenes [1980]), that this is also the case in Krylov space methods. This means that in every iteration we reduce the misfit at the expense of increasing the model norm.

Another important property of Krylov spaces is the loss of orthogonality. The filtering discussed earlier assumes that the Krylov vectors keep their orthogonality. In most cases the orthogonality breaks down and therefore the analysis is only approximate. In order to keep orthogonality the vectors could be re-orthogonalized at each iteration or partially orthogonalized as suggested in Golub and Van Loan [1996]. This can increase the cost of Krylov space algorithms. However in most straight-forward applications such orthogonalization is not needed, since the vectors are not stored for further use. In cases that the vectors are needed for further use we might use an orthogonalization procedure. We will return to this problem when using hybrid methods in Chapter 5.

We stated before that Tichonov regularization is achieved in $\mathcal{O}(4M^2N/3)$ operations. Krylov type regularization is much more efficient. In the worst case, when A is not sparse, and we have to perform almost N iterations in order to get a very low misfit, the number of operations is of the order of $\mathcal{O}(2MN^2)$. However if the matrix is sparse or the desired misfit is relatively high, then the number of operations is $\mathcal{O}(kMN)$, where k is the number of steps taken. In many applications $k \ll N$ and therefore the solution is much more efficient than Tichonov regularization.

Finally we noticed in the last section that in some situations Krylov space methods might fail. The main problem of Krylov space methods is the dependence of the solution on the right hand side, which might be very noisy. It is therefore desired to develop a method which does not use the right hand side and the solution does not depend on the noisy data. This is the goal of the next sections.

4.3 Multilevel Algorithms

Multilevel algorithms are widely studied in the context of differential equations (Briggs [1987], Hackbusch [1985]), however application of the same methods to ill-posed integral equations is not straight-forward. The main reason is the different behaviour of the spectrum of the equations. While integral equations have smooth vectors which correspond to the large singular values, differential equations possess smooth vectors which correspond to small eigenvalues. Our goal here is to develop a multilevel method for the approximation of the TSVD solution.

In order to obtain a cheap approximation for the solution, we turn to a solution from a subspace. We want our solution to be composed of the smooth singular vectors. We seek a method to approximate these smooth vectors. One of the possibilities is to introduce a coarse grid and to approximate these smooth vectors on it. In order to demonstrate, we look at the 1-D example in 1.2.1. Equation 1.6 is discretized with $129, 17$ and 5 grid points. The discretization is done by the midpoint rule. We then decompose the matrices into their singular values and singular vectors, and view the first and the fourth singular vectors of this system on the different grids (Figure 4.4).

The fourth singular vector is recovered reasonably well on 17 grid points but not so well on the 5 grid points. The first singular vector is approximated well even on the 5 point grid. This means that we could approximate the first vector reasonably well on a lower 5 point grid and save computations. We would like to use the vector which was calculated on the 5 point grid and transfer it on to the 129 point grid. This cannot be done by regular interpolation because if we naively interpolate from the coarse grid to the fine grid we would introduce elements from the null space. We therefore choose to interpolate in the following way. Let A_h be the discretized system on the fine grid and A_H be the system on the coarse grid. Any vector from the active set of A_H can be written

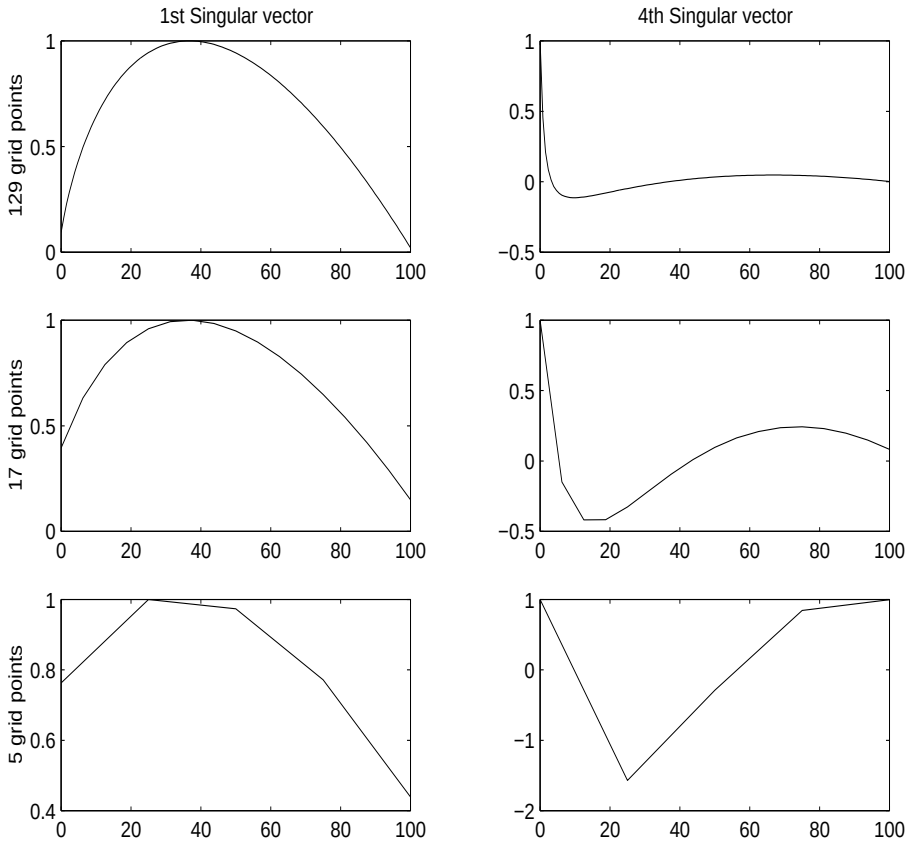

Figure 4.4: First and fourth singular vectors on 129, 17 and 5 grid points

as a linear combination of the kernels of the coarse grid:

$$v^H = \sum_{j=1}^{N} z_j^H a_j^H = A_H^T z^H \tag{4.20}$$

where H notes the coarse grid. The same applies for the fine grid:

$$v^h = \sum_{j=1}^{N} z_j^h a_j^h = A_h^T z^h \tag{4.21}$$

The N vectors z^H and z^h are the coefficients of the kernel functions on the coarse and fine grids. If the vectors v^H and v^h describe the same function on different grids then their coefficients z^H and z^h should be similar. We therefore interpolate from the coarse

grid to the fine grid by:

$$v_H^h = \sum_{j=1}^{N} z_j^H a_j^h = A_h^T z^H \tag{4.22}$$

In this way no elements from the null space of A_h are introduced to the approximation of v^h. This means that in order to interpolate a vector v^H to v_H^h, we need to find its decomposition into the kernel functions (4.20) and use these coefficients for the interpolation.

If the vector v is well represented on both grids then the approximation would be valid. However if v is not well represented on the coarse grid then the approximation is poor. Since we are not interested in very oscillatory solutions we could use a solution which is a subspace solution from the coarse grid. Each vector in this subspace would be a vector which is made from the fine grid kernels, but with coefficients from the coarse grid. We postpone the selection of the coarse grid, and explore two different methodologies to use multilevel algorithms, the approximated TSVD and the Landweber iteration.

4.3.1 The Multilevel TSVD

One way to obtain the coefficient vector z is using the SVD. Our goal is to approximate the singular vectors v_i^h on the coarse grid. We recall that the singular vectors v_i and u_i have the following relations:

$$\lambda_i^h v_i^h = A_h^T u_i^h$$

and

$$\lambda_i^H v_i^H = A_H^T u_i^H$$

This means that the coefficients of the kernels for generating the singular vectors v_i^h are just u_i^h. We therefore approximate the singular vectors as:

$$v_i^h \approx \frac{(A_h^T u_i^H)}{||(A_h^T u_i^H)||} = v_{i,H}^h \tag{4.23}$$

In order to demonstrate the efficiency of such an approximation we look at the norm of the difference between v_i^h and $v_{i,H}^h$, $||v_i^h - v_{i,H}^h||$, for $i = 1...17$ where h is the 129 point grid

and H is the 17 point grid. This difference is plotted in Figure 4.5. We see that while

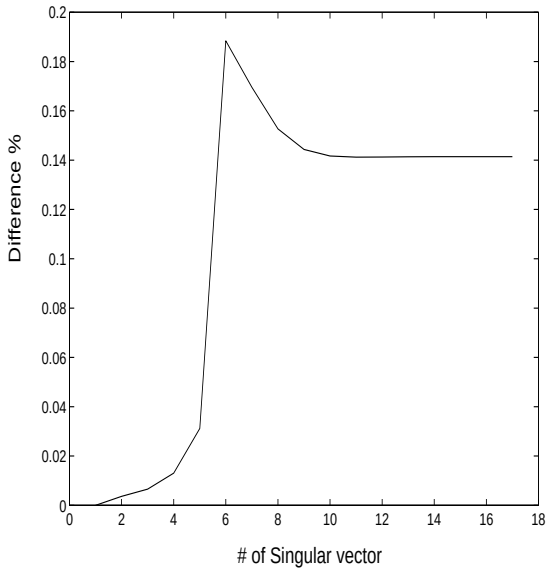

Figure 4.5: The norm of the difference between the approximated and the true singular vectors ($||v^h - v_H^h||$).

the difference is small in the first vectors, it increases as $i > 6$. In this case solutions which are built mainly from the first five vectors could be approximated well.

The solution which is achieved by the multilevel TSVD (M-TSVD) can be written as:

$$x_{MTSVD} = \sum_{i=1}^{k} z_i v_{i,H}^h \tag{4.24}$$

We now describe an algorithm to obtain the coefficients z_i. First we have to decide on the coarse level H and the fine level h. If we use some quadrature rule for the integration then the matrix A_H can be obtain by simply projecting A_h into H. In 1-D this is simple:

$$A_H(i,j) = A_h(i,lj)\frac{\Delta H}{\delta h} \tag{4.25}$$

where ΔH is the coarse grid discretization interval, δh is the fine grid discretization interval and l is the ratio of the number of points. The SVD of the coarse grid system

can be obtained at the low cost of $\mathcal{O}(12M_H^2N)$ where M_H is the number of grid points used in the coarse grid and usually (but not necessarily) $M_H < N$. This gives the decomposition:

$$A_H = U_H S_H V_H^T$$

The diagonal matrix S_H has at most M_H singular values which are different than zero. The singular vectors u_H which are associated with these singular values are then used to approximate the singular vectors v_i^h using equation 4.23. In most cases some of the singular values of the coarse grid are very small. As seen in the example, when a singular value is very small, we do not expect to have a good approximation to the singular vector. Therefore, we choose the singular value number $k < M_H$, λ_k^H, where its associated singular value is small enough, and work only with the singular vectors of the coarse system which are associated with singular values larger than λ_k^H. Let the matrix

$$Q = [v_{1,H}^h, ...v_{k,H}^h]$$

be the $M \times k$ matrix which spans the model subspace. The solution x can be written as:

$$x = Qz \tag{4.26}$$

and the inverse problem is transformed into:

$$min\ ||(A_hQ)z - b||^2 = min\ ||A_qz - b||^2 \tag{4.27}$$

where $A_q = A_hQ$ is an $N \times k$ matrix. While in principle the problem is over-determined, practically it could be rank deficient. In order to solve this problem we use again the SVD but this time apply it to the matrix A_q and write the solution z as:

$$z_{MTSVD} = \sum_{i=1}^{l} \frac{b^T u_i^q}{\lambda_i^q} v_i^q \tag{4.28}$$

where the subscript q relates to the singular values and vectors of A_q and with $l < k$. After z is found we calculate the solution x using 4.26. The algorithm can be summarized as follows:

Multilevel TSVD

- Discretize the system and calculate A_h and A_H
- Find the SVD of A_H
- Calculate the matrix $Q = [v^h_{1,H}, ...v^h_{k,H}]$
- Calculate $A_q = A_h Q$
- Solve $A_q z = b$ using TSVD
- set $x = Qz$

The cost of this algorithm is as follows: the SVD of the coarse grid system is obtained at $\mathcal{O}(12M_H^2 N)$. Calculating the matrix Q requires the multiplication of $A_h^T u_i^H \quad i = 1...k$, which is $\mathcal{O}(kNM)$, where M is the number of fine grid points. The calculation of A_q requires the multiplication of a matrix size $N \times M$ and a matrix of size $M \times k$ which requires $\mathcal{O}(kNM)$ operations. Finally we solve $A_q z = b$ using SVD, which requires $\mathcal{O}(12k^2 N)$. The total complexity of the algorithm is then:

$$C_{MG} = \mathcal{O}(12M_H^2 N) + \mathcal{O}(2kNM) + \mathcal{O}(12k^2 N) \tag{4.29}$$

If $M > N \ll M_H > k$, then the algorithm is efficient and the dominant term is order $\mathcal{O}(kNM)$. However if the required misfit is low and many vectors are needed, then the coarse grid M_H would not be so coarse and the dominant term in the cost of the problem is $\mathcal{O}(12M_H^2 N)$. In these cases the multilevel method suggested here is not going to be efficient.

4.3.2 The Multilevel Landweber Iteration

Another possible implementation of a multilevel algorithm is as a preconditioner to other methods. The most simple is the Landweber iteration (Landweber [1951]) which is:

$$x^{k+1} = x^k + A_h^T(b - A_h x^k) \tag{4.30}$$

This iteration may converge slowly; however, it can be accelerated by the Generalized Landweber iteration, which is given by:

$$x^{k+1} = x^k + \eta A_h^T D^{-1}(b - A_h x^k) \tag{4.31}$$

where D is a preconditioning matrix, and η ensures that the norm of the residual is decreasing. The optimal D is of course: $D = (A_h A_h^T)$ because the iteration then converges in one step. We now show that the multilevel iteration could be used as a Generalized Landweber iteration with $D = (A_H A_H^T)$.

As stated before any vector in the active space can be written as: $x = A_h^T z$. The Landweber iteration can be written as:

$$x^{k+1} = A_h^T z^{k+1} = x^k + \eta A_h^T D^{-1}(b - A_h x^k) = A_h^T z^k + \eta A_h^T D^{-1}(b - A_h A_h^T z^k)$$

This can be rewritten as a Richardson iteration for z

$$z^{k+1} = z^k + \eta D^{-1}(b - A_h A_h^T z^k) \tag{4.32}$$

This Landweber-Richardson iteration can be divided into two parts. The first is the preconditioning, i.e to apply D^{-1} and get $p_k = D^{-1}(b - A_h A_h^T z^k)$ and the second to find η such that the residual $r_{k+1} = ||b - A_h(x_k + \eta A^T p_k)||^2$ is minimized. We now look at the first step. We want to calculate the vector p_k:

$$p_k = D^{-1}(b - A_h A_h^T z^k) = (A_H A_H^T)^{-1} r_k$$

which is equivalent to the solution of:

$$A_H A_H^T p_k = r_k$$

This problem is equivalent to the inverse problem on the coarse grid:

$$A_H s_H = r_k$$

where $s_H = A_H^T p_k$. This means that the vector p_k contains the coefficients of the kernel functions which span the solution on the coarse grid. When these coefficients are found, they could be used on the fine grid. In order to use them we need to find the number η. We choose η such that the residual $r_{k+1} = r_k - \eta A_h A_h^T p_k$ is reduced as much as possible, i.e we look for η that will minimize $||r_{k+1}||^2 = ||r_k - \eta A_h A_h^T p_k||^2$. By differentiating with respect to η and setting to zero we get:

$$\eta_k = \frac{r_k^T A_h A_h^T p_k}{||A_h A_h^T p_k||^2} \tag{4.33}$$

Thus the algorithm for the multilevel Landweber iteration goes as follows:

Multilevel Landweber iteration

- $r_1 = b; \;\; x = 0$
- for $k = 1, 2...$
  - Solve $A_H A_H^T p_k = r_k$
  - Calculate η using 4.33
  - $x_{k+1} = x_k + \eta A_h^T p_k$
  - $r_{k+1} = r_k - A_h(A_h^T p_k)$

The matrix $A_H A_H^T$ could be rank deficient and therefore solving the problem $A_H A_H^T p_k = r_k$ might not be possible. This problem can be solved again by the TSVD. Let:

$$A_H = U_H S_H V_H^T$$

then a regularized p_k can be found by:

$$p_{TSVD} = \sum_{i=1}^{k} \frac{b^T u_i^H}{\lambda_i^H} v_i^H \tag{4.34}$$

The cost of the multilevel Landweber iteration is as follows. Decomposing A_H into its singular values and vectors is $\mathcal{O}(12M_H^2 N)$. The update of the solution takes $\mathcal{O}(NM)$, and the calculation of the residual is another $\mathcal{O}(NM)$. Thus the total cost of the multilevel iteration is:

$$C_{ML} = \mathcal{O}(12M_H^2 N) + \mathcal{O}(2kNM) \tag{4.35}$$

This cost is dominated by the $\mathcal{O}(2kNM)$ term if the coarse level is coarse enough.

4.3.3 Coarse Level Selection

The multilevel algorithms presented here used only two grids. The question is then, "how coarse should the coarse grid be"? Since it is not known *a-priori* how many vectors are going to be used we suggest to solve the problem iteratively starting with a very coarse grid and progressing to a finer grid if we cannot satisfy our stopping criteria. The algorithm is as follows:

Multilevel Algorithm

- Decide on a coarsest level H_1
- $x = 0$, $r = b$
- For $i = 1, 2, ..$
  - Use the M-TSVD or Landweber (Sections 4.3.1-2) to convergence on H_i to solve

$$A_h w_i = r_i$$

- Calculate the optimal step size obtained by this grid:

$$\theta = \frac{r_i^T A_h^T w_i}{||A_h^T w_i||^2}$$

- Update: $x_i = x_i + \theta w_i$
- Calculate the residual $r_{i+1} = b - A_h x_{H_i}^h$
- If the residual satisfies stopping criterion stop.
- Refine grid to H_{i+1}

This algorithm becomes expensive as the coarse grid becomes fine. The hope is that the stopping criterion is achieved long before this happens. In numerical experiments with the method (Chapter 6) we observed that when the noise level is about 10% or higher, the coarse grid is "coarse enough" and does not impose a real problem, however for relatively low noise levels, the multilevel algorithm becomes computationally intensive and not very useful.

4.4 Gradient Vectors

In this section we review the subspace method based on gradients which was developed by Parker *et al* [1987] for the analysis of magnetic data. The method is based on picking a space size k and generating k vectors which are made from gradients. The gradient of the j^{th} equation is:

$$g_j = a_j(a_j^T x - b_j) \tag{4.36}$$

Assuming $x = 0$, we generate k subspace vectors from a linear combination of the kernels:

$$v_j^n = \sum_{i=j_1}^{j_2} g_i = \sum_{i=j_1}^{j_2} b_i a_i \tag{4.37}$$

where j_1 and j_2 are some indices of the rows of A. The vectors are put in a matrix Q_k which spans the subspace:

$$Q_k = [v_1 ... v_k] \tag{4.38}$$

We now look for a solution x_k which is spanned by Q_k. The solution can then be written as:

$$x_k = Q_k z \tag{4.39}$$

Substituting $Q_k z$ instead of x in the equation leads to the least-squares problem:

$$minimize \; ||AQ_k z - b||^2 \tag{4.40}$$

Typically this problem can be ill-posed and the solution z is obtained through regularization. Let $A_q = AQ_k$ be an $N \times k$ matrix. There are a few ways to carry out the regularization process. The Tichonov regularization is one and we will look at the implementation of such regularization method in the next chapter. If we want to work with a subspace formulation then we could use the SVD. These two alternatives are also proposed by Parker [1994]. In this section we stick to the subspace formulation. The

matrix A_q is decomposed into its singular vectors:

$$A_q = U_q S_q V_q^T$$

and the solution is written as:

$$z_{GTSVD} = \sum_{i=1}^{l} \frac{b^T u_i^q}{\lambda_i^q} v_i^q \tag{4.41}$$

with $l < k$. Substituting back for $x_k = Q_k z$ we get:

$$x_k = Q_k V_q T_l S_q^{-1} U_q^T z \tag{4.42}$$

where T_l is a diagonal matrix which holds the TSVD filter function, i.e. one for the singular vectors which are larger than some δ and 0 for singular vectors which are smaller than δ. The algorithm can be summarized as follows:

Gradient Subspace Method

- Choose a space size k and a starting vector.
- Calculate k subspace vectors using gradients 4.37, $(j_2 - j_1 \approx N/k)$.
- Calculate the matrix $A_q = AQ_k$
- Calculate the SVD of A_q; $A_q = U_q S_q V_q^T$
- Solve $A_q z = b$ using TSVD
- set $x_k = Q_k z$

The complexity of the algorithm is as follows: Calculating the subspace vectors is only $\mathcal{O}(kM)$ operations. The calculation of the matrix A_q is $\mathcal{O}(kNM)$ operations. The calculation of the SVD of A_q is $\mathcal{O}(12k^2N)$ operations. The total cost of the algorithm is:

$$C_G = \mathcal{O}(kM) + \mathcal{O}(kNM) + \mathcal{O}(12k^2N) \tag{4.43}$$

This complexity is dominated by the $\mathcal{O}(kNM)$ term. The storage requirement of this algorithm is the storage of the matrix Q_k and the matrix A_q and its decomposition.

4.5 The Discrepancy Principle for Subspace Formulation

In the last sections we were concerned with the choice of the subspace. The question which is now asked is: how many subspace vectors should be used in order to obtain the solution? The number of vectors yields the number of subspace vectors needed in the TSVD, a stopping rule for the Krylov space methods, the coarsest level in the multilevel algorithms and the number of gradient vectors which should be used. If the number of vectors is too large, then the solution might fit the noise, and if the number of vectors is too small, then the solution might not fit the data. The problem is similar to the one we have when we solve for a regularization parameter β, however this problem is discrete and replaced with finding the integer k^* for the number of vectors. In parallel with Tichonov regularization, we review and develop methods to estimate the size of the subspace. We start with the discrepancy principle.

One simple stopping criterion is to stop when the misfit is lower than the tolerance level. Let T be the tolerance level. Let the misfit of ${\phi_d}^k > T$ be the misfit which is achieved when using k subspace vectors and let ${\phi_d}^{k+1} \leq T$ be the misfit using $k+1$ vectors. Then we could set $k^* = k+1$ as the subspace size.

This choice might not be satisfactory because, in some cases it could happen that the misfit at iteration k is $\phi_d^k \gg T$ and in the following iteration $k+1$ we could have $\phi_d^{k+1} \ll T$, however from statistical point of view, we are interested in a solution which $\phi_d \approx T$.

This problem can be easily overcome. By looking at any subspace algorithm we see that:

$$x_{k+1} = x_k + \alpha_k p_k$$

The result $\phi_d^{(k+1)} < T$ means that if take a step size α_k in the direction p_k the step is too large. A reasonable thing to do then is to go in the same direction, but not take a full

step α_k, but rather find α'_k such that:

$$||A(x_k + \alpha'_k p_k) - b||^2 = T \tag{4.44}$$

this gives α'_k as the solution of:

$$||Ap_k||^2\alpha^2 + 2(r_k^T Ap_k)\alpha + (||r_k||^2 - T) = 0 \tag{4.45}$$

This quadratic equation for α'_k gives two α'_ks that produce the same target misfit. The way to choose between them is simply to take the one which gives smaller solution norm ($||x||$). The algorithm can be summarized as follows:

Discrepancy Principle In Subspaces

- Choose a desired misfit level T
- For $k = 1, 2...$
  - Form the solution x_k and calculate the new subspace vector p_k and the solution x_{k+1}.
  - If $||Ax_{k+1} - b||^2 < T$, solve 4.45. where p_k is the last direction.
  - Given α_1 and α_2 which solves 4.45, choose α_1 such that:

$$||x_k + \alpha_1 p_k|| < ||x_k + \alpha_2 p_k||$$

- Set $x_{k+1} = x_k + \alpha_1 p_k$; return.

4.6 Generalized Cross Validation For Subspace Selection

The GCV principle is well developed to deal with Tichonov style regularization and it was reviewed in Section 3.3. Although Golub *et al.* [1979] suggested to use it for TSVD

subspace selection, it is not well developed for subspace type regularization. In this section we develop such a methodology.

Let $x_n \in S_n$ be the minimum of 4.1. Now assume we deleted the k^{th} data point, b_k from the system 4.1 and repeat the minimization without it. This can be written as:

$$\phi^k{}_s = ||Ax - b||^2 - (a_k^T x - b_k)^2 \qquad x \in S_n \tag{4.46}$$

where a_k^T is the k^{th} row of A. Let $x_n^{[k]}$ be the minimum of 4.46. We might ask the question: How well does the solution without the k^{th} data point reproduce that data point? The answer to this question is given by measuring the difference between the predicted data point without using it, and the actual k^{th} datum:

$$(a_k^T x_n^{[k]} - b_k)^2$$

If this difference is small, then the k^{th} data point is reproduced well even when it is not used. This is of course a good property since the solution does not depend on this data point in a strong way.

For a fixed subspace size n this estimate can be calculated for every k. By summing all these differences together we get the Cross Validation function:

$$CV(n) = \sum_{k=1}^{N} (a_k^T x_n^{[k]} - b_k)^2 \tag{4.47}$$

Notice that the CV depends only on n, and therefore for every n we have a measure of how well the data would be reproduced without using one datum. Since we want our solution to be independent of one datum as much as possible we want to choose n which minimizes the CV function.

The main problem with the formulation we have presented so far is that calculating the CV function looks practically impossible. In order to practically calculate the CV we have to find another expression for it. This is done next.

4.6.1 Calculating the Cross Validation Function

In this subsection we follow Wahba [1990] in order to derive an expression for the CV. We define $h_n(k,z)$ as the minimizer of:

$$\psi_n(k,z,x) = ||Ax-b||^2 - (a_k^T x - b_k)^2 + (a_k^T x - z)^2 \qquad x \in S_n \tag{4.48}$$

for a constant k and z. The function ψ is the same as the subspace function ϕ_s but with the data point b_k replaced with an arbitrary data point z. Let us find an expression for h_n. We define $b_k(z) = [b_1 ... b_{k-1}, z, b_{k+1} ... b_N]^T$. Then h_n is the minimizer of:

$$||Ax - b_k(z)||^2 \qquad x \in S_n$$

Let $S_n = span(V_n) = span([v_1 ... v_n])$. Then any vector $x \in S_n$ can be written as:

$$x = V_n q_n$$

where q_n is an n-length vector. Instead of finding h_n we find q_n which minimizes:

$$||AV_n q_n - b_k(z)||^2$$

By taking the derivative with respect to q_n and setting the result to zero, it is easy to show that:

$$h_n(k,z) = V_n q_n = V_n (V_n^T A^T A V_n)^{-1} V_n^T A^T b_k(z) \tag{4.49}$$

If we note the matrix $C_n = V_n (V_n^T A^T A V_n)^{-1} V_n^T A^T$ and $b_k(z) = b_k(0) + z e_k$ where $e_k = [0, ...1, ..0]^T$ then:

$$h_n(k,z) = C_n b_k(0) + z C_n e_k = C_n b_k(0) + z c_n^{[k]} \tag{4.50}$$

where $c_n^{[k]}$ is the k^{th} column in the matrix C_n.

We are now ready for "the leaving out one" lemma which was proved in Wahba [1990] for the Tichonov regularization case:

Let $x_n^{[k]}$ be the minimizer of 4.46. Then:

$$h_n(k, a_k^T x_n^{[k]}) = x_n^{[k]}$$

The "leaving out one" lemma means that if we replace the k^{th} data point by its predicted data point when it is not used, then the minimizer of the subspace objective function would be identical to the one without that point.

The proof is straight-forward. Let $v \in S_n$ be any vector different than $x_n^{[k]}$. Then:

$$\psi_n(k, a_k^T x_n^{[k]}, x_n^{[k]}) = ||Ax_n^{[k]} - b||^2 - (a_k^T x_n^{[k]} - b_k)^2 + (a_k^T x_n^{[k]} - a_k^T x_n^{[k]})^2 =$$

$$= ||Ax_n^{[k]} - b||^2 - (a_k^T x_n^{[k]} - b_k)^2 = \phi_s(x_n^{[k]})$$

Since $x_n^{[k]}$ is defined as the minimizer of ϕ_s then:

$$||Ax_n^{[k]} - b||^2 - (a_k^T x_n^{[k]} - b_k)^2 \leq ||Av - b||^2 - (a_k^T v - b_k)^2 \leq$$

$$||Av - b||^2 - (a_k^T v - b_k)^2 + (a_k^T v - a_k^T x_n^{[k]})^2$$

Thus $x_n^{[k]}$ is the minimizer of $\psi_n(k, a_k^T x_n^{[k]}, x_n^{[k]})$ and therefore it is equal to $h_n(k, a_k^T x_n^{[k]})$.

In order to get an expression for the CV we use the property above. First we look at the following identity:

$$b_k - a_k^T x_n^{[k]} = \frac{b_k - a_k^T x_n}{1 - c_{kk}} \tag{4.51}$$

where

$$c_{kk} = \frac{a_k^T x_n - a_k^T x_n^{[k]}}{b_k - a_k^T x_n^{[k]}} \tag{4.52}$$

Noting $\tilde{b}_k = a_k^T x_n^{[k]}$ and using the minimizer h_n we rewrite c_{kk} as:

$$c_{kk} = \frac{a_k^T h_n(k, b_k) - a_k^T h_n(k, \tilde{b}_k)}{b_k - \tilde{b}_k} \tag{4.53}$$

In order to get an analytical expression for c_{kk} we use the expression for h_n.

$$c_{kk} = \frac{a_k^T (C_n b_k(0) + b_k c_n^{[k]} - C_n b_k(0) - \tilde{b}_k c_n^{[k]})}{b_k - \tilde{b}_k} = \frac{a_k^T (b_k c_n^{[k]} - \tilde{b}_k c_n^{[k]})}{b_k - \tilde{b}_k} = \tag{4.54}$$

$$= \frac{(b_k - \tilde{b}_k)a_k^T c_n^{[k]}}{b_k - \tilde{b}_k} = a_k^T c_n^{[k]}$$

If we denote $AC_n = \tilde{C}(n)$ then we see that c_{kk} is merely the k,k element of this matrix and therefore:

$$b_k - a_k^T x_n^{[k]} = \frac{b_k - a_k^T x_n}{1 - \tilde{C}(n)_{kk}} \tag{4.55}$$

Using this identity, the CV function can be written as:

$$CV(n) = \sum_{k=1}^{N}(a_k^T x_n^{[k]} - b_k)^2 = \sum_{k=1}^{N}\frac{(b_k - a_k^T x_n)^2}{(1 - \tilde{C}(n)_{kk})^2} \tag{4.56}$$

We therefore have an expression for the CV as a function of the subspace n, the matrix A and the right hand side b. This expression is parallel to the one which was developed by Golub *et al.* [1979] for Tichonov regularization.

The CV function in the Tichonov regularization is replaced by the GCV function, which is a weighted version of the CV function. The reason is that the CV function does not keep its minimum under orthogonal transformation of A. In our case we do exactly the same and replace the CV by the GCV which is:

$$GCV(n) = \frac{||b - AV_n(V_n^T A^T A V_n)^{-1} V_n^T A^T b||^2}{trace(I - AV_n(V_n^T A^T A V_n)^{-1} V_n^T A^T)^2} \tag{4.57}$$

The GCV function for subspaces possesses similar characteristics to the regular GCV function. We use this function for the choice of the number of subspace vectors n.

The GCV expression can be further reduced. Assume that $AV_n = Q_n$ and assume that Q_n has full rank. The dimension of Q_n is $N \times n$. The GCV with is rewritten as:

$$GCV(n) = \frac{||b - Q_n(Q_n^T Q_n)^{-1} Q_n^T b||^2}{trace(I - Q_n(Q_n^T Q_n)^{-1} Q_n^T)^2}$$

The numerator is just the misfit. Let us take a closer look and the denominator. Let:

$$Q = U_Q S_Q V_Q^T$$

be the singular value decomposition of Q_n. We can then write:

$$Q_n(Q_n^TQ_n)^{-1}Q_n^T = U_Q S_Q V_Q^T (V_Q^T S_Q^T S_Q V_Q)^{-1} V_Q S_Q^T U_Q^T =$$

$$= U_Q S_Q (S_Q^T S_Q)^{-1} S_Q^T U_Q^T = U_Q U_Q^T$$

and the trace in the denominator can be written as:

$$trace(I - Q_n(Q_n^TQ_n)^{-1}Q_n^T) = trace(I - U_Q U_Q^T) =$$

$$= trace(I) - trace(U_Q U_Q^T) = N - trace(U_Q^T U_Q) = N - n$$

where the last equality is due to the identity:

$$trace(AB) = trace(BA)$$

Therefore we rewrite the GCV function as:

$$GCV(n) = \frac{||b - AV_n(V_n^TA^TAV_n)^{-1}V_n^TA^Tb||^2}{(N-n)^2} \tag{4.58}$$

4.7 The L-curve and Subspace Formulation

While the L-curve formulation is clear when we work with continuous quantities, it can also be used to estimate the size of the subspace. The basic idea is that if while expanding the subspace, we plot a discrete curve of the log of the misfit as the number of vectors of the subspace as a function of the log of the model objective function, we get a discrete L-curve. Since there is no real "corner" to a discrete curve, Hansen [1995] suggested to interpolate between the discrete points and then to find the corner of the interpolation. The size of the subspace is then set as the point which is closest to that corner. Possible problems could arise when trying to estimate the corner of the curve. In some cases the corner of the curve is not so sharp and its choice can be unambiguous.

Chapter 5

Hybrid Methods

Iterative subspace methods may well be the best way to solve a standard inverse problem. However as was demonstrated before, subspace methods could fail, the main reasons being the convergence of small singular values in the case of Krylov vectors, or the increase of the size of the subspace in multilevel and gradient methods. In these cases, we need a different approach. Hybrid methods can successfully deal with these problems. These methods use a subspace first to reduce the size of the problem while capturing the large singular vectors, and then use Tichonov style regularization inside that subspace. A general hybrid solution is the solution to the problem:

$$minimize \quad \phi = ||Ax - b||^2 + \beta||Wx||^2 \tag{5.1}$$

$$subject\ to \quad x \in S_k$$

In this section we review and develop such approaches. We first review the Krylov hybrid method and develop a new iterative approach to deal with the storage requirements and the loss of orthogonality. We then review the hybrid method based on subspaces which are spanned by gradients. This method was developed by Oldenburg *et al.* [1993], [1994]. Finally we review and develop criteria for parameter selection for hybrid methods.

5.1 Hybrid Krylov Methods

Hybrid Krylov methods can be viewed as a variation of the LSQR algorithm. Recall that the LSQR algorithm uses the Lanczos bidiagonalization to solve the system in the Krylov

subspace. The problem of the convergence of small singular values before the large ones is addressed by continuing the bidiagonalization process. In this case one hopes that larger singular values would converge if we continue the iteration enough. After k iterations, just like in the LSQR, we have a partial decomposition:

$$A \approx U_{k+1} B_k V_k^T$$

and we solve the problem (see Section 4.2.1):

$$B_k z = \beta_1 e_1 = \tilde{e}_1 \tag{5.2}$$

This equation looks identical to equation 4.9, however this is not the case. The main difference between the problem (5.2) and 4.9 in Chapter 4 is that the matrix B_k in this case is practically ill-conditioned because some of its singular values are numerically close to zero. In this case one cannot simply invert the matrix, and some regularization is needed. This gives extra degrees of freedom which we can use by choosing the right regularization method. O'Leary and Simmons [1981] suggested using the TSVD to solve 5.2, however Tichonov regularization could be used as well. The hybrid solution in this case, is a solution to the problem:

$$minimize \quad \phi = ||Ax - b||^2 + \beta ||Wx||^2 \tag{5.3}$$

$$subject\ to \quad x \in \mathcal{K}(A, b, k)$$

The solution is given by:

$$x(k, \beta) = V_k (B_k^T B_k + \beta I)^{-1} B_k^T \tilde{e}_1 \tag{5.4}$$

Notice that the hybrid solution depends on the iteration as well as the regularization parameter. For this type of hybrid method it is important to have the space size, k, capture some of the singular vectors which are associated with the small singular values.

We propose two ways for ensuring that this happens. The first option is through the LSQR algorithm. Recall (Section 4.2.1) that each iteration of the LSQR contains two steps. In the first step we calculate the Krylov vectors which bidiagonalize the matrix, and in the second step of each iteration we use these vectors to update the solution. The difference between the hybrid implementation of the LSQR and the regular implementation in Chapter 4 is that in the hybrid case while carrying each LSQR iteration we save the Krylov vectors V_k and the bidiagonal elements of B_k. The calculation of the solution for each iteration is just a simple rotation (second stage of each LSQR iteration) and therefore does not add to the cost. In a regular implementation of LSQR we would have stopped after some convergence criteria such as GCV, L-curve or discrepancy principle had been achieved. In this case we would achieve that criterion and continue iterating further. Only after we iterated enough (to capture some of the zero singular values) we would stop. Since we stored the Krylov vectors in the LSQR process, we now have the matrices V_k, U_{k+1} and B_k and we can choose the regularization parameter β which solves 5.4. This algorithm goes as follows:

Hybrid LSQR

- Begin LSQR process
  - Carry out LSQR iteration (Chapter 4 section 4.2.1)
  - Save the Lanczos vectors V_k and the bidiagonal elements of B_k
  - Check convergence
  - If converged continue iterating to iteration $n = (1 + \gamma)k$
- Use the Lanczos vectors V_n and B_n to solve for x using equation 5.4

In most cases we found that the choice $\gamma \approx 0.2$ is satisfactory, which means that most of the vectors which are associated with the large singular values have converged.

Another approach to the same problem is to use the Lanczos bidiagonalization process directly (as was described in Chapter 4 section 4.2.1). In each step of the bidiagonalization process we obtain a bidiagonal matrix B_k which contains an approximation to the singular values of the system. To ensure that we capture some of the small singular values we carry out the SVD of the small sparse matrix B_k. In this case we continue the decomposition until we have a matrix B_n with n_1 singular values which are greater than a small number δ and $n - n_1$ singular values which are smaller than δ. After the decomposition has been obtained it is used in equation 5.4 in order to calculate the solution. In most cases we found that if the number of almost zero singular values is roughly $10 - 20\%$ from the overall singular values of B_n the results are satisfactory.

The main problems of Krylov spaces as presented above is first the storage requirement. We need to store a matrix which holds the Lanczos vectors. This is a full matrix of size $k \times M$. In some sparse problems like tomography, the storage of these vectors can be larger than the storage of the whole system! In very large scale problems, although the system is full, it might not be stored and only the multiplication process of a matrix times a vector is stored. In this case the storage of the Lanczos vectors can become a problem. The second problem is the loss of orthogonality. It is very well known that Krylov methods in finite precession lose orthogonality. After a relatively low number of iterations the vectors in V_k are not orthogonal any longer and as a result, we cannot use 5.4 in order to solve the system. The solution to this problem is to re-orthogonalize the Lanczos vectors which can be done fully or partially (Golub and Van Loan [1996]). In this thesis we implement the re-orthogonalization with a modified Gram-Schmidt process, however one could use Givens rotations for the same purpose as was suggested in Kelly [1995]. To fully re-orthogonalize the Lanczos vectors at iteration k costs $\mathcal{O}((k-1)M)$ operations, which for large k might be very computationally burdensome. In practice most problems we handle in this thesis needed to be re-orthogonalized for $k > 15$. It is

therefore desirable to develop a different methodology.

5.2 Iterated Hybrid Krylov Method

Optimally we would like to use Lanczos bidiagonalization as presented in the last section to solve any inverse problem, however we might have two main problems. First, we need to store k vectors of length M. This may be a problem when the system is very large. The second difficulty is that the Lanczos vectors lose orthogonality, and the bidiagonalization process has to be carried out with a re-orthogonalization procedure. This procedure becomes computationally intensive as the problem starts to grow. To calculate that effect, assume we calculate n orthogonal vectors. Since in each iteration we orthogonalize the new vectors with respect to all previous vectors, the number of vectors to be re-orthogonalized would be one in the second iteration, two in the third iteration up to $n-1$ in the n^{th} iteration. The total number of vectors which needs to be re-orthogonalized is then:

$$1 + 2 + ... + (n-1) = \frac{1}{2}n(n-1)$$

This process makes the decomposition slow. In order to avoid these difficulties, we develop the iterated Krylov method. The idea is a combination of two techniques, the gradient subspace method developed by Oldenburg *et al.* [1993], [1994] and the Generalized Minimum Residual method with restarts (GMRES) (Saad [1996]).

The GMRES is a method for the solution of square non-symmetric well-posed problems. At the k^{th} iteration of the GMRES the vector p_k solves the least-squares problem:

$$minimize||Ap_k - r_k|| \qquad p_k \in \mathcal{K}(A, b - Ax_k, l) \tag{5.5}$$

where $r_k = b - Ax_k$. In the GMRES method the number of vectors l which is used is relatively small especially when the system is large. In this section we adopt this idea

from the GMRES method. However we do not adopt the GMRES procedure exactly. Since the problem we solve is ill-posed in nature, we solve the least-squares problem 5.5 using Tichonov regularization, similar to the subspace method which was used by Oldenburg *et al.* [1993], [1994].

The major idea of the iterated Krylov method then is to use only a small number, l, of orthogonal Lanczos vectors to obtain an update to the solution x and a new residual r. This means that we need to store only an $l \times M$ size matrix and we need only $\frac{1}{2}l(l-1)$ orthogonalization steps in each iteration of this process. The solution using only l vectors is probably not satisfactory, i.e. the solution does not satisfy some stopping criterion such as the GCV, and therefore we repeat the process. This process is similar to the restarts of the GMRES. In our case it means that we repeat the Lanczos bidiagonalization process but this time with the right hand side vector r. The iteration is done until convergence (to some kind of criterion) is achieved. If we carry out k iterations of this procedure then the total number of re-orthogonalization steps is

$$\frac{1}{2}kl(l-1)$$

which is hopefully smaller than $\frac{1}{2}n(n-1)$. The number of vectors l is chosen according to the size of the problem and the size of the available memory. Another option is to avoid the re-orthogonalization process totally and to iterate only a small number of iterations l in the bidiagonalization process, hoping that the orthogonality is not violated seriously. In this case the solution of $B_l z = r$ does not necessarily yield an optimal vector $p = V_l z$ which minimizes $||A(x+p) - b||^2$ and therefore the solution is updated similarily to the CGLS, by solving for a real number α which minimizes $||A(x+\alpha p) - b||^2$. The solution for α is simply:

$$\alpha = \frac{r^T Ap}{||Ap||^2}$$

In this thesis we use $l \approx 5-10$ for problems with $4000-30000$ unknowns. The algorithm

is as follows:

Iterated Hybrid Krylov Method

- Choose a number of vectors per iteration l
- $r = b$, $x = 0$
- For $k = 1, 2...$
  - Carry out l steps of the bidiagonalization process with starting vector r:

$$A \approx U_{l+1} B_l V_l^T$$

  - Solve $B_l z_k = \tilde{e}_1$ using hybrid methods:

$$z_i = (B_l^T B_l + \beta_k I)^{-1} B_l^T \tilde{e}_1$$

    where the regularization parameter β_k is chosen according to the stopping criterion method.

  - $p_k = V_l z_k$, $s_k = A p_k$
  - $\alpha = (r^T s_k)/||s_k||^2$
  - $x = x + \alpha p_k$; $r = r - \alpha s_k$

One of the most important steps in this algorithm is the solution of the system $B_l z_i = \tilde{e}_1$. Since we regularize the solution, the choice of regularization yields the type of solution achieved. Typically at starting iterations no regularization is needed (i.e. $\beta = 0$ is satisfactory), since the subspace V_l contains mainly vectors which are associated with large singular values. However as the iterations proceed the right hand side r contains more noise and the Lanczos vectors which are associated with it could consist of some vectors which are associated with the small singular values. In this case, regularization is

needed and the regularization parameter β grows. Finally as the right hand side contains only noise, the regularization parameter β approaches infinity which means that the perturbation p goes to 0 and the process is terminated. This solution converges more slowly than CGLS or LSQR processes however, it tends to converge to smoother types of solutions.

The total cost of the algorithm is as follows. The bidiagonalization process is $\mathcal{O}(lMN)$ and we carry out k such processes. The solution of the reduced linear system of the process is only $\mathcal{O}(l)$ and the calculation of the update and the residual is $\mathcal{O}(2MN)$. Therefore the total cost of this procedure is

$$C_{IK} = \mathcal{O}(k(l+2)MN)$$

where k is the number of iterations. Usually the number $k(l+2)$ is larger than the number of vectors needed for Lanczos bidiagonalization with no restarts, therefore this method is usually more expensive than the implementation of hybrid Lanczos bidiagonalization which was discussed in section 5.1. However since the method takes less storage space and still allows convergence of the vectors which are associated with the large singular values, in some problems where storage is a problem, this method may be the only way to obtain a reasonable solution.

5.3 Hybrid Methods Based On Gradients

In this section we review the subspace method which is based on gradients. This method was developed by Oldenburg *et al* [1993], [1994]. Similar method was suggested by Kennet and Williamson [1988]. The method is based on an iterative regularized solution of the least-squares problem, just as in the last section. The difference between this method and the previous one is that the vectors here are taken from gradients and not from a

Krylov space. The method can be viewed as an iterated version of the gradient subspace method which was discussed in Chapter 4.

Just like in the last section, this method is based on picking a space size l and generating l vectors which are made from gradients. The gradient for the j^{th} equation is:

$$g_j = a_j(a_j^T x - b_j) \tag{5.6}$$

Now assume we are in iteration k with a solution vector x_k and we want to calculate an update p_k to the solution vector. The gradient with respect to p_k can now be written as:

$$g_j^k = \frac{1}{2}(\frac{\partial}{\partial p_k}(a_j^T(x_k + p_k) - b_j)^2)_{p_k=0} = a_j(a_j^T x_k - b_j) = -r_j^k a_j \tag{5.7}$$

We generate $l-1$ subspace vectors from a linear combination of these gradients:

$$v_j^k = \sum_{i=j_1}^{j_2} g_i^k = \sum_{i=j_1}^{j_2} - r_i^k a_i \tag{5.8}$$

where j_1 and j_2 are some indexes of the rows of A. Note that if we generate only one such vector then this vector is simply the steepest descent vector. We keep one last vector v_l^k as a vector which is a descent direction to the function $||x+p||^2$ and therefore $v_l^k = x_k$. The vectors can be put in a matrix Q_l which spans the space:

$$Q_l = [v_1^k ... v_l^k] \tag{5.9}$$

The subspace vectors can be linearly dependent and in this case the matrix Q_l can be ill-posed. This could lead to later complications and therefore the vectors in Q_l are re-orthogonalized using the SVD. We now look for a direction p_l which is spanned by Q_l and decreases the misfit by some amount α, $0 < \alpha < 1$. Write

$$p_k = Q_l z \tag{5.10}$$

Substituting the subspace decomposition into the equation gives the least square problem:

$$minimize\ ||r_{k+1}||^2 = ||AQ_l z - r_k||^2 \tag{5.11}$$

$$subject\ to\ \ ||r_{k+1}||^2 \leq \alpha ||r_k||^2$$

for the coefficients z. This problem can be also written as:

$$AQ_l z = r_k$$

where the solution z is regularized such that $||r_{k+1}||^2 \leq \alpha ||r_k||^2$. In order to solve this problem let $A_q = AQ_l$ be an $N \times l$ matrix. The system can still be ill-posed and therefore regularization is needed. Again the choice of regularization yields the type of solution. Since the final goal is to decrease $||x||$ Oldenburg *et al.* suggested regularizing the system with respect to $||x + p|| = ||x + Q_l z||$. This leads to the minimization problem:

$$minimize \quad \beta ||x_k + Q_l z||^2 + ||A_q z - r_k||^2 \tag{5.12}$$

with the solution:

$$z = (A_q^T A_q + \beta I)^{-1} (A_q^T r - \beta Q_l^T x_k) \tag{5.13}$$

The regularization parameter β is adjusted such that the misfit would be reduced by a constant factor α.

After the solution z is found we calculate $p_k = Q_l z$ and update the model:

$$x_{k+1} = x_k + p_k$$

The algorithm can be summarized as follows:

Gradient Subspace Method

- Choose a subspace of size k and a starting vector x_0, $r_1 = b - Ax_0$
- For $k = 1, 2...$
  - Calculate subspace vectors using 5.8 and re-orthogonalize them.
  - Calculate the matrix $A_q = AQ_l$

- Solve: $z = (A_q^T A_q + \beta I)^{-1}(A_q^T r_k - \beta Q_l^T x_k)$
- set $p = Q_l z$
- Update the model $x_{k+1} = x_k + p$ and residual $r_{k+1} = b - Ax_{k+1}$

The number of subspace vectors l is chosen according to the size of the problem and the available memory. Since the subspace Q_l is made of gradients and is not guaranteed to be smooth it is advantageous to use a large subspace. Oldenburg *et al.* used $l \approx 10$ to 100 vectors for the solution.

The complexity of the algorithm is as follows: Calculating a subspace vector is only $\mathcal{O}(M)$ operations. The calculation of the matrix A_q is $\mathcal{O}(lNM)$ operations. The calculation of the solution of the system 5.13 is $\mathcal{O}(l^2 N)$ operations. If this process is repeated for k iterations and the total cost of the algorithm is:

$$C_G = \mathcal{O}(lM) + \mathcal{O}(lkNM) + \mathcal{O}(l^2 N) \tag{5.14}$$

This complexity is dominated by the $\mathcal{O}(lkNM)$ term.

5.4 Parameter Selection and Space Size

In previous chapters we had to estimate the regularization parameter (for Tichonov regularization) or the space size (for subspace methods). When using hybrid type methods we have to estimate both space size and regularization parameter. In this section we discuss some of the heuristics which we develop for this choice.

We start with the choice of the subspace size. In order to understand our motivation we recall the reasons which motivated us to work with hybrid methods. In general, hybrid methods require more storage than subspace methods and they converge slower. The reasons for the use of such methods is to let the singular vectors which are associated with large singular values converge, and to regularize the convergence of the singular vectors

which are associated with small singular values. Keeping this in mind, we differentiate between two types of subspace methods. The first type is the hybrid Krylov method. In this method we are not concerned with storage and therefore we can increase the size of the subspace as we wish. On the other hand, with iterated Krylov methods and gradient subspace methods, we are limited by storage. We therefore describe the heuristics for each of the criteria we have and adjust them to the method used.

5.4.1 Discrepancy Principle

Recall that the discrepancy principle is based on a target misfit. Our goal is to find a regularized model x such that $||b - Ax||^2 = N\sigma^2$. When storage is not a problem, the process is divided into two steps. In the first step we estimate the space size. Since we want to capture most of the large singular values and use Tichonov regularization on the reduced system, we make sure we capture some of the small singular values. This can be done in two ways as explained in Section 5.1. First through LSQR we can check the misfit at each iteration, and only after the misfit is reduced further than the target misfit do we stop. Alternatively we can use the bidiagonalization process and calculate, in each step, the SVD of the small bidiagonal matrix B_k. Our experience shows that when $10 - 20\%$ of the singular values of B_k are small enough (smaller than $\delta = 10^{-6}$, assuming $\lambda_1 = 1$), then most of the large singular values have converged. We then use the bidiagonal system B_k to solve the nonlinear equation for β:

$$||\tilde{e}_1 - B_k z(\beta)||^2 = ||(I - B_k(B_k^T B_k + \beta I)^{-1} B_k^T)\tilde{e}_1||^2 = N\sigma^2 \tag{5.15}$$

This nonlinear equation can be solved quickly since B_k is of small size and bidiagonal.

When the space is limited we carry out the process above iteratively. In each step we reduce the misfit in half until we get to the target misfit. In the n^{th} iteration we solve:

$$||\tilde{e}_1 - B_{k,n} z_n(\beta)||^2 = ||(I - B_{k,n}(B_{k,n}^T B_{k,n} + \beta_n I)^{-1} B_{k,n}^T)\tilde{e}_1||^2 = \tag{5.16}$$

$$\frac{1}{2}||\tilde{e}_1 - B_{k,n-1} z_{n-1}(\beta_{n-1})||^2$$

The same process is done with the gradient subspace method. However when using this method we cannot benefit from a bidiagonal system and we have to solve the $N \times k$ problem 5.13 for different β's in order to solve the nonlinear equation:

$$||A_q(A_q^T A_q + \beta I)^{-1}(A_q^T r - \beta Q_l^T x_k) - b||^2 = \frac{1}{2}||Ax_k - b||^2 \tag{5.17}$$

Although the solution of this problem is not as fast as for the bidiagonal system, it is relatively fast since $A_q^T A_q$ is only $l \times l$ in size.

5.4.2 Implementation of the GCV

The application of the GCV criterion in the case of the Krylov method is straight-forward. After the space is calculated with the bidiagonalization process, and the almost singular matrix B_k is calculated, we use the GCV principle and find the minimum of the GCV function in the subspace. This is done by minimizing:

$$GCV(k,\beta) = \frac{||(I - B_k(B_k^T B_k + \beta I)^{-1} B_k^T)\tilde{e}_1||^2}{[trace(I - B_k(B_k^T B_k + \beta I)^{-1} B_k^T)]^2} \tag{5.18}$$

The evaluation of the GCV function in this case is very cheap and the minimization can be carried out easily.

When storage is a problem we can still use 5.18 in the small subspace of k vectors. The major problem in this case is that if the subspace is too small, no regularization is needed and $\beta \to 0$. This kind of behaviour is observed in the first few iterations. However as the process proceeds, the vectors which are associated with large singular values have converged and even small problems need to be regularized. The regularization parameter then is increasing, until the residual contains mainly noise and the regularization parameter $\beta \to \infty$. Practically the values 0 and ∞ are determined versus the singular values

of B_k. If the value of $\beta \ll min(\lambda(B_k))$ then we consider the regularization parameter as 0 and if $\beta \gg max(\lambda(B_k))$ we consider the regularization parameter as infinity. Therefore the iterative hybrid process is terminated when $\beta \gg max(\lambda(B_k))$.

When working with gradients, the implementation of GCV is exactly the same, however in this case we have to minimize:

$$GCV(l,\beta) = \frac{||(I - A_q(A_q^T A_q + \beta I)^{-1} A_q^T)b||^2}{[trace(I - A_q(A_q^T A_q + \beta I)^{-1} A_q^T)]^2} \tag{5.19}$$

Minimizing this equation is a bit more expensive because we cannot benefit from the bidiagonal structure, however since the space size l is small we only need to invert an $l \times l$ matrix in each evaluation of the GCV function.

Implementation of the L-Curve Technique

Implementation of the L-curve technique is very similar to the implementation of the GCV when working with the bidiagonalization process. After the bidiagonalization is performed we use it to calculate the corner of the L-curve of the small bidiagonal system $B_k z = \tilde{e}_1$. The implementation of the L-curve in the case of limited space is different. If the space size is small, then the shape of the graph $log(\phi_d)$ as a function of $log(\phi_m)$ does not necessarily have the L-shape. In this case we might get into trouble if we simply try to use the L-curve criterion. In general I would try to avoid using the L-curve for this type of regularization. One possible implementation is to calculate a regularization parameter, exactly like in the discrepancy principle, and reduce the misfit by a certain amount. At some stage the misfit is not reduced but the model norm starts to increase. This is somewhat parallel to process in CGLS and we therefore suggest to stop at this point.

Chapter 6

Applications

The methods and heuristics developed in previous chapters were motivated through working on specific applications. The goal of this thesis is to develop methods that are generic in nature. The problems presented here are often solved in geophysics and medical physics. In some of these applications such as 3-D gravity and medical tomography, the size of the problem can prevent an attempt to invert the real problem and to deal with the non-uniqueness. In many of these cases the problem is simply under-parametrized, and the inverted image might be distorted. Another goal of this thesis is to compare the variety of methods, test their strengths and weaknesses such that given a specific problem, we can find computationally feasible method for carrying out the inversion.

This chapter is built as follows: We use the test problems to test different aspects of the inversion process. We start with gravity and use the 1-D problem to compare methods to estimate noise levels. The goal here is to test which of the methods is more generic in nature. The small size of the problem enables us to work with all methods and compare solutions. The 2-D gravity problem is then used to compare the number of floating points operations (flops) and the efficiency of the different methods. It is very common to use linear solvers as imaging operators for a crude approximation of a nonlinear problem, we therefore test our methods on the linearized nonlinear gravity problem. In the final test of the gravity problems, we use the most efficient method on a large 3-D gravity problem and apply our methodology to field data. We then do the same process for the tomography inverse problem and consider borehole tomography

and SPECT imaging. We apply the methods on field data sets in order to demonstrate robustness.

6.1 The Gravity Problem

The gravity inverse problem is a generic type of problem. As with many other geophysical problems the kernels are concentrated close to the surface and therefore it is hard to obtain good depth resolution. The physics of gravity is well explained and it is a common tool in geophysical prospecting. Numerous papers and books are written on modelling and inversion of gravity data (see for example Blakely [1995], Li [1996], Mirzaei [1996]), and the main difficulty in applying an inversion scheme is the size of the problem. In this section we test our algorithms on the one and two-dimension gravity problem, experiment with the linearized nonlinear gravity problem and finally apply the best strategy to a three dimension field example.

6.1.1 The 1-D Gravity Problem

The goal of this section is to compare different methods for estimating noise. We have four such methods: GCV in the full space (GCV-F), GCV in subspace (GCV-S), L-curve in full space (L-F) and L-curve in subspace (L-S). We are going to test these methods on different gravity problems. The tests are made from synthetic data sets for each of these problems. In the first stage we add different amounts of Gaussian noise to the data, varying from $0.1 - 30\%$ i.e.:

$$b_i \rightarrow b_i + N_i(0, \alpha b_i) \qquad \alpha = 0.001...0.3 \tag{6.1}$$

where $N_i(0, \alpha b_i)$ is a Gaussian random variable with zero mean and αb_i standard deviation. Real world data are seldom strictly Gaussian and therefore we test our noise

estimation methods with a combination of Gaussians:

$$b_i \rightarrow b_i + N_i(0, \alpha b_i) + N_i(0, \sigma ||b||) \tag{6.2}$$

where $\alpha = 0.01...0.3$ and $\sigma = 0.01...0.2$. We then test the methods for two extreme cases. We generate Gaussian correlated noise and test the case:

$$b \rightarrow b + R\ N(0, \alpha ||E(b)||) \tag{6.3}$$

where $N(0, \alpha||E(b)||)$ is a vector of random numbers, each number is generated from a Gaussian distribution which has a zero mean and $\alpha||E(b)||$ standard deviation. $R^T R = C$, and C is a data correlation matrix. Finally, we add this correlated noise to Gaussian non-correlated noise and test the case where the noise is:

$$b \rightarrow b + R\ N(0, \alpha||b||) + N(0, \sigma||b||) \tag{6.4}$$

Recall from Chapter 1 Section 2 that the integral equation in the 1-D gravity case is:

$$b_i = \gamma \int_0^\infty (\frac{\pi}{2} - arctan(\frac{x_i}{z}))\Delta\rho(z)dz \tag{6.5}$$

where x_i is the position of the measurement and $\Delta\rho(z)$ is the anomaly density model, and γ is the gravitation constant.

In order to carry out the synthetic experiment, we assume that we measure 32 data points at distances 0 to 70 meters from the fault. We choose a smooth model and calculate the data by discretizing the integral equation 6.5 into 129 grid points using the trapezoidal rule as explained in Chapter 2. The kernels are plotted in Figure 6.1. The kernels decay rapidly with depth and therefore we do not expect to obtain good depth resolution. The important role which the spectrum plays in the problem was emphasized in Chapter 4 . Since this problem is small, we calculate the SVD of the discrete system and plot the singular values (Figure 6.2). The singular values decay rapidly and we plot

the function e^{-n} (where n is the singular value index) to demonstrate that. The problem is very ill-conditioned as the condition number is $1E17$. The model and the data used for this example are plotted in Figure 6.3 As a final stage of the forward modelling we add noise to the data. To summarize the process of the forward modeling we have:

- Choose 32 measurement points equally spaced from $0 - 70$ meters.
- Pick a model.
- Discretize the system using 129 grid points and calculate the integral using the midpoint rule.
- Add noise to the data.

In order to carry out the inversion we need to choose an objective function. For this example, we chose an objective function which is a discrete representation of the continuous operator:

$$[0.1\nabla^2, f(z)I]^T$$

When discretizing the operator ∇^2 we do not use boundary conditions. The function $f(z)$ is plotted in Figure 6.4. The function penalizes the amplitude of the model at the surface and at great depth.

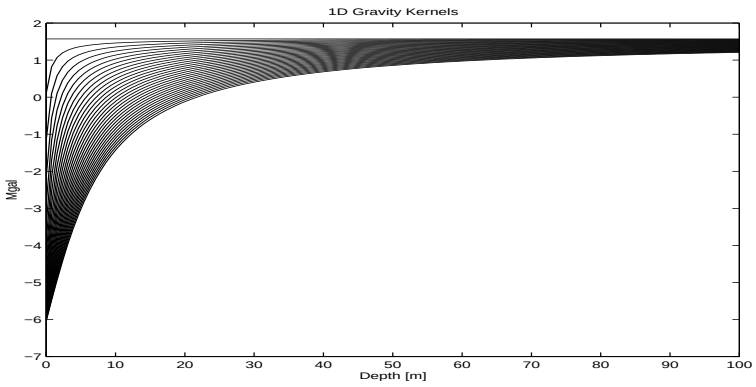

Figure 6.1: 1-D Gravity Kernels

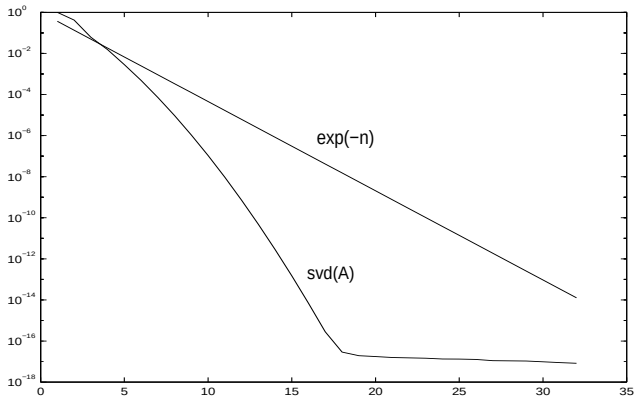

Figure 6.2: 1-D Gravity Singular Values

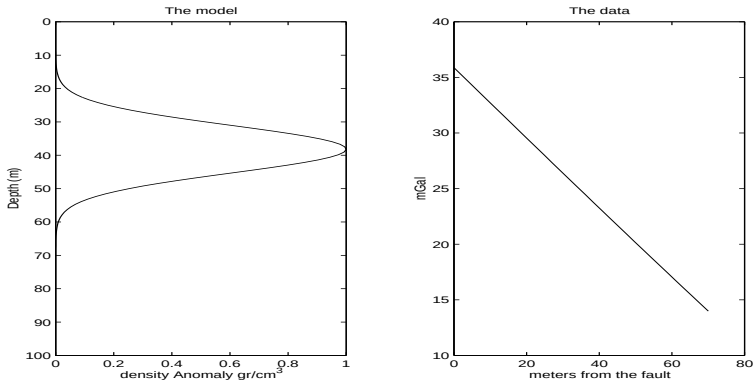

Figure 6.3: 1-D Gravity model and data

We then invert the system using different noise levels in the right hand side. Table 6.1 shows the misfit which was produced with different methods in the case of Gaussian noise. The curves of the GCV-F and L-F are plotted for high level noise (20%) and for low level noise 0.1% in Figure 6.5.

Secondly we carry out the same experiment but with the right hand side contaminated by noise from two Gaussian distributions. The first component of the noise is proportional to each datum and the second is proportional to the norm of the data. In Table 6.2 we summarize the results of the predicted misfit for this case.

Noise	GCV-F	GCV-S	L-F	L-S	True $\|\|\epsilon\|\|$
30%	3.2E-1	3.3E-1	3.4E-1	3.3E-1	3.4E-1
20%	1.6E-1	1.5E-1	1.6E-1	1.6E-1	1.8E-1
10%	1.1E-1	1.1E-1	1.1E-1	1.1E-1	1.1E-1
5%	6.1E-2	6.3E-2	6.3E-2	6.8E-2	6.5E-2
1%	8.4E-3	8.2E-3	9.1E-3	8.2E-3	9.3E-3
0.1%	9.2E-4	9.1E-4	9.6E-4	9.6E-4	0.10E-3

Table 6.1: Predicted square root of the misfit using the different methods versus the true square root of the misfit $||\epsilon||$. The noise is Gaussian $\epsilon_i = N_i(0, \alpha b_i)$

Noise	GCV-F	GCV-S	L-F	L-S	True $\|\|\epsilon\|\|$
$0.3N_1 + 0.1N_2$	2.7E-1	2.7E-1	2.8E-1	2.7E-1	2.9E-1
$0.2N_1 + 0.1N_2$	1.9E-1	1.9E-1	1.9E-1	1.8E-1	1.9E-1
$0.1N_1 + 0.1N_2$	1.2E-1	1.1E-1	1.2E-1	1.2E-1	1.2E-1
$0.05N_1 + 0.05N_2$	6.7E-2	6.6E-2	7.1E-2	5.6E-2	7.1E-2
$0.01N_1 + 0.01N_2$	1.3E-2	1.1E-2	1.3E-2	1.3E-2	1.3E-2
$0.001N_1 + 0.001N_2$	1.5E-3	1.5E-3	1.6E-3	1.5E-3	1.5E-3

Table 6.2: Predicted square root of the misfit using the different method versus the true square root of the misfit $||\epsilon||$. The noise in the first column is made from a combination of $\alpha_1 N_1 + \alpha_2 N_2$. N_1 is Gaussian with 0 mean and standard deviation which is proportional to the datum. N_2 is Gaussian with 0 mean and standard deviation which is proportional to the norm of the data.

Noise	GCV-F	GCV-S	L-F	L-S	True $\|\|\epsilon\|\|$
30%	1.9E-1	1.9E-1	3.3E-1	2.4E-1	3.0E-1
20%	1.2E-1	1.3E-1	1.3E-1	1.3E-1	2.1E-1
10%	5.3E-2	4.3E-2	6.2E-2	7.2E-2	9.6E-2
5%	2.3E-2	2.1E-2	4.9E-2	3.1E-2	5.1E-2
1%	1.0E-2	1.1E-2	1.3E-2	1.0E-2	1.9E-2
0.1%	8.6E-4	9.5E-4	1.0E-3	1.0E-3	1.7E-3

Table 6.3: Predicted square root of the misfit using the different methods versus the true square root of the misfit $||\epsilon||$. The errors are correlated.

In the next stage we add correlated noise to the data. Table 6.3 summarized this experiment. The covariance matrix is plotted in Figure 6.6. The covariance for a datum b_i measured at location L_i and a datum b_j measured at location L_j is assumed to be:

$$COV(b_i(L_i), b_j(L_j)) = ce^{-\frac{|L_i - L_j|}{L}}$$

where L is a characteristic distance and c is a constant. For this example we use $L = 10m$. Finally in Table 6.4 we combine our correlated noise with Gaussian uncorrelated noise with a uniform standard deviation proportional to the norm of the data. We use the same covariance matrix. Table 6.4 which contains the combinations of the correlated

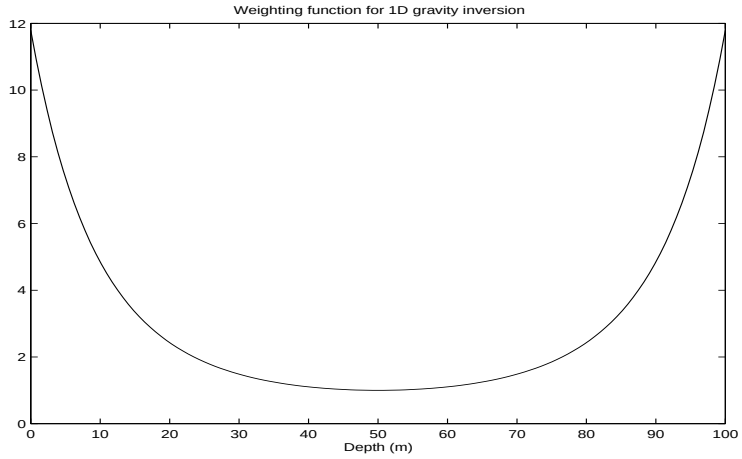

Figure 6.4: Weighting function used for the inversion

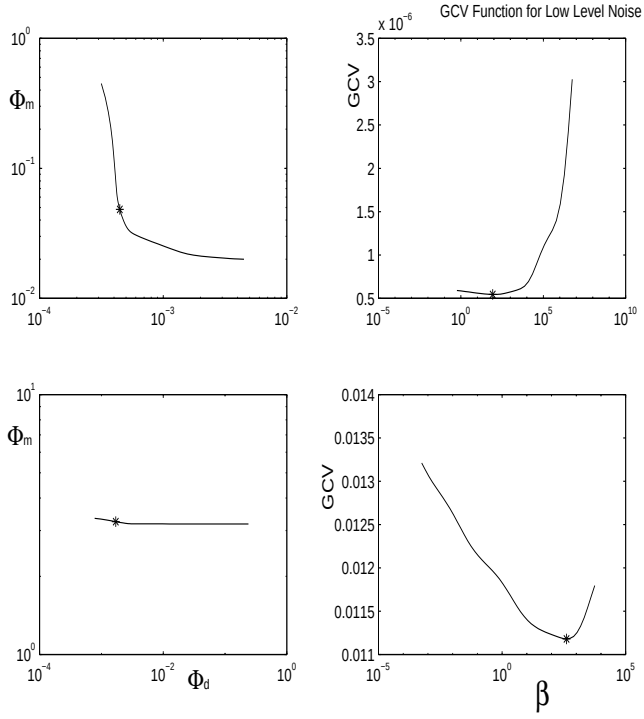

Figure 6.5: GCV and L-curves for 20% noise case (bottom) and for the 0.1% noise (top). The star represents the point chosen for noise estimation.

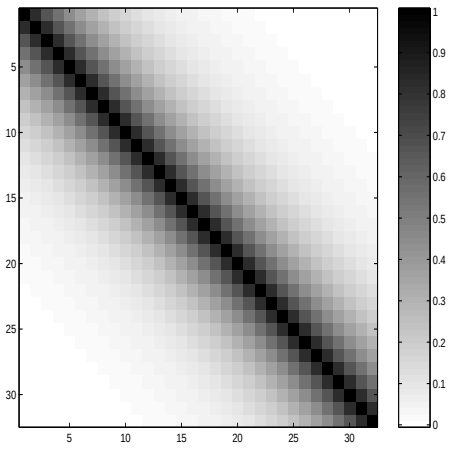

Figure 6.6: Covariance Matrix for 1-D example

Noise	GCV-F	GCV-S	L-F	L-S	True $\|\|\epsilon_{nc}\|\|$	True $\|\|\epsilon_c\|\|$
$0.1N + 0.2C$	1.2E-1	1.3E-1	1.4E-1	1.4E-1	1.0E-1	2.1E-1
$0.05N + 0.2C$	1.1E-1	1.4E-1	1.5E-1	1.4E-1	7.3E-2	2.4E-1
$0.1N + 0.1C$	1.1E-1	1.E-1	1.7E-1	1.1E-1	1.0E-1	1.0E-1
$0.01N + 0.1C$	9.1E-1	8.7E-1	7.8E-1	5.1E-1	7.2E-1	9.9E-1
$0.05N + 0.05C$	5.0E-2	4.7E-2	5.3E-2	5.2E-2	4.3E-2	4.8E-2
$0.1N + 0.05C$	9.3E-2	9.4E-2	9.8E-2	9.8E-2	1.0E-1	4.8E-2
$0.05N + 0.01C$	5.5E-2	5.4E-2	5.6E-2	5.4E-2	5.8E-2	1.0E-2

Table 6.4: Predicted square root of the misfit using the different method versus the true correlated noise $||\epsilon||_c$ and the non-correlated noise $||\epsilon||_{nc}$.

and non-correlated noise is interesting. It shows that our methods tend to detect the non-correlated noise and ignore the correlated noise. This observation for the GCV is not new and Wahba [1990] refers to other such observations. One last aspect which does not appear in the tables is the behaviour of the model norm. The model norm of all the techniques above was equivalent and therefore at least from an interpretation point of view the results obtained from different methods are comparable. This can be seen graphically in Figure 6.7, where we plot the different models which are obtained in the 1% case. Further examination of the model norm with different methods is discussed in the 2-D case.

As a summary for the 1-D example, from the Tables and Figures we conclude:

- When the noise is Gaussian with range $1 - 20\%$ all methods did well.
- When the noise level was very low GCV methods could have problems. The minimum of the GCV function was flat and the regularization parameter was not well-determined. This occurred when noise levels were lower than 0.1%. However, all GCV methods preformed well and detected the noise level quite accurately.
- When the noise level was very low L-curve methods were well-defined.

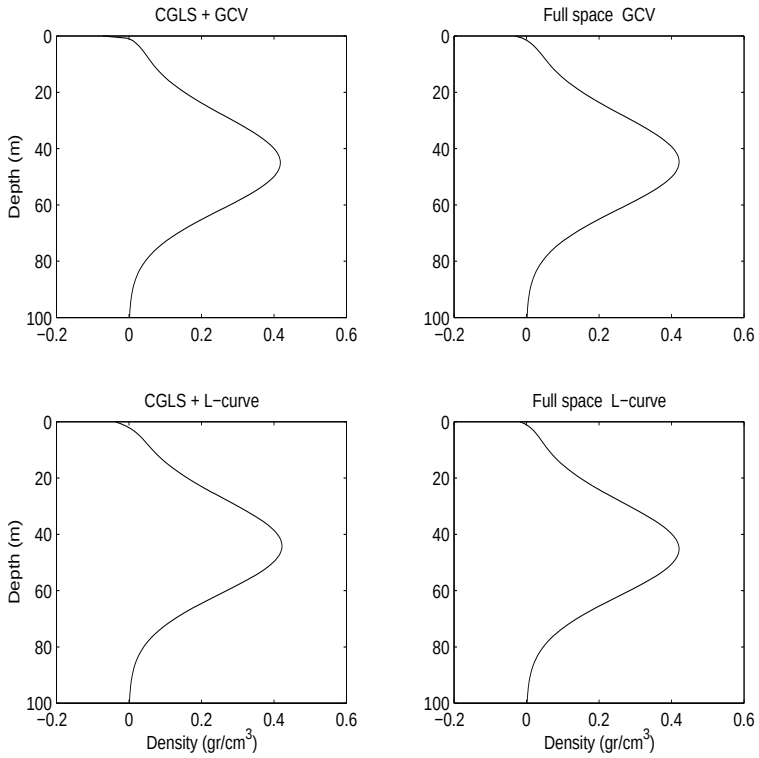

Figure 6.7: Models which are obtained using the different methods in the 1% case.

- When the noise level was high, L-curve methods had a problem. The L-curve did not have the typical L-shape, and therefore detecting the corner was hard.
- When the noise level was high, the GCV function possessed a sharp minimum.
- All methods did well when the noise was not strictly Gaussian, so long as the noise was not correlated.
- All methods did not do well when the noise was correlated. The predicted square root of the misfit is lower than the true square root of the misfit by a factor of 2.
- When noise is made from a correlated part and a non-correlated part, the methods tend to detect mainly the non-correlated part and the square root of the misfit was roughly equivalent to the norm of the non-correlated noise.

The above experiments demonstrate the robustness of the techniques we have in estimating noise levels in different scenarios. The experiments also show that subspace methods are not only computationally efficient, but they are robust for noise estimation and give practically the same results as full-space methods. In the next examples the size of the problem solved is larger and Tichonov regularization in the full space is very expensive. We therefore use mainly subspace and hybrid methods.

6.1.2 The 2-D Gravity Problem

In this section we concentrate on the amount of computation needed for the solution of a problem. We test subspace methods: CGLS with GCV, CGLS with the L-curve, multilevel iteration with GCV and gradients with GCV. We compare the number of floating points operations (flops) which are needed to solve the problem for noise levels of $5, 10$ and 20%. We then carry out the same comparison between hybrid methods: hybrid LSQR, iterated Krylov and iterated gradient.

Recall that the gravity data in two dimensions is given by the integral equation:

$$b_i = b(x_i, h) = \gamma \int_D \frac{\Delta\rho(x,z)\ z}{(x - x_i)^2 + (z + h)^2} dx\ dz \tag{6.6}$$

We assume we measure 200 data, $b_i = b(x_i, h)\ \ i = 1..200$. The data are assumed to be measured at height $h = 0.5$ meters above the surface, at equally spaced intervals from 0 to 100 meters. In order to calculate the data we pick a model made of two positive anomalies. The model is plotted in Figure 6.9 (bottom). In order to calculate the data, the integral 6.6 is discretized into $50 \times 30 = 1500$ cells and the integral is approximated using the midpoint rule as explained in Chapter 2. As in the 1-D case, the kernels decay rapidly with depth. Although the number of cells used for the problem is large, we have data only above the surface and therefore the number of data is not large. We can therefore calculate the eigenvalues of AA^T which is an $N \times N$ matrix. The eigenvalues of this system are the squared singular values of the system. The square root of the eigenvalues of AA^T (which are the singular values of the discrete 2-D gravity system) are plotted in Figure 6.8. This time the singular values decay in steps. This is an important observation for the implementation of iterative methods. Recall from Chapter 4 that Krylov space methods work well on such problems. The problem is ill-conditioned as the condition number is $3.6E8$.

To summarize the process of calculating the data and discretizing the system we have:

- Measured 200 data points 0.5 meters above the ground.
- The data are evenly spaced over the interval $0 - 100$.
- The model is made of two smooth anomalies.
- The model space is discretized into 50×30 points.
- The data are calculated using the midpoint rule.

Since the model is smooth and the kernels are localized near the surface, we chose an objective function which minimizes roughness and penalizes structure close to the surface and at infinity. We discretize the Laplacian operator in 2-D without using boundary conditions. Depth penalization is obtained by a weighting function $f(x, y)$ which is plotted in Figure 6.10. The function is assumed to represent *a priori* information. Thus, the final weighting operator is:

$$W = -\nabla^2 + f(x, y)I$$

This problem is of a large scale and we cannot easily calculate the product $A^T A$, store it, and invert $A^T A + \beta W^T W$ for different β's. We therefore use this problem to

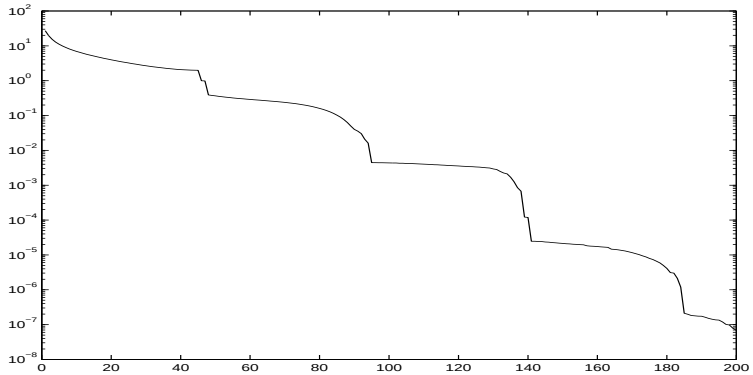

Figure 6.8: 2-D Gravity singular values

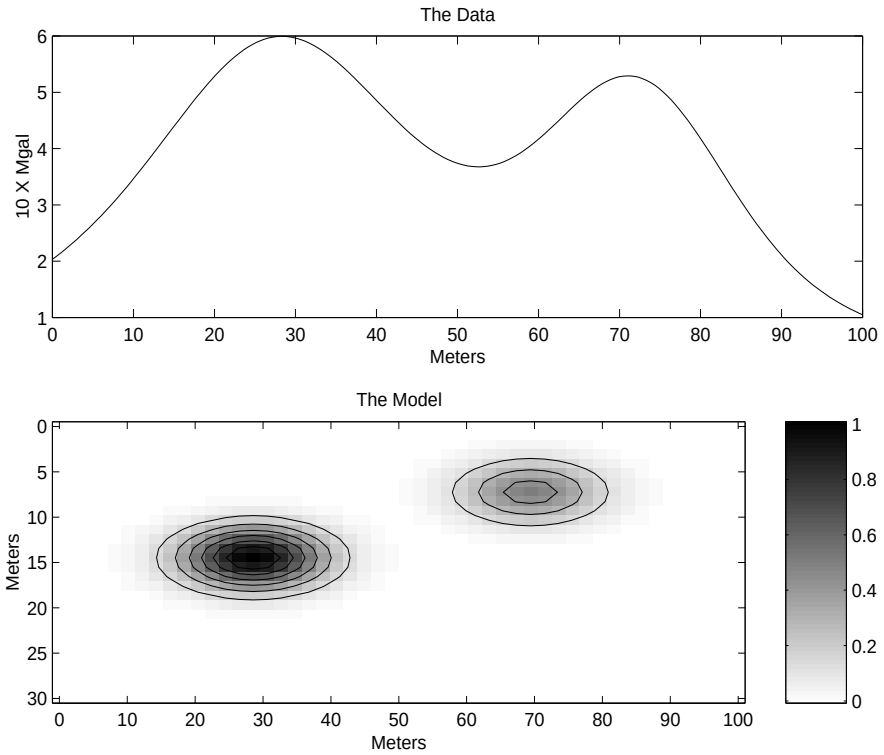

Figure 6.9: 2-D Gravity Model and Data

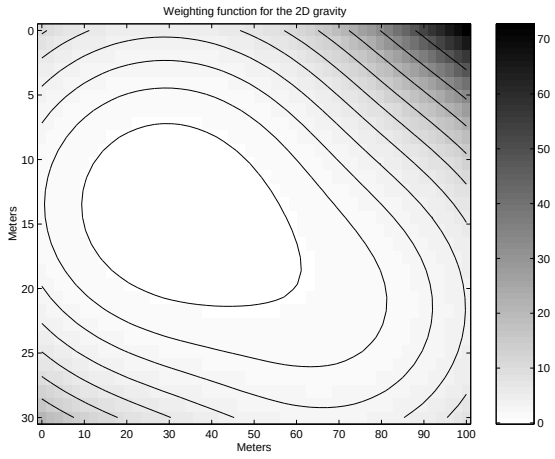

Figure 6.10: 2-D gravity weighting function

test subspace methods, i.e. CGLS-LSQR, multilevel and gradients. We also test hybrid methods, i.e. Lanczos, iterated Krylov and iterated gradients. While the main goal of the last section was to test different noise estimation methods, the emphasis of this section is on computational properties. Our goal is to compare the amount of computations needed in order to solve the system using a specific regularization method and a specific method for noise estimation. We test the difference between the solutions by looking at the model norm of each of the solutions and we compare the predicted and real misfit. The results are in Tables 6.5-7.

Method	Pred $\|\|Ax-b\|\|$	ϕ_m	flops
CGLS+GCV	973.2	0.173	6.3E7
CGLS+L	973.2	0.173	7.8E7
Multilevel+GCV	971.2	0.171	19E7
Gradients+GCV	973.9	0.175	13E7

Table 6.5: Comparison between Subspace Methods for the solution of the 2-D gravity problem. Noise level is 20%, real square root of the misfit is 978.45.

Method	Pred $\|\|Ax-b\|\|$	ϕ_m	flops
CGLS+GCV	513.4	0.166	6.3E7
CGLS+L	513.4	0.166	8.4E7
Multilevel+GCV	515.2	0.165	20E7
Gradients+GCV	513.9	0.164	14E7

Table 6.6: Comparison between Subspace Methods for the solution of the 2-D gravity problem. Noise level is 10%, real square root of the misfit is 515.57.

The inverted models using CGLS+GCV, hybrid LSQR and using gradients with GCV for the case of 10% are plotted in Figure 6.11.

From the above Tables we conclude the following:

- All methods are comparable in terms of predicting the misfit.

- All methods are comparable in terms of the model norm.
- CGLS is the most computationally effective. The gradients and multilevel methods are two to three times more expensive.

Our next experiment is the comparison of hybrid methods. Here we compare Lanczos bidiagonalization, iterated Krylov and iterated gradients. We use the GCV as a stopping criterion. Results for $5, 10$ and 20% noise levels are in Tables 6.8-10.

The result of the inversion of hybrid LSQR is in Figure 6.11. From the Tables we conclude that:

- All methods are comparable in terms of predicting the misfit.
- All methods are comparable in terms of the model norm.
- Hybrid LSQR is the most computationally effective. Iterated Krylov and iterated gradients are more or less equivalent and are $1.5 - 2$ times more expensive than hybrid LSQR.

We end this subsection with a comparison of the most effective subspace, which is the Krylov subspace (CGLS and LSQR) and hybrid methods. Notice that the difference in model norm is small and from an interpretation point of view, the models are equivalent. However from the number of computations point of view, Krylov subspace methods need

Method	Pred $\|\|Ax - b\|\|$	ϕ_m	flops
CGLS+GCV	270.3	0.148	6.3E7
CGLS+L	279.2	0.142	9.5E7
Multilevel+GCV	275.2	0.144	20E7
Gradients+GCV	270.4	0.150	14E7

Table 6.7: Comparison between Subspace Methods for the solution of the 2-D gravity problem. Noise level is 5%, real square root of the misfit is 273.4.

Method	Pred $\|\|Ax-b\|\|$	ϕ_m	flops
H-LSQR	269.7	0.143	1.3E8
I-KRY	269.7	0.144	2.1E8
I-GRAD	270.3	0.148	2.7E8

Table 6.8: Comparison between Hybrid Methods for the solution of the 2-D gravity problem. Noise level is 5%. H-LSQR - hybrid LSQR, I-KRY - iterated Krylov, I-GRAD - iterated gradients. True square root of the misfit was 273.4.

Method	Pred $\|\|Ax-b\|\|$	ϕ_m	flops
H-LSQR	510.8	0.160	1.3E8
I-KRY	512.4	0.1603	2.1E8
I-GRAD	511.7	0.158	2.7E8

Table 6.9: Comparison between Hybrid Methods for the solution of the 2-D gravity problem. Noise level is 10%. H-LSQR - hybrid LSQR, I-KRY - iterated Krylov, I-GRAD - iterated gradients. True square root of the misfit was 515.5.

only half of the amount of flops and are substantially more efficient. In terms of storage, subspace methods need to store only three vectors which belong to the model space while hybrid methods need to store more. We therefore conclude that unless there is no other option, it is better to use Krylov subspace methods over hybrid methods.

Method	Pred $\|\|Ax - b\|\|$	ϕ_m	flops
H-LSQR	973.7	0.173	1.3E8
I-KRY	970.4	0.175	2.0E8
I-GRAD	974.1	0.171	2.6E8

Table 6.10: Comparison between Hybrid Methods for the solution of the 2-D gravity problem. Noise level is 20%. H-LSQR - hybrid LSQR, I-KRY - iterated Krylov, I-GRAD - iterated gradients. True square root of the misfit was 983.3

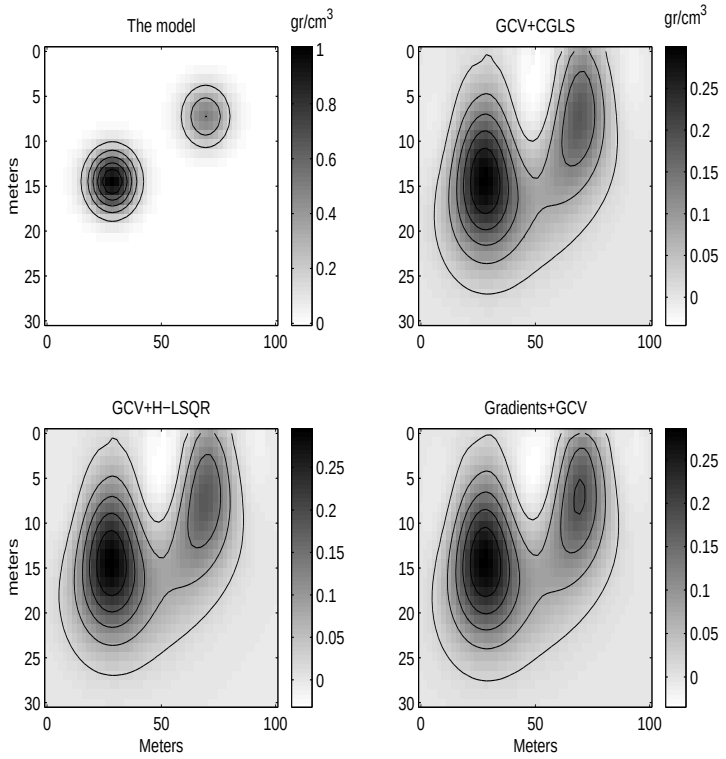

Figure 6.11: 2-D Gravity Inversion

6.1.3 Imaging of Nonlinear Gravity

Another important type of problem is imaging. This type of problem arises, in general, when a nonlinear problem is linearized. Assume that we have a nonlinear problem:

$$F[x] + \epsilon = b \tag{6.7}$$

The operator F can be very complicated and we might not know how to calculate it. However for some situations we could write F as:

$$F[x_0 + (x - x_0)] = F[x_0] + A(x_0)(x - x_0) + R(x_0, x) \tag{6.8}$$

where we know how to calculate the linear operator $A(x_0)$ analytically. Such examples are the Born and Rytov approximations (Born [1975]). In this case we can solve the linear problem for x:

$$A(x_0)x + R(x_0, x) + \epsilon = b - F[x_0] + A(x_0)x_0 \tag{6.9}$$

It is of great interest to check how our linear inversion algorithms work for this case, first, because this type of approximation is very common in industrial settings (Claerbout [1985]) but more importantly, the "noise" for the linear operator is made of a Gaussian random noise and the nonlinear part $R(x_0, x)$ that is not necessarily small and is obviously correlated. It is interesting to observe the result of an imaging algorithm for this very different type of noise. This is the goal of this test example. In parallel with the Born approximation, we use the Fréchet derivative operator derived from a half space as the operator $A(x_0)$.

Recall from Chapter 1 that the gravity measurement b_j due to a change in the subsurface topography $m(x, y)$ from the assumed subsurface topography $h(x, y)$ is given by (setting $\gamma\Delta\rho = 1$):

$$b_j = \int\int_{\mathcal{D}} \frac{1}{r_h(x, y, h(x, y); x_j, y_j)} - \frac{1}{r_m(x, y, m(x, y); x_j, y_j)} dxdy \tag{6.10}$$

with:

$$r_h(x, y, h(x, y); x_j, y_j) = \sqrt{(x - x_j)^2 + (y - y_j)^2 + h^2}$$

and

$$r_m(x, y, m(x, y); x_j, y_j) = \sqrt{(x - x_j)^2 + (y - y_j)^2 + (h + m)^2}$$

In order to experiment with this type of problem we assume that 30×30 gravity data are measured on the surface. The data are equally gridded in the interval $[0, 100] \times [0, 100]$ meters. In order to calculate the data we pick a model with average depth of $20m$. We therefore set the reference height $h(x, y) = 20$ meters in equation 6.10. The model, $m(x, y)$, has zero mean about this surface. The integral 6.10 is calculated with the midpoint rule. The $100m \times 100m$ square domain is divided into 49×49 grid points. The model and the data are plotted in Figure 6.12.

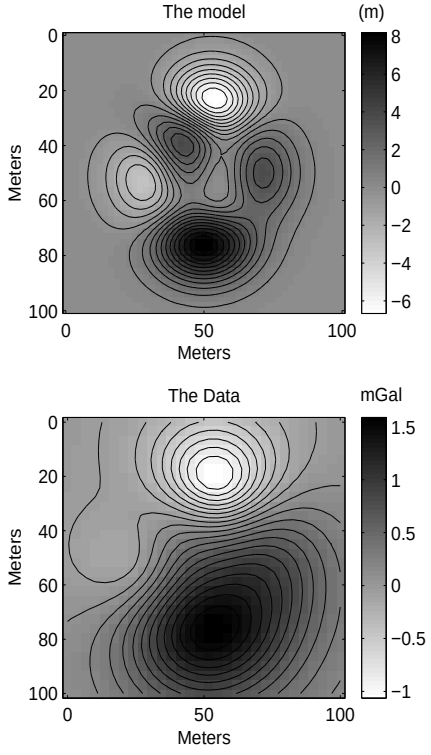

Figure 6.12: The model and data for the nonlinear example.

In order to calculate $A(0) = A(m = 0)$ we set $m_0(x, y) = 0$. Recall from Chapter 1 that the Fréchet derivative operator in this case is given by:

$$A(0)(.) = \int\int_{\mathcal{D}} \frac{h(x,y)(.)dxdy}{[(x - x_j)^2 + (y - y_j)^2 + h(x,y)^2]^{\frac{3}{2}}} \tag{6.11}$$

Using the same discretization, the operator is discretized into a matrix A. We used 2401 model parameters and we have 900 data and therefore the size of A is 900×2401. This leads to the linear system:

$$Ax = b - F_0 \tag{6.12}$$

where:

$$F_0 = F[m = 0] \tag{6.13}$$

Again A is large and full and again direct methods cannot be applied. We therefore solve this problem using subspace and hybrid methods. This example is used to test noise estimation techniques and the computational properties of the different methods. The singular values which are larger than $1E - 9$ are plotted in Figure 6.13, and a typical kernel is plotted in figure 6.14. Note that the kernel is very different from the ones we

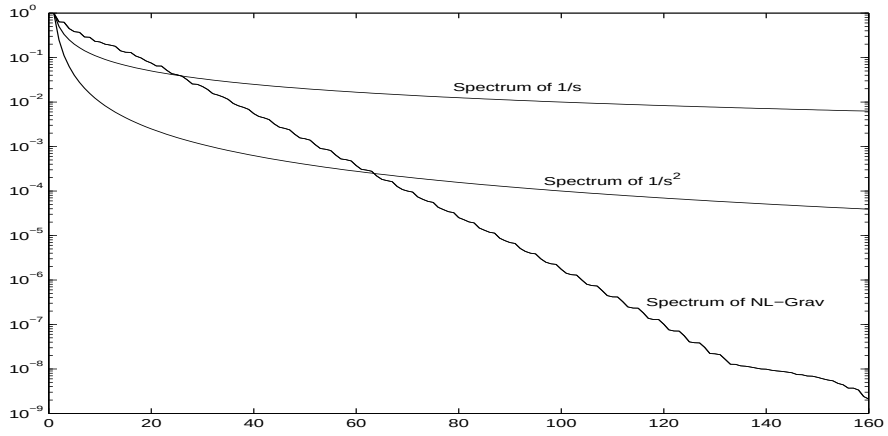

Figure 6.13: 2-D Nonlinear Gravity SVD

encountered so far. The kernel is localized around the point of observation. This type

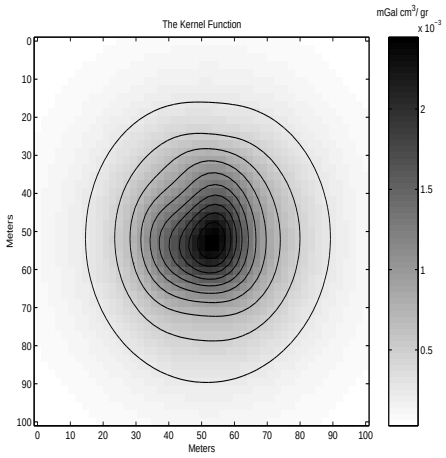

Figure 6.14: A nonlinear gravity kernel for the data point measured at point $[50, 50]$

of kernel is ill-posed, and with a spectrum which decays slowly. This is demonstrated in Figure 6.13 where we can see that the functions a/n and a^2/n^2 (a is used for scaling), decay faster than the spectrum of this problem at least for the first 40 singular values. We therefore anticipate slow convergence when the CGLS algorithm is used.

In order to test our algorithms we add random Gaussian noise to the data and invert the system using the different methods. We compare the prediction of noise (the random and the nonlinear) by the different methods and their computational properties. We compare the square root of the linear misfit, $\sqrt{\phi_d}^{lin} = ||Ax - (b - F_0)||$, the square root of the nonlinear misfit, $\sqrt{\phi_d}^{nonlin} = ||b - F[x]||$, the model norm and the number of flops needed for the inversion. While the linear misfit represents how well the system was inverted, the nonlinear misfit provides information about the nonlinearity of the problem. If the linear misfit and the nonlinear misfit are similar then the nonlinear operator is approximated well by the linear one and the term $R(x_0, x)$ is small. However if the linear misfit is significantly different than the nonlinear one then the nonlinear term $R(x_0, x)$ is large and the linear approximation breaks down. Our goal is to test our methods exactly under this different condition. This test would serve us later when

Method	$\|\|Ax - b + F_0\|\|$	$\|\|F[x] - b\|\|$	ϕ_m	flops
CGLS+GCV	0.27	3.71	9.47	4.4E8
CGLS+L	0.27	3.71	9.47	6.3E8
LSQR+GCV	0.26	4.10	9.60	5.6E8
LSQR+L	0.26	4.10	9.60	6.6E8
GRAD+GCV	0.28	3.62	9.31	25E8
ML+G	0.31	3.51	8.99	31E8
H-LSQR	0.33	3.72	9.23	14E8
I-KRY	0.31	3.41	9.31	42E8
I-GRAD	0.35	3.23	9.11	44E8

Table 6.11: Comparison between all methods for 1% noise. True square root of the misfit is 0.25.

Method	$\|\|Ax - b + F_0\|\|$	$\|\|F[x] - b\|\|$	ϕ_m	flops
CGLS+GCV	1.25	4.31	9.23	3.6E8
CGLS+L	1.25	4.31	9.23	5.2E8
LSQR+GCV	1.23	4.32	9.26	4.7E8
LSQR+L	1.23	4.32	9.26	5.4E8
GRAD+GCV	1.53	4.67	8.77	22E8
ML+G	1.29	4.39	9.20	26E8
H-LSQR	1.31	4.21	9.15	12E8
I-KRY	1.36	4.18	9.10	35E8
I-GRAD	1.33	4.15	9.06	36E8

Table 6.12: Comparison between all methods for 5% noise. True square root of the misfit is 1.23.

solving the nonlinear problem. Results of the different experiments are in Tables 6.11-13.

In Figure 6.15 we plot the models which were obtained using the different methods for the 10% noise level. From the Figure and Tables we conclude that:

- Predicted linear misfit is approximately equal to the random noise. This is observed both for the case of large nonlinear terms versus random noise and for the case that

Method	$\|\|Ax - b + F_0\|\|$	$\|\|F[x] - b\|\|$	ϕ_m	flops
CGLS+GCV	2.50	4.81	8.97	1.1E8
CGLS+L	2.50	4.81	8.97	1.1E8
LSQR+GCV	2.53	4.79	8.9	2.0E8
LSQR+L	2.53	4.79	8.9	2.0E8
GRAD+GCV	2.98	4.11	7.9	2.0E8
ML+G	3.29	3.91	6.88	22E8
H-LSQR	2.66	4.06	6.95	9.2E8
I-KRY	2.73	3.92	6.81	27E8
I-GRAD	2.79	3.81	6.75	29E8

Table 6.13: Comparison between all methods for 10% noise. True square root of the misfit is 2.50.

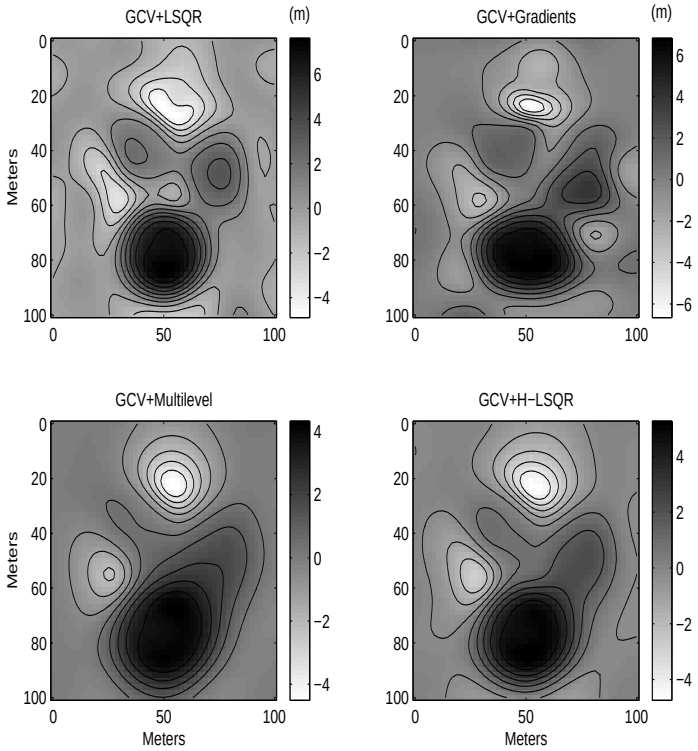

Figure 6.15: 2-D Nonlinear Gravity Imaging.

the random noise is of the scale of the nonlinear terms.

- All methods predict random errors but do not predict nonlinear terms. The methods tend to see the nonlinear terms as a part of the signal.
- The models obtained using different methods are comparable. This can be seen by looking at the model norm and Figure 6.15. However hybrid solutions tend to be smoother.
- From a computational point of view, CGLS and LSQR with a combination of GCV are the most efficient methods.

6.1.4 3-D Gravity Problem

The last of the gravity problems tackled in this thesis is a large 3-D gravity problem. Three-dimensional gravity is a common measurement in geophysics however the shear size of the problem often prevents any attempt to invert it, and usually conclusions are based solely on interpreting the data. The goal of this subsection is to apply the methods developed previously to a real field data example. In order to do this we first test our algorithm on a synthetic example. Recall from Chapter 1 that the gravity data is given by (setting $\gamma = 1$):

$$b(x_i, y_i) = \int\int\int_D \frac{z\ \Delta\rho(x, y, z)\ dx\ dy\ dz}{[(x - x_i)^2 + (y - y_i)^2 + z^2]^{\frac{3}{2}}} \tag{6.14}$$

In order to make the test as close as possible to the real data example, we assume we measure 982 data on the surface, located at the same place as the real data. The measurement points are plotted in Figure 6.16. We divide the earth into $40 \times 40 \times 20 =$

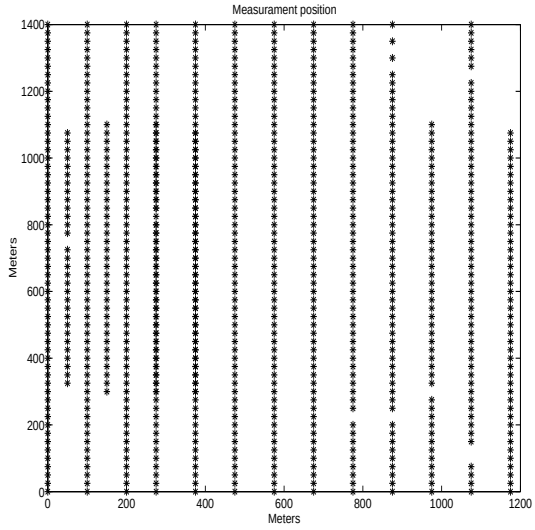

Figure 6.16: Measurement points in the field data set.

32000 cells, and carried out the integration using an analytic expression for each cell (Nagi [1966]). The synthetic experiment emulates the field data set and therefore, the kernels

of the synthetic example are the same as the field data example which will be tested later. This discretization leads to a full system of size 982×32000. The system cannot be stored and therefore we never compute the system directly. Since we use iterative methods which need only the calculation of the product of a matrix and a vector and the product of the matrix transpose and a vector, we calculate these products as follows: First we calculate the forward product of the matrix A and an arbitrary vector v by calculating each datum separately using the midpoint rule on the integral equation 6.14. In this way we avoid storing the matrix, however this means that we have to calculate the elements of the matrix A for each matrix-vector product. The transpose product is given by reciprocity. Noticing that the operation of row i of A^T on a vector u ($u \in R^N$ describes the gravity measurement on the surface) is equivalent to the gravity measurement in cell i due to a density which is distributed on the surface and given by the vector u. We can therefore calculate Av and $A^T u$ and we are ready to carry out the inversion.

We have developed a variety of methods for doing the inversion but, since the problem is so large, we want to use the most efficient one. The obvious choice is the combination of CGLS and GCV, however in order to be certain that this choice is justified we need to know something about the spectrum of our problem. Since the problem is so large, there is no chance to calculate the SVD directly. However we do not need to know the exact SVD of the problem, and it would be sufficient to know the behaviour of the spectrum. We therefore discretize the problem using $11 \times 11 \times 8$ grid points and check the spectrum of this system. The spectrum is plotted in Figure 6.17. The spectrum is of the steps type and therefore we feel safe with the choice of CGLS+GCV. We proceed by picking a model and calculating the data. The model and the data are plotted in Figure 6.18.

In order to invert the system we need to choose a weighting matrix and we choose the discrete version of $W = -\nabla^2 + \epsilon I$. The operator ∇^2 does not have boundary conditions and therefore is singular. We use $\epsilon = 0.01$ to ensure that the operator is positive definite.

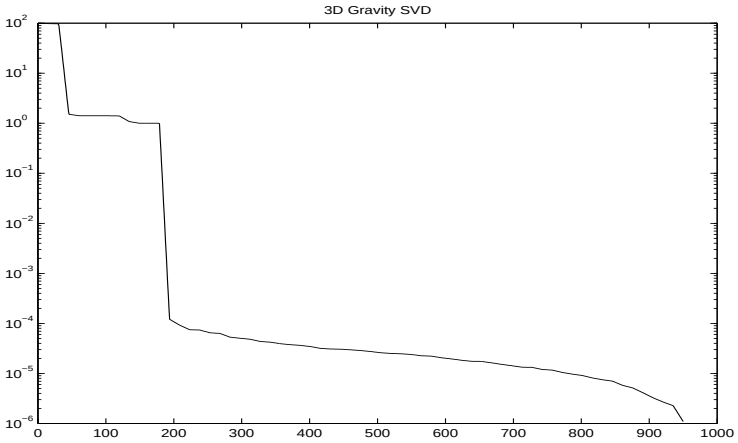

Figure 6.17: Estimation of the SVD of the 3-D gravity matrix.

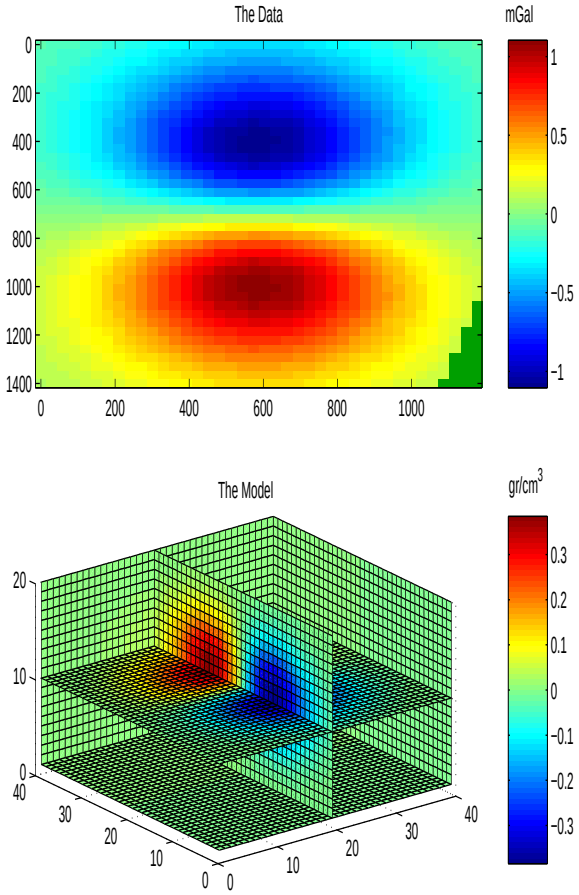

Figure 6.18: Three Dimensional model and data.

Noise Level	Number of Iterations	$\|\|Ax-b\|\|$	$\|\|\epsilon\|\|$	flops
5%	18	1.357E4	1.346E4	1.9E9
10%	16	2.745E4	2.853E4	1.7E9
20%	12	5.341E8	5.431E4	1.4E9

Table 6.14: Performance of the CGLS+GCV algorithm for the inversion of 3-D gravity

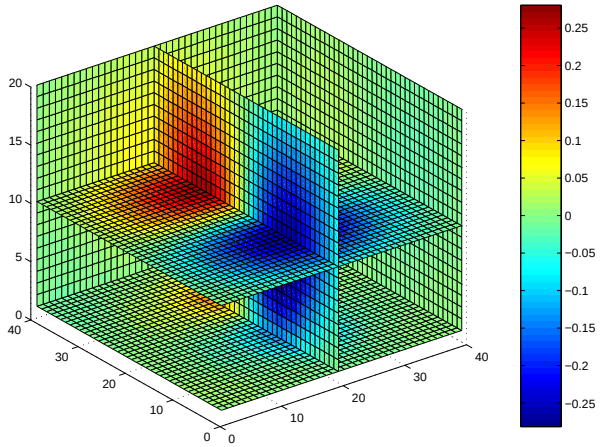

Figure 6.19: Results of 3-D gravity inversion.

The results of the inversion are summarized in Table 6.14 and plotted in Figure 6.19. From the table we see that the methods we developed could predict the noise levels for this problem and produced reasonable models.

Next we use the same procedure to invert the field data from Heath-Steel-Stratmat. The data are plotted in Figure 6.20. The results of the inversion are plotted in Figure 6.21. The result of this inversion agrees well with other inversions of the same data set (Li [1996]) which were obtained using gradient subspace methods and fits to the geology of the area.

As a conclusion to this section we see that the methodologies which where developed allow us to work with large problems, calculate solutions and predict noise. Again the conjugate gradient type algorithm has been found to be robust and easy to implement.

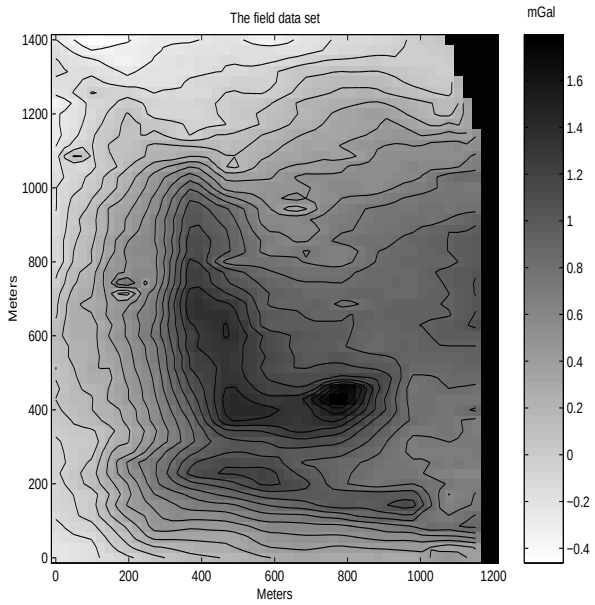

Figure 6.20: The 3-D Gravity data set

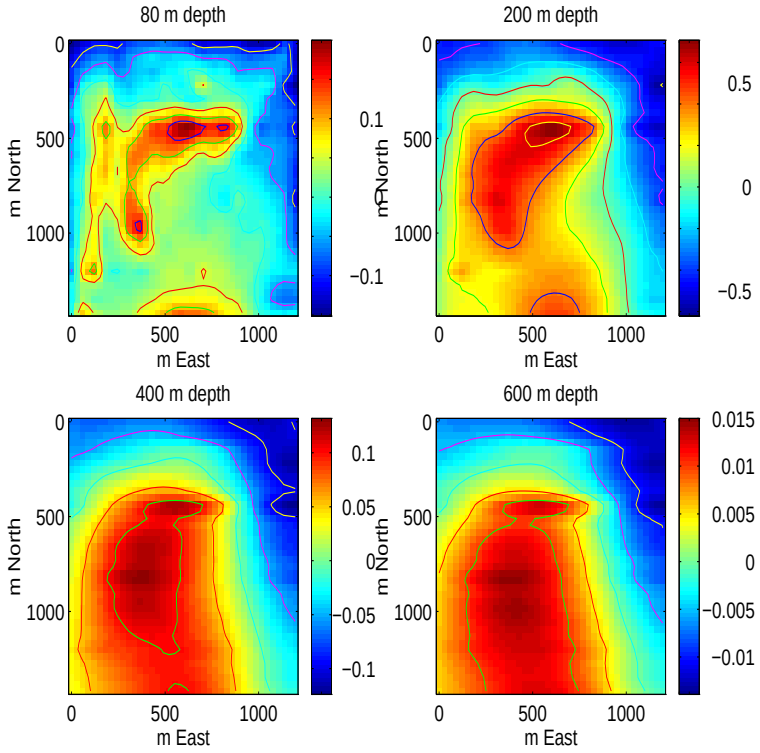

Figure 6.21: 3-D Gravity Inversion of the field data

6.2 The Tomography Problem

A very different type of problem arises in tomography where the kernels describe rays and the matrix is sparse rather than full. In this section we test our algorithms on different problems which arise in tomography and use the different methods on two field data sets.

6.2.1 Borehole Tomography

The first geometry we deal with is borehole tomography. In this case transmitters are put on one side of the area we want to image and receivers are put on the other. Recall from Chapter 1 that the governing equation for the tomography experiment is:

$$b(l_i) = \int_{l_i(\xi,\eta)} m(\xi,\eta) dl_i(\xi,\eta) \tag{6.15}$$

where $l(\xi,\eta)$ describes the ray path.

In order to solve the problem we discretize the integral equation into M cells. Assuming that $m(x,y)$ can be written as piecewise constant functions $\psi_i(\xi,\eta)$ in these cells, the model is:

$$m(\xi,\eta) = \sum_{j=1}^{M} x_j \psi_j(\xi,\eta) \tag{6.16}$$

and the integral is transformed into:

$$b(l_i) = \sum_{j=1}^{M} x_j \int_{l_i(\xi,\eta)} \psi_j(\xi,\eta) dl_i(\xi,\eta) \tag{6.17}$$

which gives the system of equations:

$$Ax = b$$

where:

$$A_{ij} = \int_{l_i(\xi,\eta)} \psi_j(\xi,\eta) dl_i(\xi,\eta) \tag{6.18}$$

The calculation of A_{ij} is straight-forward. It is the length of the i^{th} ray in the j^{th} cell.

In order to test our algorithms, we use radio imaging (RIM) field data. Exactly like in the three dimension gravity example, we would like to build a synthetic example with the same geometry of the field data set. We therefore use the location of the transmitters and the receivers as was given in the data set and generate the tomography matrix. We have 51 sources and 41 receivers. Not all sources interact with the receivers and we have a total number of 1153 rays which cover the area. In order to show the ray coverage, we plot the geometry of the boreholes and the vector:

$$c_j = \sum_{i=1}^{N} A_{ij}$$

The vector c represents the ray coverage of each cell since it shows the total length of rays which pass through the j^{th} cell. The vector c is plotted in Figure 6.22.

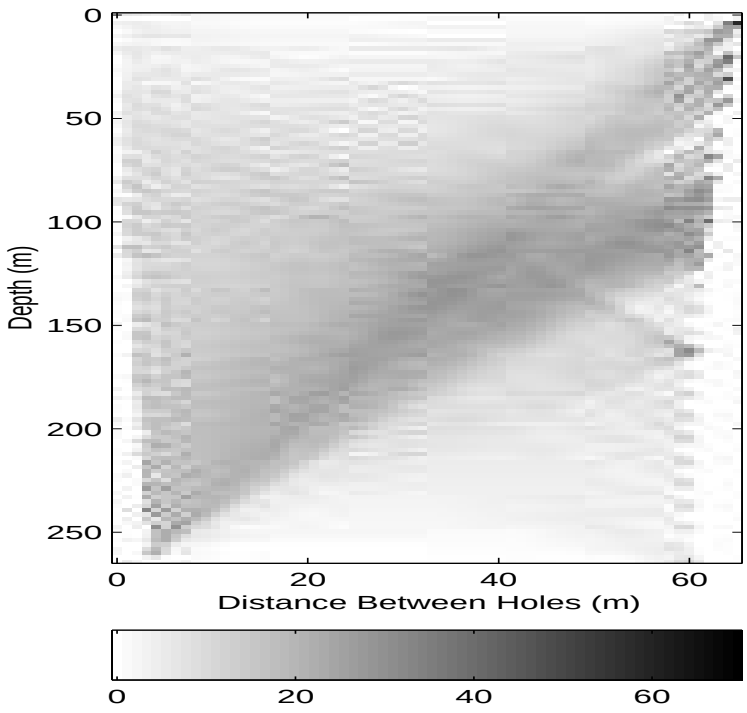

Figure 6.22: Borehole Tomography Ray Coverage. Notice that the coverage is not uniform.

The ray coverage of the model is not uniform and therefore we would like to test the ability of this experiment to predict the true model. First we carry out a synthetic experiment and pick a smooth model and generate data for it. The model is plotted in Figure 6.23 (A), and is assumed to be an absorption anomaly above a baseline of $6m^{-1}$. The data anomalies are plotted in Figure 6.24. The model is made from $64 \times 128 = 8192$

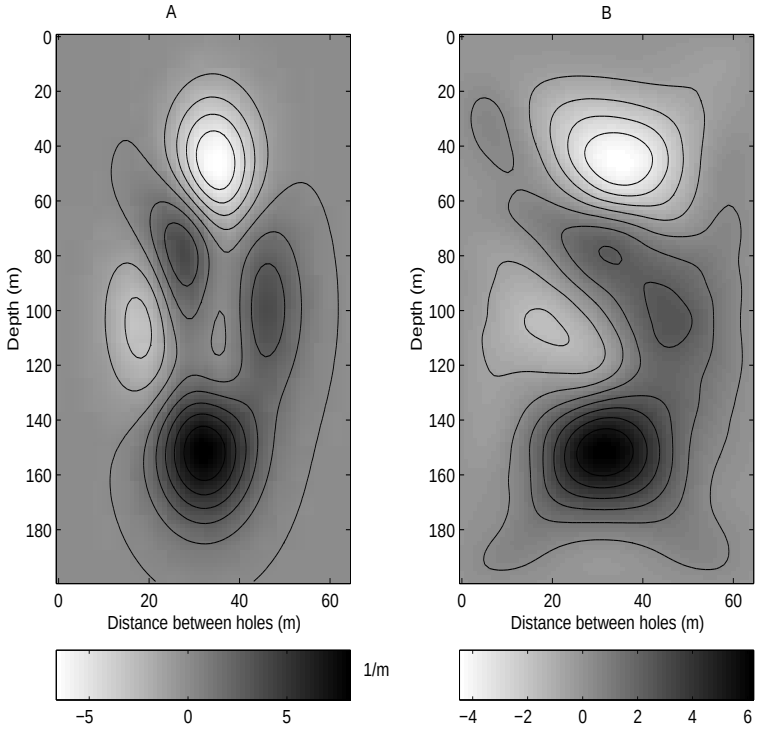

Figure 6.23: Synthetic Borehole Tomography Example. The true model is plotted in Figure A and the reconstructed is in Figure B. This reconstruction is for the 5% noise case. The units of the models are in m^{-1} above a background of $6m^{-1}$.

cells.

In order to get a smooth model we pick the weighting to be:

$$W = -\nabla^2 + \epsilon I$$

with $\epsilon = 0.01$ and the ∇^2 operator does not contain boundary conditions. In the next stage we add $5, 10$ and 20% noise to the data and invert the system using CGLS, iterated

Method	$\|\|Ax - b\|\|$	ϕ_m	flops
CGLS+GCV	161.4	2.53	1.0E8
CGLS+L	161.4	2.53	1.3E8
LSQR+GCV	161.6	2.53	1.2E8
LSQR+L	161.6	2.53	1.6E8
ML+G	168.2	2.50	5.9E8
H-LSQR	163.2	2.52	3.1E8
I-KRY	169.7	2.49	4.1E8
I-GRAD	170.3	2.49	4.6E8

Table 6.15: Comparison between all methods for 5% noise. True square root of the misfit is 164.2.

Krylov methods, iterated gradients and multilevel methods. A comparison between the methods is given in Tables 6.15-17.

From the tables we conclude once again that the combination of conjugate gradient

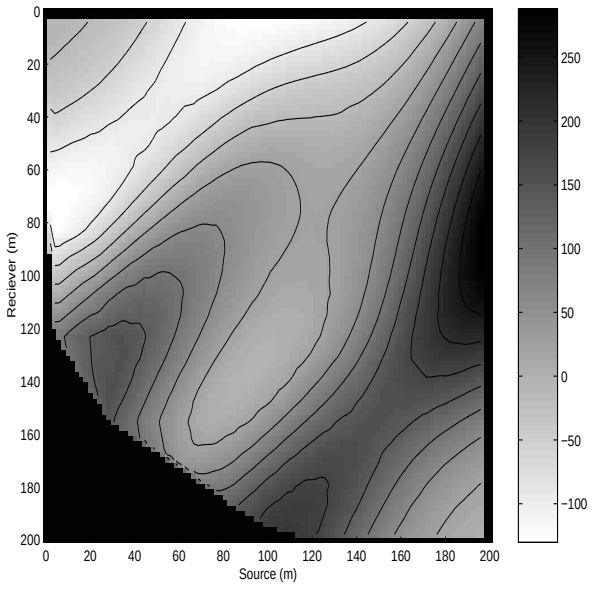

Figure 6.24: Synthetic Borehole Tomography Data. The data is plotted as a function of the z position of the transmitter and the receiver. Notice that the data have no physical dimensions.

Method	$Ax - b\|\|$	ϕ_m	flops
CGLS+GCV	323.3	2.42	8.1E7
CGLS+L	326.2	2.41	1.1E8
LSQR+GCV	321.7	2.40	1.0E8
LSQR+L	321.7	2.40	1.3E8
ML+G	331.2	2.36	9.2E8
H-LSQR	322.7	2.40	2.8E8
I-KRY	330.1	2.36	3.9E8
I-GRAD	329.5	2.37	4.1E8

Table 6.16: Comparison between all methods for 10% noise. True square root of the misfit is 324.2.

Method	$\|\|Ax - b\|\|$	ϕ_m	flops
CGLS+GCV	638.1	2.52	8.1E7
CGLS+L	638.1	2.52	1.2E8
LSQR+GCV	638.2	2.52	9.8E8
LSQR+L	638.2	2.52	1.3E8
ML+G	644.7	2.49	8.9E8
H-LSQR	640.1	2.52	2.8E8
I-KRY	647.9	2.49	3.8E8
I-GRAD	649.3	2.47	4.0E8

Table 6.17: Comparison between all methods for 20% noise. True square root of the misfit is 642.1.

methods and the GCV is the most efficient method for the solution of the borehole tomography problem. The result of the inversion for the 5% case is shown in Figure 6.23 (B). When trying to understand why these methods work as well as they do we look again at the singular value distribution of the problem. The problem is very large and therefore straight-forward calculation of the SVD is not practical. In order to calculate an approximation to the spectrum of the problem we use a coarser discretization and carry out the SVD of the same system which is discretized on $16 \times 32 = 512$ cells. The SVD of this system is plotted in Figure 6.25. Again the singular values of the system

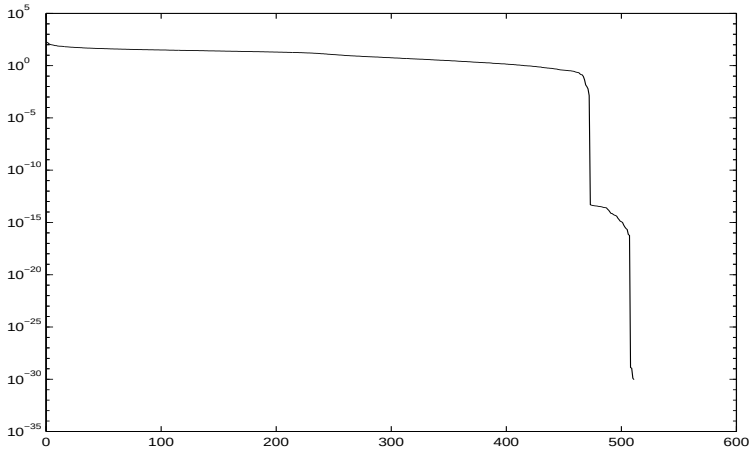

Figure 6.25: The singular values of the tomography system.

are decaying in steps and therefore the conjugate gradient algorithm is expected to work well when applied to this problem.

After testing this problem with the synthetic data set we use a field data set, using the same discretization and the same weighting matrix. The data set is plotted in Figure 6.26. Notice that the real data do not have a zero mean and therefore we would like to find a simple reference model which can decrease the misfit by a large amount. The most simple choice is a half space and we would try to find the "best half space" which fits the data. The half space can be described as a vector of constant absorption αe, where α is

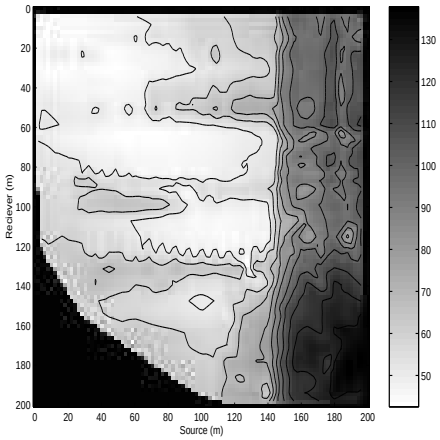

Figure 6.26: The field data.

a real number of the unknown absorption and $e = [1, 1, ...1]^T$. This leads to the following minimization problem:

$$minimize \quad ||\alpha Ae - b||^2 \tag{6.19}$$

The optimal α is given by:

$$\alpha = \frac{b^T Ae}{||Ae||^2} \tag{6.20}$$

Using the data and the tomography matrix we evaluate the background α to be $6.34\ m^{-1}$. We now generate the residual $r = b - \alpha Ae$ and use this residual as the data for this problem. The result of the inversion of these data is plotted in Figure 6.27. The inversion is performed using the CGLS and GCV. The final estimated noise on this problem is 37% which explains why other studies of the same data set where not very successful (Haber and Oldenburg [1996], McGaughey [1994]), where the noise level was estimated to be $10 - 20\%$. This field data set example shows the importance of noise estimation techniques.

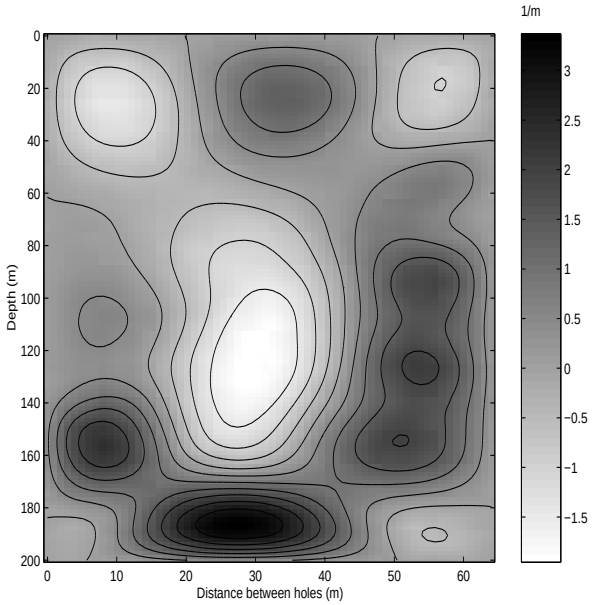

Figure 6.27: The result of the field data inversion.

6.2.2 Medical Tomography - SPECT

Another very common type of tomography is the Single-Photon-Emission-Computed-Tomography (SPECT). This method is used on a daily basis in almost every large hospital. A common way to reconstruct SPECT images is by filtered back-projection (FBP). The advantage of this reconstruction is its computational time which is very low, however this technique does not take the non-uniqueness of the problem into consideration. The technique does not allow incorporating additional *a priori* information and as a result the images are often distorted. Recently Smith *et al.* [1992] suggested to use the TSVD for the reconstruction and suggested the discrepancy principle as a stopping criterion. Ros *et-al.* [1996] looked at the role of the regularization parameter, however it is commonly believed that SPECT images are unique and therefore no considerable effort was made to try and choose model objective functions. Noise estimation techniques are rarely used rigorously and to my knowledge the GCV principle was never applied to SPECT reconstruction. In this thesis we show that by a simple choice of weighting matrix and noise estimation techniques we can improve the image with only a little sacrifice in computational time. It is also important to understand that the solution of SPECT reconstruction is not unique. The ability to define a new objective function and obtain different models which fit the data to the same extent, is an important step in this direction. In this section we test our algorithms on a real data set which was obtained from Vancouver General Hospital.

Recall from Chapter 1, that the data are obtained by putting an array of bins in a plane which is rotated around the object to be reconstructed. Each plane in the rotation is a projection of the image to that plane. The data are then plotted as a collection of these planes (Figure 6.28). A filtered back projection inversion is plotted in Figure 6.29. This reconstruction is definitely not accurate since it does not take into account

the fact that there are areas which are outside the patient's body. We now show that with very few assumptions the image can be improved. We choose a weighting matrix, W, which incorporates some of our *a priori* information about the problem. A similar choice was used by Haber *et al.* [1996]. Our goal in choosing the weighting matrix is to automatically specify a support region for the source activity (i.e. find the active region) and to impose smoothness. We therefore choose our weighting by noting first that there are many data which are zero. This means that the rays which generated these data probably did not pass through any active area (since there is no negative activity). We define such a ray as a null ray. Our goal is to locate pixels which the null rays passed through so we can discriminate against having some activity in these pixels. Let I_z be the set of indices of all null rays. We sum the rows of the null rays together:

$$r_{z(j)} = \sum_{i \in I_z} A_{ij} \tag{6.21}$$

where A is the tomography system matrix, and each row of A represents a ray. Each

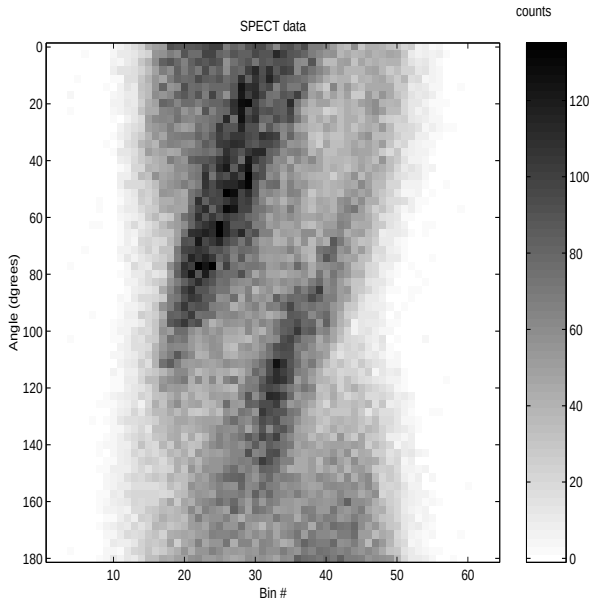

Figure 6.28: SPECT data. The bin number is plotted in the x direction and the plane angle is plotted in the y direction. The projections were collected from $0 - 180^o$.

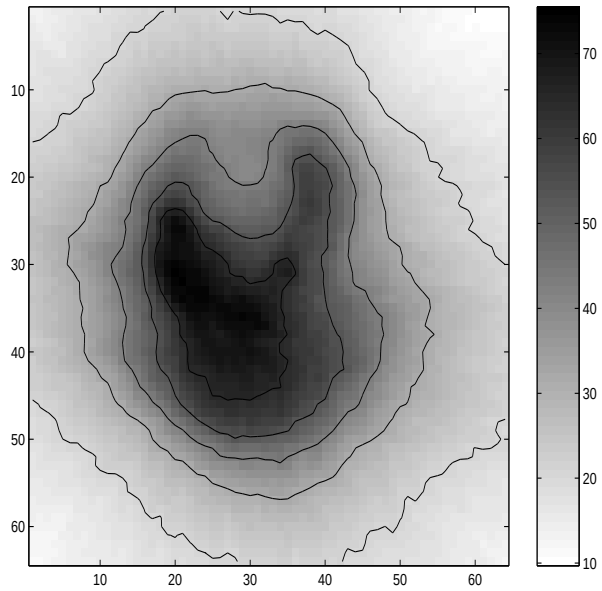

Figure 6.29: Filtered back projection reconstruction of the data.

element of the vector r_z is the area of all the null rays in a particular pixel. If this number is large, then many null rays pass through this pixel. However if this number is 0, no null rays pass through this specific pixel and we can therefore assume that the probability of having activity in the pixel is high. We normalize this result by dividing by the total area of rays which pass through the pixel, and hence the final weighting is:

$$w_j = \frac{\sum_{i \in I_z} A_{ij}}{\sum_{i=1}^{N} A_{ij}} \tag{6.22}$$

The result of such weighting is plotted in Figure 6.30. Using this simple weighting we invert the data and use hybrid LSQR and GCV as a stopping criterion. The result is plotted in Figure 6.31. The improvement in the inverted image is obvious. Other regularization such as smallness, smoothness and incorporating the anatomy as *a priori* information is possible using this strategy. This will result in improved images.

So far we have demonstrated that SPECT images can be improved significantly by adding simple *a priori* information and noise estimation techniques. We now test the

different methods to achieve this goal. As done before, first we plot the spectrum of the problem. Again since we cannot calculate the SVD of the SPECT system we discretize it on a coarser grid of 32×32 cells and carry the SVD on this system. The spectrum of the problem decays very slowly and therefore if the noise level is high, Krylov subspace methods may fail. In Table 6.18 we compare the misfit which is obtained by different techniques, the model norm and the number of flops. In this test Krylov subspace methods did not perform well and the noise estimation is too low. This is obvious when comparing the images (Figure 6.33). The Krylov subspace images are noisy while other methods give smooth images. When comparing the methods developed here to the FBP, we see that these methods are about twice as computationally intensive as the FBP, however the images are significantly improved and while the noise level is not predicted by using the FBP, it could be predicted using hybrid methods. Since the computational time is short (about 120 seconds per inversion on a SPARC 10 work station), using inversion techniques should not be a problem on a daily basis.

As a summary to this section we note that:

- Incorporating *a priori* information to SPECT images improves the recovered image.

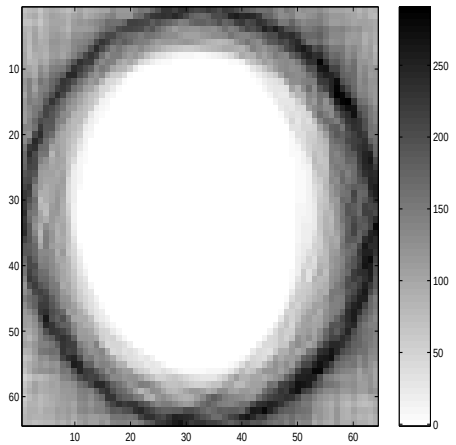

Figure 6.30: The special weighting of SPECT image.

Method	$\|\|b - Ax\|\|/\|\|b\|\|$	ϕ_m	flops
CGLS+GCV	0.17	115	4.2E7
CGLS+L	0.21	107	5.3E7
LSQR+GCV	0.17	115	4.9E7
LSQR+L	0.21	107	5.8E7
ML+GCV	0.28	68	23E7
H-LSQR	0.29	71	6.3E7
I-KRY	0.26	70	14E7
I-GRAD	0.27	72	13E7
FBP	0.35	33	3.2E7

Table 6.18: Comparison between methods for SPECT inversion. Note that subspace methods underestimate the noise level.

- The spectrum of the SPECT tomography matrix decays slowly and the noise is relatively high, therefore Krylov subspace methods tend to fail when applied to SPECT inversion.

- Hybrid methods and subspaces which are not data dependent such as multilevel and gradients, perform well when applied to SPECT.

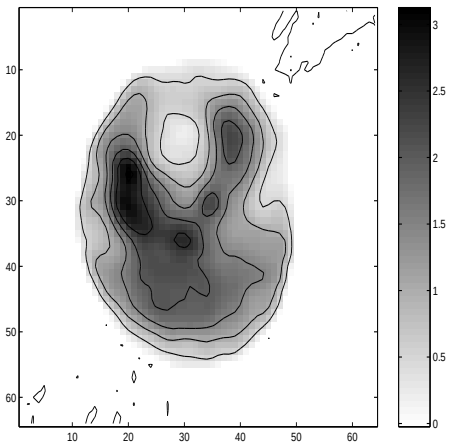

Figure 6.31: Inversion of SPECT using the weighting in Figure 6.30.

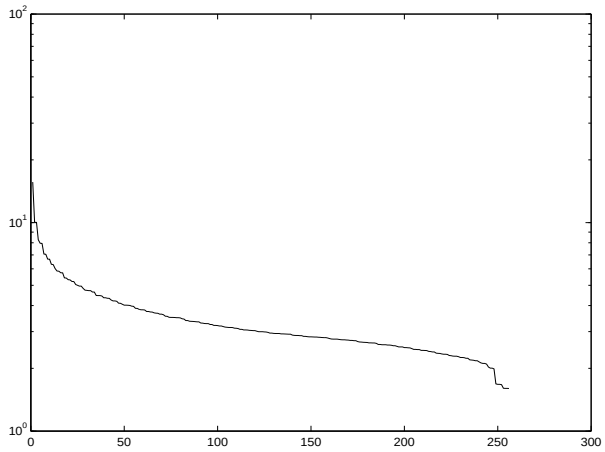

Figure 6.32: The SPECT spectrum.

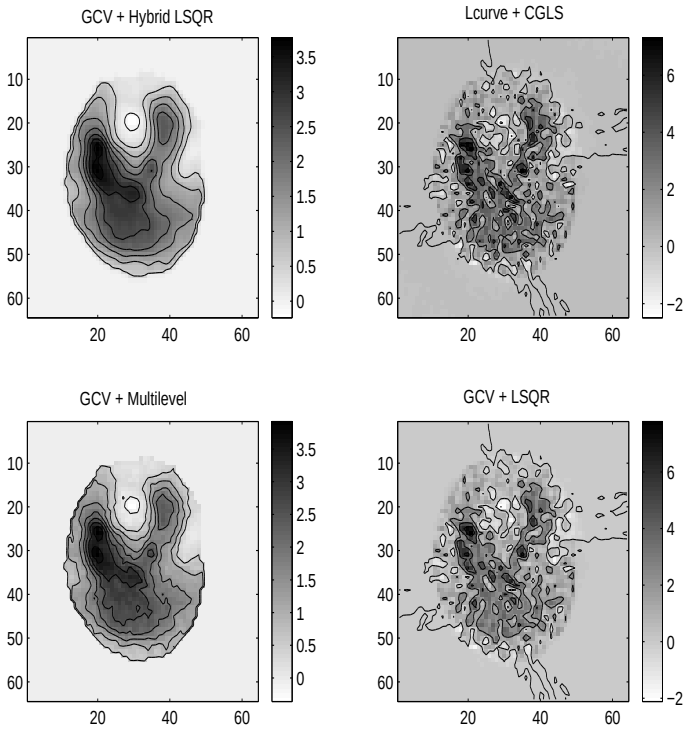

Figure 6.33: Inversion of SPECT data with different techniques. Notice that the CGLS+GCV and GCV+L-curve fail to predict noise levels. Hybrid methods on the other hand give reasonable inverted models.

6.3 Summary

In this chapter we tested the techniques which were developed in previous chapters on a wide range of problems. The problems can be characterized by three main features: their size, the distribution of their singular values, the noise levels and types. A one-dimensional problem gives rise to small systems which can be easily handled and therefore we used the 1-D gravity data to compare noise estimation methods. We have found that for this type of problem, which is characterized by a fast decaying singular values, all methods work well, however Krylov space methods are the most efficient. The nonlinear gravity spectrum behaves in a similar way, and therefore it is not surprising that the same methods work well for these problems as well. The second type of spectrum was characterized by steps. The two and three dimension gravity and the borehole tomography have this type of spectrum. Almost all methods work well on this type of spectrum and Krylov space methods are the most robust in terms of computations. A comparison of the relative amount of computation for the different techniques is in Figure 6.34. The

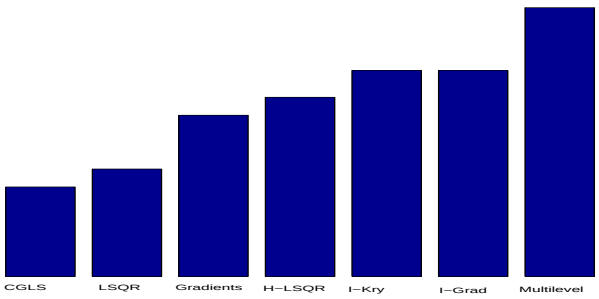

Figure 6.34: A comparison between the relative number of flops for the solution of a linear problem with different methods.

advantage of Krylov space methods over any other technique is substantial and therefore unless there is no other possibility, I would try to use this method on problems which possess fast or steps decaying spectrum.

A different type of problem is SPECT tomography. In this case the spectrum of the system decays slowly and the noise level is very high. In this case Krylov space methods tend to underestimate the noise level and converge to a noisy solution. The reason is that oscillatory vectors may converge prior to the smooth vectors. In this case either subspace techniques which do not depend on the right hand side such as gradients and multilevel methods, or hybrid LSQR iterated gradients, or iterated Krylov are the solution for this type of problem. If storage is not a problem, then hybrid LSQR tends to be the most efficient method. However if storage is a problem, then iterated Krylov methods or iterated gradients are the optimal solution to the problem.

When facing a new linear inverse problem I suggest the following guide for the choice of a method:

- Estimate the spectrum of the problem. If the spectrum has steps in it or it decays quickly, use Krylov space methods.
- If using Krylov space methods, estimate the noise level. If the noise is very low (lower than 1%), use the L-curve as a stopping criterion, otherwise use the GCV.
- If the spectrum decays slowly use a hybrid method.
- If there is no storage problem, use the hybrid LSQR combined with the GCV.
- If using hybrid methods and storage is a problem, use iterated gradients or iterated Krylov methods.

Chapter 7

Nonlinear Inverse Problems

This chapter presents the formulation and general ideas of nonlinear inverse theory. It is the equivalent of Chapters 2 and 3. In the first section a description of two common methodologies for regularizing nonlinear ill-posed problems is presented. We then explain why one of these methodologies is faulty in nature.

Solving nonlinear ill-posed problems leads to nonlinear optimization problems. In the second section a review of the two main strategies for carrying out an optimization problem, damped Gauss-Newton and trust regions, is presented. Finally, a review of some of the commonly used techniques for the solution of nonlinear ill-posed problems and a discussion of some of the difficulties using these existing methods is presented. A simple example is given to demonstrate how one of these methods can fail.

7.1 Formulation of Nonlinear Ill-Posed Problems

Let $x \in \mathcal{H}$ be the model, $b \in R^N$ the data and $\epsilon \in R^N$ be the noise. First we assume a general connection between the model and the data:

$$b_j = F_j[x] + \epsilon_j \qquad j = 1...N \tag{7.1}$$

This formulation is very general and therefore not a lot can be said about methods for solution. We therefore look closer into the source of geophysical inverse problems. Most geophysical inverse problems come from a physical description of the world. Usually this

description can be presented in the form of a differential equation:

$$\mathcal{L}(x,u) = f \tag{7.2}$$

where $\mathcal{L}$ is a linear differential operator, u is some field, x is the model and f is the source. For example the seismic velocity model is related to the wave field by the wave equation, the conductivity structure of the earth is related to the electric field through Maxwell's equations. A common practice to obtain a relation between the model, x, and the data which is the measured field, is to use a Green's function (see for example Tichonov [1963], [1977], Devaney [1989], Chow [1990], Li [1992]). First the field u is decomposed into a primary field u_0 which is due to the background (and is therefore known) and an unknown secondary field, u_1. The problem 7.2 is transformed into an integral equation of the form:

$$u_1(r) = u_0(r) + \int_v \mathcal{G}(x(\rho), r; \rho) u_1(\rho) d\rho = u_0 + G(x)u_1 \tag{7.3}$$

where $\mathcal{G}$ is the Green function and G represents the linear operator which depends on x. This equation is a linear Fredholm integral equation of the second kind for u_1 and the solution for u_1 can be written as:

$$u_1(r) = (I - G(x))^{-1} u_0 \tag{7.4}$$

This is a nonlinear relation between the model x and the secondary field u_1. Since the data are just the field values measured at some places $r_j \ \ , j = 1...N$, this gives a relation between the model and the data. In order to understand the general behaviour of such an expression we use the Neumann series expansion:

$$(I - G(x))^{-1} = I + G(x) + G(x)^2 + \quad ... \tag{7.5}$$

The series is nonlinear for the model x. However we see that as a first-order approximation (which is known as the Born approximation), the relation between the model and the data

can be represented by a linear Fredholm integral equation of the first kind for x. If we take more terms in the series then the equation is a nonlinear Fredholm integral equation of the first kind. In this thesis we assume that the inverse problem can be transfered into a form such that the relation between the model and the data can be represented by a nonlinear integral equation. If this is the case then all the characteristics of linear integral equation of the first kind and the methods we have developed in the previous chapters can be applied to the linearized equations. In order to be able to work with linearized problems we make the important assumption that the operator F is twice Fréchet differentiable.

Since the operator F is general, we avoid the question of how the forward modelling should be computed. It is possible to carry out the forward modeling through the differential or the integral relation and it is usually problem dependent. We stick to the guiding principle which we had in Chapter 2 which is to discretize the model, x ($x \in \mathcal{H}$), with M unknowns such that $M > N$ where N is the number of data.

The first question that can be asked is about the existence and uniqueness of the solution. For a general operator not a lot can be said. However for nonlinear Fredholm integral equations of the first and second kind, Jerri [1985] proved the existence of the solution under some conditions. We assume that the solution exists for all of our problems and also assume that since we have only N data and M unknowns with $N < M$, that the solution is non-unique and that the operator is ill-posed, i.e. there might be an infinite number of models x which fit the same data and for a small perturbation in the data we get a large perturbation in the model. Since we assume non-uniqueness and ill-posedness, regularization is needed. The different ways to deal with nonlinear inverse problems are revealed by the methods of regularization. In the next section we discuss two strategies to deal with nonlinear ill-posed problems.

7.2 Formulation of the Solution

In this section we review the two main strategies for nonlinear inverse problems. In order to solve a nonlinear problem it is common to use linearization and iteration. Other strategies such as genetic algorithms and simulated annealing are also possible. However these methods are very computationally intensive and therefore we do not deal with them in this thesis.

7.2.1 Creeping Versus Leaping

Traditionally, there has been two basic approaches to tackle nonlinear inverse problems. In both cases we want to find a model x^* which fits the data to some extent. This leads first to the solution of the equation:

$$0 = ||F[x] - b||^2 - T \tag{7.6}$$

where T is some tolerance level. This problem is not linear and therefore we choose an initial model x and linearize equation 7.6 with respect to this model by using the Taylor expansion for F:

$$F[x + \delta x] = F[x] + J(x)\delta x + R(x, \delta x) \tag{7.7}$$

where:

$$J(x) = \frac{\partial F}{\partial x} \tag{7.8}$$

is the Fréchet derivative and $R(x, \delta x)$ is the residual. The linearization leads to the equation:

$$b - F[x] - R(x, \delta x) = J(x)\delta x \tag{7.9}$$

Quadratic approximation for the function F is also possible in principle:

$$F[x + \delta x] \approx F[x] + J(x)\delta x + \frac{1}{2}\delta x^T H(x)\delta x \tag{7.10}$$

where

$$H(x) = \frac{\partial}{\partial x} J(x)$$

While a quadratic approximation is obviously better than a linear one, the second Fréchet derivatives H are not usually calculated since that involves an inordinate amount of calculation even for a small problem. We therefore stick to the formulation of calculating only the first Fréchet derivatives.

The linearized formulation is local in nature, and if we are close enough to the minimum we can neglect the residual, $R(x, \delta x)$, which is of $\mathcal{O}(\delta x^2)$. This leads to a linear equation for δx:

$$b - F[x] = J(x)\delta x \tag{7.11}$$

While the non-uniqueness is not observed at first sight when working with the nonlinear operator F, the Fréchet derivative $J(x)$ is underdetermined (and therefore not invertible) since there are M model parameters and only $N < M$ data points. Furthermore, since F is usually a nonlinear integral equation, the linearization can be represented as a linear Fredholm equation of the first kind and therefore, the Fréchet derivative typically has a large condition number. In order to solve equation 7.11 we have to add regularization to the problem. As in the linear problem, different regularizations yield different types of solutions. The first way to impose regularization is a naive regularization. Noticing that we have to solve a linear inverse problem for δx, we could use the same methods we used for the linear case and transform this linearized ill-posed problem into an optimization problem:

$$minimize \quad \beta||W\delta x||^2 + ||b - F[x] - J(x)\delta x||^2 \tag{7.12}$$

Such methods are often refered to as creeping since they obtain a small perturbation δx. Hanke [1997 a,b] suggested using this type of regularization and proved that these methods converge to a solution x^* of the nonlinear inverse problem. The main problem

with using these kind of methods is that while a solution with the desired misfit is obtained, we do not have any control on the characteristics of the solution. The solution x^* depends on the starting point x_0 and on the path of minimization. The final solution can be written as:

$$x^* = x_0 + \sum_{k=1}^{P} \delta x_k \tag{7.13}$$

While every δx_k is small, the final result x^* does not have to be small. Parker [1994] has shown that this type of method might converge to an unacceptable model. Finally, the most important aspect of this solution, is consistency. There are many ways to calculate the updates δx_k. One could use Krylov space methods or one could use a full-space method. The result would be that the end model, x^*, even if it is reasonable, is different even if the same kind of regularization operator W is used, and thus the final results of the same problem using two optimization techniques may not be similar. Interpretation of such results is somewhat meaningless because the features in the final model might not be there because of *a priori* information but as a result of the minimization algorithm. This type of algorithm does not take into consideration the non-uniqueness of the original nonlinear inverse problem and therefore although commonly used, is faulty by its nature.

A different methodology was suggested by Oldenburg [1983], who altered the formulation such that a global objective function of the model could be minimized. This formulation was used by Constable *et al.* [1987] to find the "simplest model". This is often referred as "Occam's method", named after the English philosopher who claimed that "it is vain to do with more what can be done with fewer". The interpretation of this philosophy to the inverse problem is that the model which solves to a problem should be as "simple" as possible. In order to obtain such a solution the regularization is not imposed on the perturbation δx but on the model x, i.e. at iteration $k+1$ we solve the

problem:

$$minimize \quad \beta||W(x_k + \delta x)||^2 + ||b - F[x_k] - J(x_k)\delta x||^2 \tag{7.14}$$

In this way the regularization takes into account the non-uniqueness of the nonlinear problem. This method is often referred to as "leaping" since the step size δx might not be small. Since the regularization is global, this problem can also be written as the linearization of:

$$minimize \quad \phi = \beta||Wx||^2 + ||F[x] - b||^2 \tag{7.15}$$

which is a nonlinear optimization problem. The problem is consistent, and if the misfit term is convex and W is not singular, it possesses one global minimum (Luenberger [1969]). In this formulation the solution does not depend on the starting point x_0 (assuming convexity) and on the nonlinear optimization process. In this thesis we shall study this optimization problem further.

7.2.2 Penalty Formulation Versus Lagrangian Formulation

The minimization problem given by 7.15 is an unconstrained optimization problem. A similar, yet different, problem can be derived from the point of view of constrained optimization. We review these considerations and observe the differences between the problem as formulated in 7.15 and the constrained formulation.

Assume that we have a target misfit T and we actually know that the noise level of our problem can be represented by this target misfit. We could then formulate the inverse problem as follows (Parker [1994]):

$$minimize \quad ||Wx||^2 \tag{7.16}$$

$$subject\ to \quad ||F[x] - b||^2 = T$$

The next step is to form a Lagrangian:

$$\mathcal{L}(x,\beta) = ||Wx||^2 + \beta^{-1}(||F[x]-b||^2 - T) \tag{7.17}$$

The Lagrangian is then minimized with respect to x and we want to find a saddle point at β^{-1}. When working with the linear problem, this formulation is equivalent to the penalty formulation expressed in 7.15. However for nonlinear methods the Lagrangian formulation is somewhat different. First we note that the domain $||F[x]-b||^2 = T$ is not convex. This is easily shown since if

$$||F[x_1]-b||^2 = ||F[x_2]-b||^2 = T$$

then usually:

$$||F[x_1+\gamma(x_2-x_1)]-b||^2 \neq T \qquad 0<\gamma<1$$

This is obvious for the case where F is linear. Therefore if we are on the constraint, i.e. we have reached a point x_c such that $||F[x_c]-b||^2 = T$, we might not be able to move from this point to the minimum point of the Lagrangian while staying on the constraint. The second reason that this methodology does not fit most data sets is that the tolerance level T is usually unknown *a priori*. We therefore stick to the penalty formulation of trying to find the unconstrained minimum of the quadratic expression 7.15 exactly as in the linear case.

7.3 Methods for Nonlinear Optimization

After a global objective function as in 7.15 is chosen, we are facing a well-posed nonlinear optimization problem, with an unknown regularization parameter β. In this section we assume β is known and review methods for the solution of the nonlinear problem 7.15. We review the work of Gauss [1801], Marquardt [1963], Armijo [1966], Cost [1983], Dennis

et al. [1988], Dembo and Steihaug [1983], Eisenstat and Walker [1994], Fraley [1989], Elster and Neumaier [1997], Gulliksson *et al.* [1997] and Knoth [1996], and examine two types of nonlinear optimization algorithms, the first is the Damped Gauss-Newton (DGN) iteration and the second is a trust region (TR) algorithm.

7.3.1 Damped Gauss-Newton Method

The DGN methods start with the differentiation of 7.15 with respect to x and setting the result equal to zero. This gives:

$$\frac{\partial \phi}{\partial x} = g(x) = \beta W^T W x + J(x)^T (F[x] - b) = 0 \tag{7.18}$$

where $g(x)$ is the gradient of 7.15. If we find x^* which solves 7.18, then we have found the desired solution. The main problem is that 7.18 is a nonlinear equation, which already involves the derivative of F with respect to x. In order to avoid the calculation of the second derivative of F to x, this equation is linearized by:

$$\beta W^T W(x + \delta x) + J(x)^T (F[x] + J(x)\delta x - b) = 0 \tag{7.19}$$

This gives a linear system of equations for δx. This process is repeated and our goal in the k^{th} iteration is to find the perturbation δx which solves the linearized equation 7.19. Rearranging the terms in the equation gives:

$$(J(x_k)^T J(x_k) + \beta W^T W)\delta x = J(x_k)^T (b - F[x_k]) - \beta W^T W x_k \tag{7.20}$$

This problem is identical to the least-squares problem:

$$\begin{bmatrix} J(x_k) \\ \sqrt{\beta} W \end{bmatrix} \delta x = \begin{bmatrix} b - F[x_k] \\ -\sqrt{\beta} W x_k \end{bmatrix} \tag{7.21}$$

Writing $x_{k+1} = x_k + \delta x$ equation 7.19 is identical to:

$$(J(x_k)^T J(x_k) + \beta W^T W) x_{k+1} = J(x_k)^T (b - F[x_k] + J(x_k) x_k) \tag{7.22}$$

This problem is identical to the least-squares problem:

$$\begin{bmatrix} J(x_k) \\ \sqrt{\beta}W \end{bmatrix} x_{k+1} = \begin{bmatrix} b - F[x_k] + J(x_k)x_k \\ 0 \end{bmatrix} \tag{7.23}$$

Now, we can solve for either the perturbation δx or for the model x_{k+1} and go to the next step. If the steps δx are small, then this approach would work, however in most geophysical problems we do not start very close to the solution and therefore the linearization process which involves neglecting the residual $R(x_k, \delta x)$ from equation 7.9 might not be justified. Recall that $R(x_k, \delta x)$ is of $\mathcal{O}(\delta x^2)$ and therefore when the step size is large the minimization of the linearized equation might cause an increase in the objective function. Such a step is obviously a bad choice because it does not take us closer to the minimum of the nonlinear problem. This defect can easily be corrected. The Gauss-Newton step is a descent direction (Dennis and Schnabel [1996]) and therefore if the step size is small enough, the nonlinear function will have a similar behaviour to the linearized problem. Based on this observation, Armijo [1966] suggested trying a step size of $\delta x \leftarrow \omega \delta x$ where $0.1 < \omega < 0.5$ and to repeat the process. Finally, for small enough step, the nonlinear function decreases and we get closer to the minimum of the nonlinear function. This algorithm can be written as follows:

Damped Gauss-Newton Minimization

- Choose a starting model x_0 and regularization parameter β
- Calculate $\phi = \beta||Wx_0||^2 + ||b - F[x_0]||^2$
- Set $\phi^{old} = \phi$
- Until convergence do
  - Linearize 7.15: Calculate the Fréchet derivative $J(x_k)$ and the gradient $g(x_k)$

- Check for convergence (see discussion next).
- Solve 7.20 and obtain δx
- Calculate $\phi^{new} = \beta||W(x_k + \delta x)||^2 + ||b - F[x_k + \delta x]||^2$
- If $\phi^{new} < \phi^{old}$ accept the step and set: $x_{k+1} = x_k + \delta x$, $\phi^{old} = \phi^{new}$
- Elseif $\phi^{new} \geq \phi^{old}$ reject the step and set $\delta x \rightarrow \omega \delta x$ where ω insures that the function decreases.

There are a few important details when carrying out the DGN method. The first is checking for convergence. Our final goal is to find a model x^* which solves the nonlinear equation for the gradient of ϕ, 7.18. It is therefore a good idea to check how well this equation is solved. A very simple convergence criterion is to demand that $||g(x)|| < \delta$ where δ is some small number. This demand is usually not satisfactory since scaling is a factor. If we minimize $\alpha\phi$ instead of ϕ then the gradient is multiplied by the factor α. For this reason it has been suggested by Dennis and Schnabel [1996] to replace this criterion by:

$$\frac{|g(x)|^T|x|}{\phi(x)} < \delta \tag{7.24}$$

or

$$\frac{||g(x)||}{\phi(x)} < \delta \tag{7.25}$$

where $|g(x)|$ is the absolute value of the elements of the vector $g(x)$. Another important point in the DGN algorithm is the step size reduction factor, ω. The choice of $0.1 < \omega < 0.5$ is somewhat arbitrary and more efficient line search methods can be used in order to find the minimum of the function 7.15 in the direction δx. For a collection of such methods see Luenberger [1969].

7.3.2 Trust Regions

Another very popular method to deal with globally convergence optimization is to use a trust region (TR). Again we are faced with the solution of the nonlinear equation 7.18 where we might take steps that are too large and the Taylor expansion might not hold. The general idea of the trust region strategy is not only to limit the step size but also to obtain a new descent direction. Assume that we know that the linearized problem 7.19 and the nonlinear problem 7.18 are equivalent in a region of size η. We could then look for the best perturbation δx in this region by introducing a new optimization problem:

$$minimize \quad \beta||W(x+\delta x)||^2 + ||F[x] + J(x)\delta x - b||^2 \tag{7.26}$$

$$subject\ to \quad ||\delta x||^2 \leq \eta$$

The radius η is known as the size of the trust region and its size is discussed later in this section. Using the penalty formulation, this constrained problem leads to the minimization of:

$$\phi_{TR} = \beta||W(x+\delta x)||^2 + ||F[x] + J(x)\delta x - b||^2 + \mu(\eta)||\delta x||^2 \tag{7.27}$$

where $\mu(\eta)$ is another penalty parameter which depends on the trust region size. This leads to the linear system:

$$(\beta W^T W + \mu I + J(x)^T J(x))\delta x = J(x)^T(b - F[x]) - \beta W^T W x \tag{7.28}$$

or, in the least-squares formulation:

$$\begin{bmatrix} J(x) \\ \sqrt{\beta}W \\ \sqrt{\mu}I \end{bmatrix} \delta x = \begin{bmatrix} b - F[x] \\ -\sqrt{\beta}Wx \\ 0 \end{bmatrix} \tag{7.29}$$

After this equation is solved and δx is found we check that the new update actually decreases the function ϕ. If this is the case then we update the model and go to the next

step, however if the function does not decrease, then we have to decrease the trust region size. Before we proceed and discuss how to determine the trust region size, we need to have a method to find a perturbation δx such that $||\delta x||^2 = \eta$.

Calculating The Trust Region Step

The problem of finding a perturbation δx with a small norm is akin to the linear problems that were dealt with in Chapters 3, 4 and 5. There, a small model which minimizes $||x||^2$ such that $||Ax - b||^2 = T$ had to be found. Here a model which minimizes 7.26 with $||\delta x||^2 = \eta$ has to be obtained. We therefore use three of the methods outlined in those chapters. The first method is a full space methodology and it is a variation of the algorithm suggested by Dennis and Schnabel [1996]. The second is a subspace algorithm suggested in Haber and Oldenburg [1996], and the third is a hybrid algorithm based on a method of Golub and Von Matt [1991].

In the full space method we solve equation 7.28 or 7.29 for different μ's. For each μ we calculate $||\delta x||$, so effectively we need to solve the nonlinear equation for μ:

$$||\delta x||^2 = ||(\beta W^T W + \mu(\eta) I + J(x)^T J(x))^{-1}[J(x)^T(b - F[x]) - \beta W^T W x]||^2 = \eta \quad (7.30)$$

This equation can be solved using a variety of methods. In this work we use a Newton-secant method for the solution of this nonlinear one-dimensional equation.

While solving for μ directly is possible when the problem is small, it is not practical for large-scale problems. As discussed in the chapters on linear inversion, the inversion of the matrix in 7.28 takes at least $\mathcal{O}(M^3)$ operations, and therefore it is desirable to find a cheap variation of this process. We recall from Chapter 4 that Krylov space methods have regularization properties and that the norm of δx increases as the iterations proceed. The use of Krylov space methods is also justified since for very large μ the full space

solution behaves like:

$$\delta x \approx \frac{1}{\mu}(J(x)^T(b - F[x]) - \beta W^T W x)$$

which is the gradient direction. For very small μ the solution behaves like the full Newton step. In CGLS the first direction is the gradient direction and it is therefore similar to $\mu \rightarrow \infty$. If the CGLS is carried to its completion then we have the full Newton step and it is similar to the case of $\mu \rightarrow 0$.

We therefore suggest the use of CGLS as a method to find a trust region update. We change the problem 7.26 into:

$$\begin{aligned} &minimize \quad \beta||W(x + \delta x)||^2 + ||F[x] + J(x)\delta x - b||^2 \qquad (7.31) \\ &subject\ to \quad ||\delta x||^2 = \eta \\ &and \quad \delta x \in \mathcal{K}(A(x), y(x), n) \end{aligned}$$

where:

$$A(x) = \begin{bmatrix} J(x) \\ \sqrt{\beta}W \end{bmatrix}$$

and

$$y(x) = \begin{bmatrix} b - F[x] \\ -\sqrt{\beta}Wx \end{bmatrix}$$

The implementation of such an algorithm is straight-forward. We solve the system

$$A(x)\delta x = y(x)$$

using CGLS and in each iteration check the norm of the solution. If at iteration k $||\delta x_k||^2 < \eta$, we proceed to the next iteration. At some stage we get to an iteration n such that $||\delta x_n||^2 \leq \eta$ and $||\delta x_{n+1}||^2 > \eta$. At this stage we terminate our process and we have a perturbation δx_n with a norm which is smaller than the trust region size η.

It is also possible to obtain a model such that $||\delta x_n^*||^2 = \eta$, using the same method proposed in Chapter 4 when we wanted to hit the target misfit. Assume we are at iteration n with $||\delta x_n||^2 < \eta$ and that at iteration $n+1$ we have

$$||\delta x_{n+1}||^2 = ||\delta x_n + \alpha p||^2 > \eta$$

where p is the last CGLS direction and α is its coefficient which is calculated in the CGLS process. We seek a parameter γ such that:

$$||\delta x_n^*||^2 = ||\delta x_n + \gamma\alpha p||^2 = \eta$$

This leads to the quadratic equation:

$$(\alpha^2||p||^2)\gamma^2 + 2\alpha(\delta x^T p)\gamma + (||\delta x||^2 - \eta) = 0 \tag{7.32}$$

This equation has two roots since $\alpha^2||p||^2 > 0$ and $||\delta x||^2 - \eta < 0$. We choose the root which gives rise to a smaller $||A(x)\delta x_n^* - y(x)||$ as the trust region step.

The last method which can be used for determining a step for the trust region approach is a hybrid method. Golub and Von Matt [1991] showed that using a Krylov space decomposition of the matrix $A(x)$ one could construct an upper and a lower bound of $||\delta x(\mu)||^2$. Similarly, we use the hybrid LSQR which was developed in Chapter 5 to estimate the regularization parameter μ in 7.28 subject to the restriction that the solution is in a Krylov subspace.

Determining the Trust Region Size

We now know how to take a step in the trust region method. In order to make the algorithm complete we need to determine two further things. First, we need to estimate the trust region size η and second how to change this size if it is not suitable for the current step. We use the same methodology as in Dennis and Schnabel [1996] for both of these processes.

First we set the trust region size η to be the size of the Gauss-Newton step, i.e. no regularization is used for the step size. After the Gauss-Newton direction is calculated, we check whether our nonlinear function decreases, i.e. whether:

$$\phi(x + \delta x) < \phi(x)$$

If the function decreases, then the Gauss-Newton step is satisfactory and we proceed. However, if the Gauss-Newton step is not satisfactory then we decrease the trust region by a factor of two and try again.

Now assume we are at iteration k and we have a trust region size from the previous iteration η_{k-1}. In order to calculate this iteration it would be useful to start using this η_{k-1} as an estimation for the trust region size in order to save expensive function evaluations, and therefore in iteration k we set $\eta_k = \eta_{k-1}$ as a starting trust region size. We use the same strategy we had before to decrease the trust region size if it is needed. So far the trust region size can only decrease in the course of optimization. If the iterations proceeds we might find ourselves stepping in smaller and smaller steps and not utilizing the full capacity of our Fréchet derivatives. We therefore need to add a condition for increasing the trust region size. This is done as follows, after the update δx is calculated we can estimate two functions, the nonlinear function:

$$\Delta\phi(x; \delta x) = \beta||W(x + \delta x)||^2 + ||F[x + \delta x] - b||^2 - \beta||Wx||^2 - ||F[x] - b||^2$$

and the linear function

$$\Delta\phi^{lin}(x; \delta x) = \beta||W(x + \delta x)||^2 + ||F[x] + J(x)\delta x - b||^2 - \beta||Wx||^2 - ||F[x] - b||^2$$

If the agreement of these two quantities is very good, i.e.,

$$\frac{|\Delta\phi - \Delta\phi^{lin}|}{max(\Delta\phi, \Delta\phi^{lin})} < \tau \tag{7.33}$$

we suspect we are not using the full capacity of the trust region and therefore we double the trust region size. Dennis and Schnabel [1996], suggested that $\tau = 0.1$ and we adopt this strategy.

Finally we summarize the trust region algorithm as follow:

Trust Region Algorithm For Gauss-Newton Iteration

- Choose a starting model x_0 and regularization parameter β
- Calculate
$$\phi = \beta||Wx_0||^2 + ||b - F[x_0]||^2$$
- Set $\phi^{old} = \phi$
- Until convergence do
  - Linearize 7.15: Calculate the Fréchet derivative $J(x_k)$
  - Check for convergence (equation 7.24).
  - If iteration number is 1, solve 7.20 and obtain δx. Set $\eta_1 = ||\delta x||^2$
  - If iteration number is different than 1 then solve 7.26 using either full-space, Krylov space or subspace methods and obtain δx
  - Calculate
$$\phi^{new} = \beta||W(x_k + \delta x)||^2 + ||F[x_k + \delta x] - b||^2$$
$$\phi^{lin} = \beta||W(x_k + \delta x)||^2 + ||F[x_k] + J(x_k)\delta x - b||^2$$
$$\Delta\phi(x; \delta x) = \phi^{new} - \phi^{old}$$
and the linearized function
$$\Delta\phi^{lin}(x; \delta x) = \phi^{lin} - \phi^{old}$$

- If $\phi^{new} < \phi^{old}$ and $|\Delta\phi - \Delta\phi^{lin}|/max(\Delta\phi, \Delta\phi^{lin}) > \tau$

  accept the step and set: $x_{k+1} = x_k + \delta x$, $\phi^{old} = \phi^{new}$

- Elseif $\phi^{new} < \phi^{old}$ and $|\Delta\phi - \Delta\phi^{lin}|/max(\Delta\phi, \Delta\phi^{lin}) \leq \tau$

  increase $\eta \rightarrow 2\eta$ and re-solve for δx

- Elseif $\phi^{new} > \phi^{old}$

  reject the step and set $\eta \rightarrow 0.5\eta$ and re-solve for δx

7.3.3 Comparison Between Damped Gauss-Newton and Trust Region

We have reviewed two common methods for the solution of the minimization problem 7.15. The main question to ask therefore is which method to use and when? The question does not have a trivial answer and so, as a means of answering this question, we summarize the advantages and disadvantages of each of the methods considering the special structure of our optimization problem. When facing a specific problem one should decide which method to use based on the weaknesses of the methods and the bottle-necks of the problem.

The main advantage of the DGN method is simplicity. Determining the step direction is not related to the step length and therefore each one can be done independently. This gives rise to simple and elegant programs. Another advantage of the DGN formulation is that there are not a lot of parameters which need to be determined. The only parameter which is chosen is the step length ω and there are simple criteria to choose it. From a computational point of view, the separation of the direction and size yields only a single matrix inversion at each DGN step. After the matrix has been inverted, preferably using iterative methods, we work only with a single vector and therefore from a storage point of view we need only to store the matrix (or have a method to calculate it) and three more vectors for the iterative solver. Notice that in the regular implementation of DGN

the iteration matrix $\beta W^T W + J(x)^T J(x)$ is usually positive definite and unless β is very small or W is ill-posed itself, its condition number is reasonable and therefore usually convergence is obtained after a few number of iterations.

There are two major disadvantages to the DGN method. First, if the Gauss-Newton direction is not satisfactory and the step length has to be very short, in the next step we would face a similar Fréchet derivative and the minimization process might stagnate. In order to stop this stagnation it might be better to take a different direction and the trust region algorithm might be better. The second disadvantage of the DGN method is that in each step we solve the linear system to a high accuracy which may not be needed. This disadvantage can be overcome by using Krylov space methods and stopping the iteration with a relatively large residual. This was suggested by Brown and Saad [1990], [1994], and by Eisenstat and Walker [1994]. In this work we use CGLS and stop the iterations when the normalized residual is lower than 0.01.

The main advantages of the trust region method occur exactly where the DGN fails. By changing the direction of the step when the step is very small, there is a better chance to find a satisfactory step and not to get into stagnation. However the disadvantages of the TR method is that it is harder to implement. First, since the direction is related to the step size the program is usually not as elegant as DGN programs. Second, the implementation of TR requires the choice of some arbitrary parameters. Both the parameter τ (which determines if we need to increase the step size) and the reduction of trust region size are somewhat arbitrary. From a computational point of view, calculating the parameter μ for large scale applications is usually not possible. Working with hybrid methods requires extra storage for the subspace vectors and therefore implementation of TR for large scale problems can be done efficiently only through subspace methods.

As a summary, I would prefer DGN over TR for most inverse problems since it is easier to program and implement. However for very nonlinear problems, and especially

for problems where the iteration matrix $\beta W^T W + J(x)^T J(x)$ is not positive definite because the matrix W does not have full rank (such as in Haber and Oldenburg [1996]), I would prefer to use TR methods.

7.4 Common Nonlinear Strategies For Nonlinear Inverse Problems - Review

In this section we review two common techniques for the solution of nonlinear inverse problems. This section is a review of the work of Tarantola [1987], Parker [1994] and Engl *et al.* [1996]. We review the algorithms and the main difficulties of these algorithms. Based on the strengths and weaknesses of the algorithms, we shall suggest new algorithms in the next chapter.

In the previous section we have introduced the ideas of global regularization and reviewed two methods to solve the nonlinear problem which arises from this regularization. However, there is one question which we did not answer. We work with the penalty method and minimize:

$$\phi = \beta||Wx||^2 + ||F[x] - b||^2 \tag{7.34}$$

The main question then is how to determine the regularization parameter β. We now review two strategies for this problem.

7.4.1 The Method of Fixed Regularization Parameter

The first method, which was suggested by Tarantola [1987], Parker [1994] and Engl *et al.* [1996], was to fix the regularization parameter $\beta_k \;\; k = 1, 2...$ and solve the nonlinear optimization problem with similar methods to those suggested in the last section. After the problem has been solved with a regularization parameter β_k we can check if the solution is satisfactory, i.e. if it obeys some criteria. The only criterion suggested was the discrepancy principle. This method is similar to the Tichonov regularization in full

space for the linear problem and it involves the sequential solution of the minimization problem:

$$\phi_k = \beta_k ||Wx||^2 + ||F[x] - b||^2$$

for different β's. Now assume that we have solved the problem for β_k with a solution x_k^*, and found that $\phi_d^*(\beta_k) = ||F[x_k^*] - b||^2 > T$. In this case we pick a new regularization parameter β_{k+1} such that $\beta_{k+1} < \beta_k$. Exactly as for the linear problem we can interpolate between different ϕ_d^* in order to approximate the nonlinear function $\phi_d^*(\beta)$. After solving a few nonlinear problems we could find an acceptable β^* which yields the model x^* such that $||F[x^*] - b||^2 = T$.

The algorithm can be summarized as follows:

Fixed Regularization Parameter

- Choose regularization parameter β_1 and a starting model x_0
- For $k = 1, 2...$ do
  - Solve the minimization problem 7.34 and obtain x_k^*
  - If $|\ ||F[x_k^*] - b||^2 - T| < \delta$ terminate process.
  - Elseif $|\ ||F[x_k^*] - b||^2 - T| > \delta$

    Interpolate/extrapolate the function $\phi_d^*(\beta_k)$ and find a new regularization parameter β_{k+1}

The main difficulty in this algorithm is that we need to solve a new nonlinear problem at each stage and therefore this algorithm is computationally expensive. The algorithm is substantially cheaper if we estimate β_1 to be close to the optimal β^*. In Engl *et al.* [1996] and Bakushinsky *et al.* [1994], considerable effort is made to choose such a β_1. However, the estimates there are asymptotic in nature and 7.34 has to be solved many times.

In order to better understand this and other such algorithms, we introduce the L-curve for the nonlinear problem. Although there is no guarantee that the L-curve would have the typical L-shape it has in the linear case, we assume that this is the case. Every point on this curve is a solution to the problem 7.34 and we could plot the path of minimization on this curve. This path is plotted in Figure 7.1 From this figure it is clear

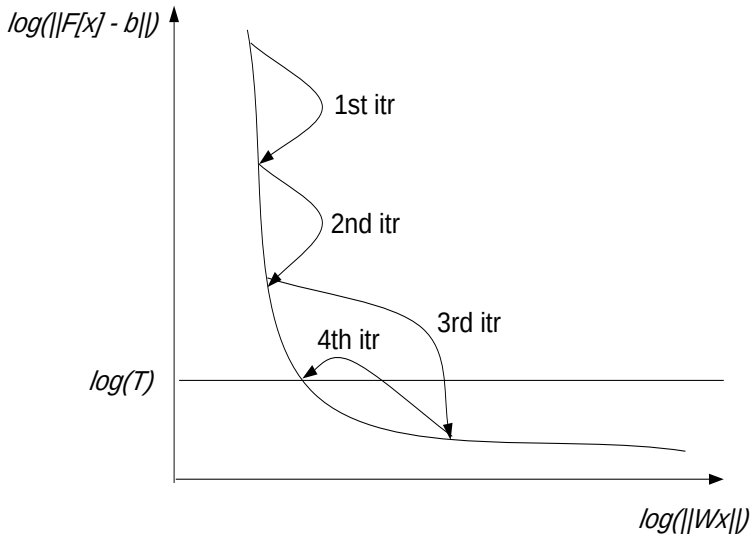

Figure 7.1: A hypothetical path for the solution of a nonlinear ill-posed problem by minimizing for four different β's.

that some of the work which is done using this method is a waste. The method works hard in order to get to a good solution for the minimization problem at the k^{th} stage when such a solution is not needed. For this reason, this method is not used very much and a cheaper substitute has to be developed.

7.4.2 The Two-Stage Method of Constable Parker and Constable

As a cheaper substitute to the fixed regularization parameter, Constable *et al.* [1987] have developed a different methodology. In their famous paper, the authors named their algorithm Occam's inversion because they wanted to differentiate their algorithm from

the creeping algorithm which we discussed in the beginning of this chapter. This thesis tries to find only models which minimize a global objective function and therefore the underlying philosophy of algorithms here is not different from theirs. However, the way to achieve this Occam goal is different. We therefore refer to the Constable *et al.* algorithm as the Two-Stage Method (TSM). The reason for the name is revealed next.

The goal of the TSM is to minimize the global objective function 7.34. As in all Newton-type methods, the equation is linearized:

$$\phi \approx ||F[x] + J(x)\delta x - b||^2 + \beta||W(x + \delta x)||^2 = \phi_d^{lin} + \beta\phi_m \tag{7.35}$$

Differentiating with respect to δx and equating to zero gives:

$$(J(x)^T J(x) + \beta W^T W)\delta x = J(x)^T(b - F[x]) - \beta W^T W x \tag{7.36}$$

As explained before, the main problem is that the regularization parameter β is not known. The major idea of the TSM is to choose a different β in each iteration. The process is divided into two stages (hence the name two-stage method). In stage one the misfit ϕ_d is reduced to some target misfit and in stage two this target misfit is kept constant while the model norm ϕ_m is reduced. The process can be viewed as combining two methods. Stage one is a penalty stage in which the regularization parameter takes the role of penalizing the model norm. The second stage can be viewed as a Lagrange process in which we try to minimize a function while staying on the constraints. Although these two processes have similarities, they are different. While one is trying to get to the constraint and therefore use a global objective function, the other stays on the constraint which means that it tries to obtain a saddle point. The algorithm can be summarized as follows:

Occam's Method through the Two-Stage Method

Stage 1.

- 1.a. Pick a starting model x_0, a tolerance misfit level T and tol.
- 1.b. while $||b - F[x_k]||^2 > T$
  - line search:

    for $i = 1 : number\ of\ \beta$'s
    * Solve
    $$(J_k^T J_k + \beta_i W^T W)\delta x_k^{(i)} = J_k^T(b - F[x_k]) - \beta_i W^T W x_k$$
    * Form the update:
    $$x_{k+1}^{(i)} = x_k + \delta x_k^{(i)}$$
    * Calculate the misfit for the updated model:
    $$\phi_d{}^{(i)} = ||b - F[x_{k+1}^{(i)}]||^2$$
  - From all the steps $\delta x_k^{(i)}$ pick the one which gives the lowest misfit, or a step such that $T = ||b - F[x_k + \delta x_k^{(i)}]||^2$. Call this step $\delta x_k^{(*)}$.
  - Update the model:
  $$x_{k+1} = x_k + \delta x_k^{(*)}$$

Stage 2.

- 2.a. while $||x_{k+1} - x_k|| > tol$
  - line search

    for $i = 1 : number\ of\ \beta$'s
    * Solve
    $$(J_k^T J_k + \beta_i W^T W)\delta x_k^{(i)} = J_k^T(b - F[x_k]) - \beta_i W^T W x_k$$

* Form the update:

$$x_{k+1}^{(i)} = x_k + \delta x_k^{(i)}$$

* Calculate the misfit for this model:

$$\phi_d{}^{(i)} = ||b - F[x_{k+1}^{(i)}]||^2$$

– From all the steps $\delta x_k^{(i)}$ pick the one which gives the misfit, $T = ||b - F[x_k + \delta x_k^{(i)}]||^2$ and possesses the smallest model norm. Call this step $\delta x_k^{(*)}$.

– update:

$$x_{k+1} = x_k + \delta x_k^{(*)}$$

Since there is no formal proof of convergence of this algorithm the question which can be asked is: Does the algorithm reach the goal of minimizing a model objective function subject to fitting the data to some tolerance T?

In our work we have found that the answer to this question can be no. In order to demonstrate, let us construct an extremely simple problem. Suppose we measure the gravity field next to a fault. Assume we have two thin layers with known density anomalies ρ_1 and ρ_2 but unknown depths x_1, x_2 (Figure 7.2). It can be shown that the gravity field (the data) at distance d from the fault is given by:

$$F[x] = F[x_1, x_2] = \rho_1 \; arctan(\frac{d}{x_1}) + \rho_2 \; arctan(\frac{d}{x_2}) + \epsilon = b \qquad (7.37)$$

Now suppose we have only a single datum with some noise measured at $d = 1$ meter away from the fault, and we want to recover the "best" combination x_1, x_2. In order to do that we define the objective function:

$$\phi = \beta||x||^2 + ||F[x] - b||^2 =$$

$$\beta(x_1^2 + x_2^2) + (\rho_1 \; arctan(\frac{1}{x_1}) + \rho_2 \; arctan(\frac{1}{x_2}) - b)^2$$

Our goal in this simple example is to minimize this function such that the misfit $T = 0.01||b||^2$. For this simple example we let $\rho_1 = \rho_2 = 0.785$ and $b = 1$. We now apply the TSM starting from the point $x_0 = [3, 2]$.

In order to view the results we put in a table, in each iteration, the misfit as a function of $x = [x_1, x_2]$. We also plot all the possible models which are achieved by the β line search (step 1.b). Four of the iterations are plotted in Figure 7.3 . Notice that for all of the plots as β goes to infinity, the models go to zero as expected. In each iteration we try to decrease the misfit as much as possible. The minimum norm solution subject to fitting the data to the desired tolerance level is marked by a star at the point $[1.2, 1.2]$. Note that for the four iterations plotted we did not get closer to the minimum.

It is clear that the TSM failed and did not reach the true minimum in this extremely simple problem. We now ask the question why?

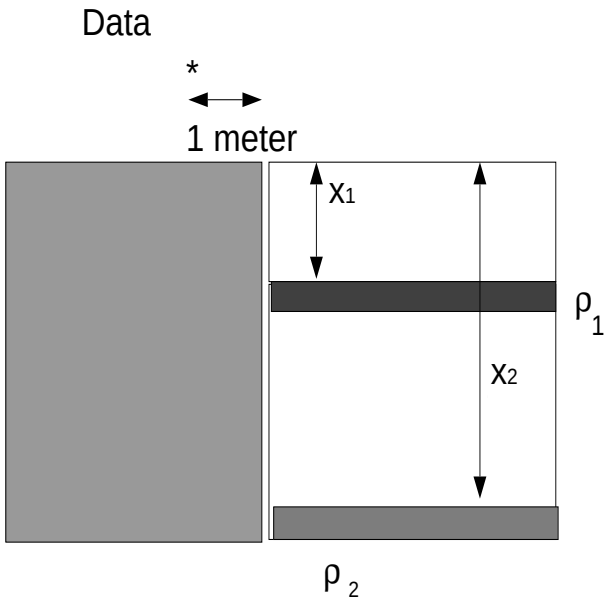

Figure 7.2: The example problem: One datum is collected near the edge of a fault. We try to recover x_1 and x_2.

x_1	x_2	$\|\|F[x]-b\|\|^2$	$\|\|x\|\|^2$
3	2	0.14	13.0
0.91	1.45	0.016	2.93
1.71	0.83	0.011	3.61
0.82	1.74	0.011	3.69

Table 7.1: Path of minimization

There are three main reasons for the failure. The first one is that while the method takes care of the non-uniqueness by minimizing a global objective function, it does not take into consideration the nonlinearity of the problem. The Taylor expansion is $F[x+\delta x] = F[x] + J(x)\delta x + R(x,\delta x)$. We are allowed to neglect the remainder only if it is small. Since each step in the TSM is not constrained, it might well be that $F[x]+J(x)\delta x$ is a very bad approximation to $F[x+\delta x]$. In this case the next iterate does not have any real connection to the function we are minimizing.

A second important aspect of the TSM is that in stage one we accept steps according to their misfit reduction. Thus at the first stage we are really concerned only with the misfit and therefore minimize $||F[x]-b||^2$. However the mathematical problem we posed was to minimize a combination of misfit and model norm: $\beta||x||^2+||F[x]-b||^2$. Therefore it could well be that while the misfit was reduced the function we attempted to minimize actually increased! Thus the TSM could give a totally faulty step.

The third reason the method can fail is in stage two. In stage two we are supposed to stay on the same misfit level while reducing the model norm. This process can be written as a constrained minimization problem:

$$minimize \quad ||Wx||^2$$

$$subject\ to \quad ||F[x]-d||^2 = T$$

As discussed before, the domain $||F[x]-b||^2 = T$ is non-convex and therefore even if

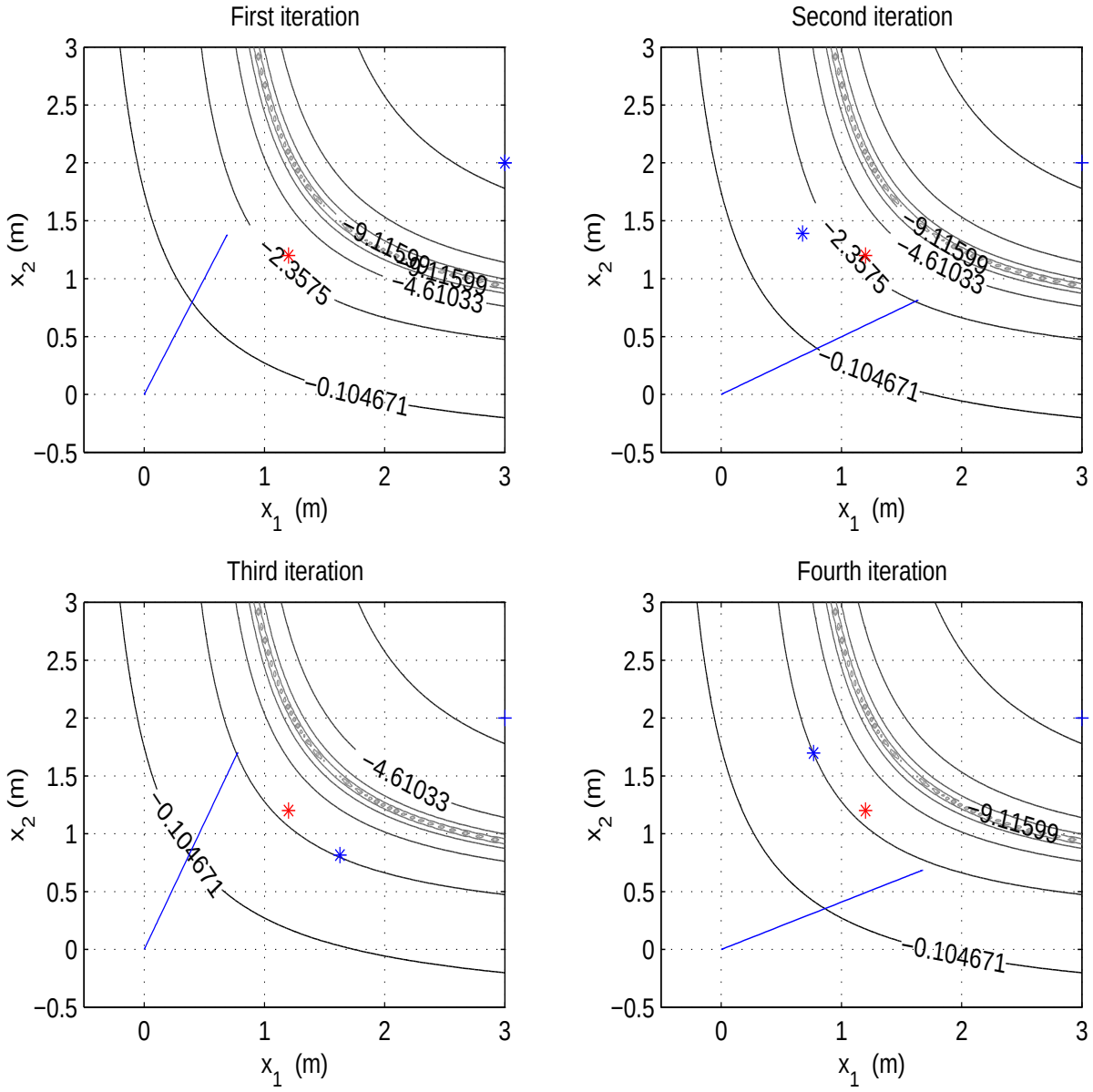

Figure 7.3: Four iterations of the example problem. The contour represents the log of the misfit, $log(||F[x_1, x_2] - b||^2)$. The star at $[1.2, 1.2]$ denotes the location of the true minimum. The straight black line represents all feasible solutions to equation 7.35 with β changing from $1E-9$ to $1E6$

$||F[x] - b||^2$ is a convex function we might not get to the true minimum due to the non-convexity of the domain.

Although we have just shown that this algorithm might not converge to the right result, previous experiments (see for example Parker [1994], Jingsheng and Elsworth [1995], Smith and Booker [1991], Thompson *et al.*, Zhang [1995], Li and Oldenburg [1996], Farquharson [1995]) shows that it may give reasonable results. In order to understand why the TSM seems to work in many cases we examine this algorithm with the L-curve. The paths of minimization for a "successful" TSM and an "unsuccessful" TSM are plotted in Figure 7.4. Since in the first stage of the TSM we are concerned only with the misfit we can end this stage anywhere on the line $\phi_d = T$. If we are lucky enough, this point is going to be close to the minimum and with some local improvement in stage two, we get close to the minimum and the result of such minimization is considered succesful. This

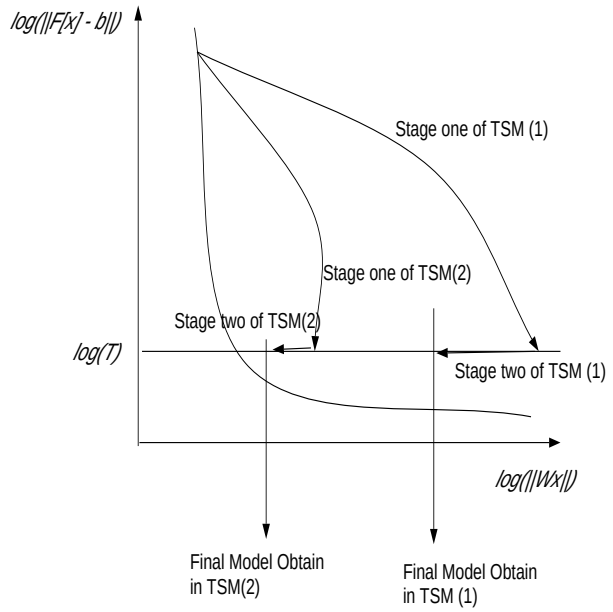

Figure 7.4: Two possible paths for the TSM. In the first path TSM(1) we end with a model with the right misfit but with large model norm and in the second, TSM(2), we end with a reasonable model but, not the smallest model.

path is plotted by the TSM(2) line. However if we are not so lucky we might end stage one of the TSM very far from the minimum, and since stage two of TSM could not get us all the way back to the minimum, we would end the process with an unsuccessful model. Such a path is plotted in TSM(1). The possible failure of the TSM when decreasing the misfit very fast is a known phenomenon, however it has not been explained so far, and therefore different algorithms try to avoid reducing the misfit fast by setting *ad hoc* parameters, which are obtained by trial and error.

Obviously the basic idea of Occam is good because it deals with the non-uniqueness of the inverse problem, however the way it is being carried out in the TSM, can lead to non-optimum solutions. Our goal is to present a methodology which achieves the goal of Occam's idea, and minimize a global objective function without ignoring the nonlinearity of the problem. This will be done in the next chapter.

Chapter 8

Methods for Nonlinear Inversion

In the last chapter we have reviewed the current methodologies for nonlinear inversion. This chapter is dedicated to the development of new strategies. We start with a straightforward improvement of the constant regularization parameter technique, then develop a new technique based on the GCV. Finally, we discuss subspace and hybrid methods that can reduce computations.

8.1 The Cooling Method

In the last chapter we discussed the method which uses a constant regularization parameter. Recall that we want to minimize:

$$\phi = \beta||Wx||^2 + ||F[x] - b||^2 = \beta\phi_m + \phi_d \tag{8.1}$$

In the constant regularization method, we pick a regularization parameter and carry the nonlinear minimization all the way through to its solution. If the result is not satisfactory, we pick a new regularization parameter and repeat the process. We define each minimization process for a different regularization parameter as the outer iteration, we define every DGN or TR step as the inner iteration, and we define the linearized system which is solved in each inner iteration as the inner-inner iteration.

The process for a constant regularization parameter, which is plotted in Figure 7.1, is not efficient since we invest work in every outer iteration only to get to a minimum of 8.1 which is not desired and is used only as a starting point for the next outer iteration. The

first improvement to be made then, is to generate a process in which we do not invest in expensive nonlinear solutions for regularization parameters which we think will not give the desired misfit. Another goal of this process is to generate a series of models which gradually give a better fit to the data, and give rise to a gradually increasing model norm. These are the major features of the cooling process which is developed next.

In order to understand the cooling process, first note that equation 8.1 contains two parts. The model norm ϕ_m is quadratic and gives rise to a linear system of equations, and the misfit ϕ_d is the nonlinear part. The function ϕ is almost quadratic if the regularization parameter is large, and it is very non-quadratic if the regularization parameter is small. It is very well known that solving a nonlinear optimization problem takes less work if we start close to the solution, and therefore if we "jump" from one regularization parameter to the other in very large steps, the minimum of 8.1 would change significantly and the solution of one problem would not be a good starting point for the next minimization problem. The cooling process then, is a process in which the regularization parameter changes or cools slowly, such that the solution which was obtained using one regularization parameter is a good starting point for the next.

The first question which can be asked then is how to choose the first regularization parameter and the first starting model. Since we want to minimize $||Wx||$, where this norm is supposed to be based on *a priori* information, we pick as a starting model, the model x_0 which gives $||Wx_0|| = 0$. Starting from other models does not make sense since if we think that the model is similar to a model x_1 such that $||Wx_1|| \neq 0$, we are not utilizing our *a priori* information about the problem, and therefore miss the purpose of regularization. In this case we should change the problem and add this information into the inverse problem by minimizing $||W(x - x_1)||$.

After x_0 has been chosen, we need to choose the first regularization parameter. There are three possible options. The first is to start with a small regularization parameter

and then to increase it slowly. This process does not seem to be reasonable since our starting point x_0 is not close to the minimum of the function ϕ which possesses a small regularization parameter, β. Another possible option is to guess a "reasonable" β and to start from there. The trouble with this option is that it is hard to have such a good estimation, and therefore this process is not very practical. A third and reasonable choice is to start with a regularization parameter β which is very large. This option has not only the advantage of a good starting point x_0 which is close to the minimum of that function, but also if the regularization parameter is large then the function ϕ is almost quadratic and therefore the minimum should be obtained after a few steps. We therefore start with a large regularization parameter. One option for ensuring that the regularization parameter is large, is to start with a regularization parameter which ensures the condition:

$$\beta W^T W + J(x_0)^T J(x_0) \approx \beta W^T W$$

This can be done by looking at the largest singular value of $J(x_0)$ and assuming that $W = I$ (which can be done using the transformation to standard form), and choosing a regularization parameter $\beta_1 = \gamma \; max(SVD(J(x_0)))$. In this work we have found that $\gamma = 2$ was satisfactory. If the SVD of $J(x_0)$ is not avaliable then we approximate it by choosing a random vector $v \in R^M$ and estimating the regularization parameter by:

$$\beta_1 = \gamma \frac{||J(x_0)v||}{||v||} \tag{8.2}$$

We can now start the iteration process and minimize $\phi(\beta_1, x)$ starting with x_0. However we have a good idea that this minimization process would not yield the desired model x^* and therefore we terminate this iteration relatively fast i.e., we use the criterion for convergence 7.24. with higher tolerance level.

$$\frac{|g(x)|^T |x|}{\phi(x)} < \delta_1 \tag{8.3}$$

where δ_1 is relatively large. In this work we used $\delta_1 = 0.01$. We found out in most experiments that we need only one or two inner iterations to achieve this criterion in the first outer iteration. In the next stage we have to pick a new regularization parameter β_2 and repeat the process.

One very common demand from this regularization parameter β is that it would decrease the misfit by a factor η. We now present a method to guess such a regularization parameter. Assume that we have carried out $k > 1$ outer iterations, thus we have minimized (to some extent) k functions $[\phi(\beta_1), .., \phi(\beta_k)]$ and we have the quantities

$$[\phi_m(\beta_1), .., \phi_m(\beta_k)] = [\phi_m^1, .., \phi_m^k]$$

$$[\phi_d(\beta_1), .., \phi_d(\beta_k)] = [\phi_d^1, .., \phi_d^k]$$

and

$$[\beta_1, .., \beta_k]$$

Our goal is to choose a new regularization parameter β_{k+1} based on the knowledge of these quantities. In order to do that we notice that at a stationary point of $\phi(\beta)$:

$$\Delta\phi = \Delta\phi_d + \beta\Delta\phi_m = 0 \tag{8.4}$$

Therefore

$$-\frac{\Delta\phi_d}{\Delta\phi_m} = \beta \tag{8.5}$$

This means that if we plot the misfit versus the model norm, the slope of the curve is given by the regularization parameter β. We can then write the following equations:

$$\phi_d(\phi_m + \delta\phi_m) \approx \phi_d(\phi_m) - \beta\delta\phi_m \tag{8.6}$$

and

$$\beta(\phi_m + \delta\phi_m) \approx \beta(\phi_m) + \frac{\partial\beta}{\partial\phi_m}\delta\phi_m \tag{8.7}$$

From our experience we know that the change in β is usually logarithmic, while the change in ϕ_m is linear. It is therefore useful to replace 8.7 by:

$$log(\beta(\phi_m + \delta\phi_m)) \approx log(\beta(\phi_m)) + \frac{\partial(log(\beta))}{\partial\phi_m}\delta\phi_m \tag{8.8}$$

As stated at the beginning, the goal is to find a regularization parameter which yields a reduction of η in the misfit and therefore we let:

$$\phi_d(\phi_m + \delta\phi_m) = \phi_d(\phi_m) - \beta\delta\phi_m = \eta\phi_d(\phi_m)$$

From this equation we can find that the desired change in the model norm $\delta\phi_m$ is:

$$\delta\phi_m = \frac{(1-\eta)\phi_d}{\beta} \tag{8.9}$$

Substituting the expression for $\delta\phi_m$ in the equation 8.8 we get:

$$log(\beta(\phi_m + \delta\phi_m)) = log(\beta(\phi_m)) + \frac{\partial(log(\beta))}{\partial\phi_m}\frac{(1-\eta)\phi_d}{\beta} \tag{8.10}$$

Thus if we can approximate the derivative of $log(\beta)$ with respect to ϕ_m we have the desired approximation. The derivative $\partial log(\beta)/\partial\phi_m$ is approximated by the secant approximation:

$$\frac{\partial log(\beta)}{\partial\phi_m} \approx \frac{log(\beta_k) - log(\beta_{k-1})}{{\phi_m}^k - {\phi_m}^{k-1}} \tag{8.11}$$

which gives:

$$log(\beta_{k+1}) = log(\beta_k) + \frac{log(\beta_k) - log(\beta_{k-1})}{{\phi_m}^k - {\phi_m}^{k-1}} \quad \frac{(1-\eta){\phi_d}^k}{\beta_k} \tag{8.12}$$

Using this strategy we decrease the regularization parameter β and the misfit until we expect to get the target misfit T. At this stage we switch the tolerance level in the convergence criterion 8.3 to our real tolerance level δ which is used in Chapter 7.

Two questions which are still open are the second step of the algorithm and the misfit reduction factor η. Since we can start to predict β_{k+1} only if we already calculated $\beta_k \quad k > 1$, the question which needs to be answered is how to predict β_2. After the first

outer iteration we cannot have information about the derivative of β with respect to ϕ_m and therefore we just reduce β by a fixed amount. In this work we have used $\beta_2 = 0.9\beta_1$. This choice yields fast convergence of the inner iteration (usually 1-2 inner iterations). In the next stage we continue with the β estimation 8.12. The second question is the misfit reduction parameter η that has to be chosen for each outer iteration. Recall that the whole process of cooling is based on relatively small variations in the different minimization problems which are solved in the outer iteration, thus we do not want to decrease the misfit at the expense of an abrupt increase in the model norm. We therefore allow only an increase of a factor γ in the model norm, where $\gamma = 0.5$. This choice is somewhat arbitrary and if the problem is highly nonlinear we might want to make γ smaller. In order to predict the relation between the change of misfit to the change of model norm we use equation 8.9. First we set $\eta = 0.5$. The predicted change in model norm for this choice is:

$$\delta\phi_m = \frac{\phi_d}{2\beta} \tag{8.13}$$

If this choice is satisfactory, i.e., $\delta\phi_m/\phi_m < \gamma$, we want to find η such that:

$$\frac{(1-\eta)\phi_d}{\beta} = \phi_m\gamma \tag{8.14}$$

which gives:

$$\eta = 1 - \frac{\gamma\beta\phi_m}{\phi_d} \tag{8.15}$$

Finally we summarize this algorithm as follows:

Inversion Through Cooling

- Choose a model x_0 such that $||Wx_0|| = 0$. Calculate the Fréchet derivatives $J(x_0)$, $\beta_1 = 2max(SVD(J(x_0)))$ or use 8.2.

- Outer Iteration: For $k = 1, 2, 3...$

- Inner iteration: minimize 8.1 using DGN or TR with stopping criteria 8.3 and $\delta_1 = 10^{-2}$.
- Calculate ${\phi_d}^k, {\phi_m}^k$
- If $k = 1$, $\beta_2 = 0.9\beta_1$
- Elseif $k > 1$, choose $\eta = 0.5$ or $\eta = 1 - \gamma\beta\phi_m/\phi_d$ (as explained in the above section). Choose β_{k+1} using 8.12.
- If the predicted misfit is achieved (using 8.5) switch δ_1 in the stopping criteria for the inner iteration to δ.

Finally we view the iteration on the L-curve. The curve demonstrates that this

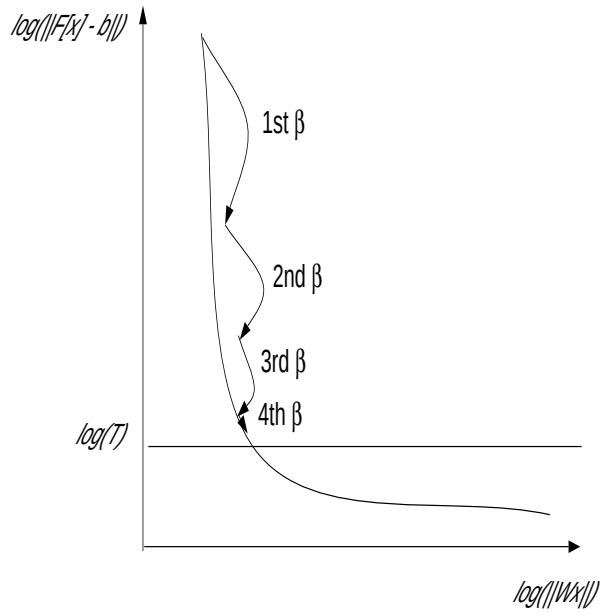

Figure 8.1: The path of the cooling strategy on the L-curve.

algorithm can be cheaper and safer than the regular β search as suggested in the previous chapter. It is cheaper because we do not waste expensive function evaluations on trying to obtain the total minimum of the objective function for β values which we are not

interested in. Also, by starting each inner iteration from a point which is close to the minimum, we speed up the inner iteration. The method is safer since we move slowly down the L-curve so even if we do get trapped in a local minimum, at least it would be a local minimum with a small model norm. The main problem with this method is that in order to use it we need to have a predicted target misfit, which is usually not avaliable. Second, the path to the point x^* is still not optimal. As stated before, if we could only predict the right β the process would be shorter. This is the goal of our next section.

8.2 Nonlinear Inversion Through Generalized Cross Validation

In the last section we used a fixed regularization parameter in the outer iteration, in order to perform a few Gauss-Newton inner iteration steps and achieve a specific target misfit. In this section we discuss how to make only one step of the inner iteration for each outer iteration, and to obtain a regularization parameter based on the GCV principle which we reviewed in Chapter 3.

8.2.1 Full-Space Nonlinear Inversion

Our goal is to decide on an adaptive regularization parameter. In order to do that we notice that at the k^{th} iteration we face a linearized problem of the form:

$$J(x_k)\delta x = b - F[x_k] + \ nonlinear\ terms\ +\ noise \tag{8.16}$$

We therefore have to deal with two problems. The first is the measurement noise and the second is the nonlinear terms. While the noise has to be treated through a global regularization, i.e. assessing the regularization parameter, β, for a global objective function, the nonlinear terms have to be treated by reducing the step size in DGN or obtaining a local regularization parameter μ for TR algorithm. These processes are not necessarily pulling in the same direction and therefore we want to treat them separately and differently.

First we deal with the measurement noise. Ideally, we need a method which can differentiate between the Gaussian noise and the correlated nonlinear terms. Such a method would provide a value of β for the current iteration. Then using this β, we carry out one step of DGN or TR iteration. The damped DGN or TR takes the nonlinearity into consideration and makes sure that the nonlinear terms are actually small.

Applying such a process is not straight-forward. The first thing we have to remember is that at each linear iteration we are minimizing a different norm. The norm we want to minimize at the k^{th} iteration is $||W(x_k+\delta x)||$ and therefore the regularization parameters for noise estimation and the step length should be taken with respect to this norm. Our algorithm can be summarized as follows:

Nonlinear inverse problem through Noise Estimation

- 1. Calculate the sensitivities $J(x_k)$
- 2. Calculate the regularization parameter β.
- 3. Use the regularization parameter to calculate δx using DGN or TR.
- 4. Update using step length strategy.
- 5. Check for convergence and go to 1

There are three important points in this algorithm which need to be explained. First we need to explain how to pick a regularization parameter (stage 2). We delay this explanation for now. We also need to explain how to make the update (stage 4) and how to check for convergence (stage 5).

The main problem with the update is that the objective function changes from iteration to iteration. While in the Gauss-Newton method it is clear that $\phi(x_k) > \phi(x_{k+1})$,

it is not clear that this is the case in this algorithm. The main reason is that

$$\phi(\beta_k, x_k) = \beta_k ||W x_k||^2 + ||F[x_k] - b||^2$$

while

$$\phi(\beta_{k+1}, x_{k+1}) = \beta_{k+1} ||W x_{k+1}||^2 + ||F[x_{k+1}] - b||^2$$

Since the global objective function ϕ is changing at each iteration, the demand of decreasing the value of the objective function is not reasonable. We therefore replace it with the consistent demand:

$$\phi(\beta_{k+1}, x_k) \; > \; \phi(\beta_{k+1}, x_{k+1}) \tag{8.17}$$

which means that:

$$\beta_{k+1} ||W x_k||^2 + ||F[x_k] - b||^2 > \beta_{k+1} ||W x_{k+1}||^2 + ||F[x_{k+1}] - b||^2 \tag{8.18}$$

Thus every step in the algorithm is also equivalent to a one-step descent from the model x_k to x_{k+1} with regularization parameter β_{k+1}. This point is extremely important if we want our algorithm to be consistent with the objectives of the minimization.

The second important point is convergence (stage 5). For every iteration k we need to know whether to stop the process or to continue on. Since the objective function is changing the question is what criterion should be used? The answer is again given by consistency. If each iteration is a Gauss-Newton iteration with different parameter β_k then our convergence criterion is the same as for a Gauss-Newton algorithm 7.24.

The third and the last thing to explain is the method in which we choose β for each iteration. As stated before we need a method which can differentiate between the nonlinear terms and the noisy terms. Recall from Chapter 6 the experiment on the linearized gravity problem. We found that GCV did not detect the nonlinear terms at each iteration and hence it yielded a regularization parameter which penalizes only

against the uncorrelated Gaussian correlated noise. Our observation is not the only one for this behaviour. Altman [1987] and Nychka *et-al.* [1984] experimented with smooth correlated errors, and noticed that the GCV did not penalize correlated smooth noise. This observation can be used to our benefit. It means that GCV responds only to the Gaussian noise and therefore it can be used to estimate a global regularization parameter. The GCV does not regularize nonlinear terms, and therefore we need to add the second regularization which ensures that the steps are small enough and that the linearization process holds.

To summarize, our methodology is based on three main components.

- 1. A method to pick a regularization parameter (GCV)
- 2. A method to pick a step size. (DGN or TR)
- 3. A method to accept/reject a step

Step one is based on the ability of GCV to differentiate between noise and signal while steps two and three are based simply on DGN or TR methods. In general, our method can be simply viewed as a variation of the fixed regularization parameter method, with a regularization parameter β which is changing at each iteration. If the regularization parameter approaches a specific value β^* then our algorithm turns into a Gauss-Newton algorithm. However if the GCV process at each iteration yields a different regularization parameter the process might not converge. Our experience has been that this has not happened and that the regularization parameter tends to converge quickly to its final value. This point will be demonstrated in Chapter 10.

Finally we discuss our algorithm on the L-curve. If the regularization parameter is correctly estimated, then it does not change very much through the process and the path from the starting point x_0 to the final model x^* is almost direct. The process is plotted in Figure 8.2

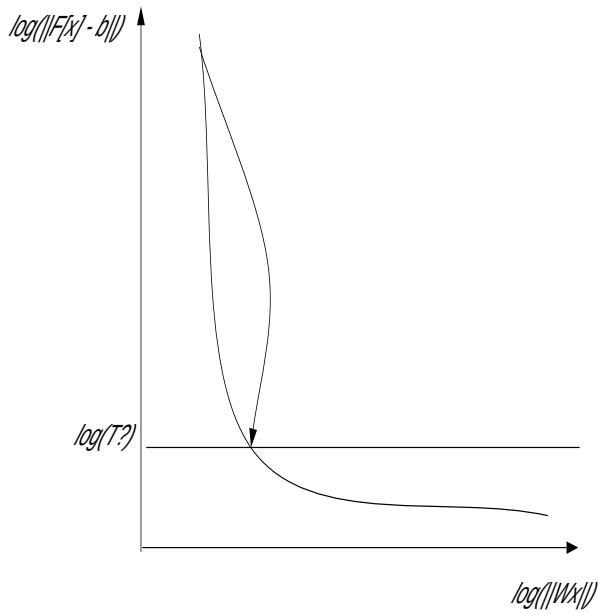

Figure 8.2: The path of the GCV strategy on the L-curve.

8.2.2 Subspace Methods For Nonlinear Inversion

So far we have discussed Tichonov-style regularization for nonlinear problems. These techniques are based on the ability to estimate a regularization parameter and invert the regularized Fréchet derivatives. The main problem is that just as for the linear case, the inversion of the Fréchet derivatives for large scale problems is costly, and therefore some shortcuts are needed. The obvious shortcut is to use the methodology we have developed in the linear case and to use a cheap matrix inversion through Krylov subspace, in order to calculate the solution of the matrix inversion. We have suggested this for the calculation of the TR step and the calculation of the DGN step.

The second shortcut, which is more effective, is to use the properties of the subspace in order to substitute for the regularization parameter β, just like in the linear case. The space size acts as a global regularization parameter which penalizes against noise, and then we use another local regularization in order to penalize against the nonlinearity.

The problem can be formulated as follows:

$$minimize \quad ||F[x] - b||^2 \tag{8.19}$$

$$subject\ to \quad x \in \mathcal{K}(J_W(x), r(x), n)$$

where $r(x)$ is some right hand side (which will be define next), $J_W(x)$ is the reduction of the matrix $J(x)$ to the standard form with the matrix W.

In order to work with the subspace formulation we recall from Chapter 7 that the Gauss-Newton iteration can be formulated either for the perturbation, δx, or for the model at the next iteration, x_{k+1}. If we formulate the iteration using the perturbation, then we would have terms that depend on β on the right hand side (see equation 7.20). However if we formulate the problem using the next iteration then we get (7.22):

$$(J(x_k)^T J(x_k) + \beta W^T W)x_{k+1} = J(x_k)^T(b - F[x_k] + J(x_k)x_k) \tag{8.20}$$

This is identical to the least squares problem:

$$\begin{bmatrix} J(x_k) \\ \sqrt{\beta}W \end{bmatrix} x_{k+1} = \begin{bmatrix} b - F[x_k] + J(x_k)x_k \\ 0 \end{bmatrix} \tag{8.21}$$

Which is identical to the Tichonov regularization of:

$$J(x_k)x_{k+1} = b - F[x_k] + J(x_k)x_k$$

As was demonstrated in Chapters 4 and 5, a similar solution to that problem can be obtained using Krylov space regularization. Therefore if problem 8.21 is transformed into its standard form it could be replaced with the subspace problem:

$$minimize \quad ||J(x_k)x_{k+1} - b + F[x_k] - J(x_k)x_k||^2 \tag{8.22}$$

$$subject\ to \quad x_{k+1} \in \mathcal{K}(J_W(x_k), r(x_k), n)$$

where now we see that

$$r(x_k) = b - F[x_k] + J(x_k)x_k$$

In every step we use the Krylov space in order to regularize the problem and obtain a suggested model x_{k+1}. This new model, although it solves the linearized misfit problem, does not necessarily reduce the nonlinear misfit which we try to minimize. A simple solution to this problem is to use a step length strategy and calculate the direction $\delta x = x_{k+1} - x_k$ and to use the same method as in the DGN method to calculate the step length. The algorithm can be summarized as follows:

Krylov Subspace - Damped Gauss-Newton Method

- Choose an initial model x_0. Calculate the misfit $\phi_d = ||b - F[x_0]||^2$
- For $k = 1, 2...$
  - 1. Calculate $J(x_k)$
  - 2. Solve:

    $$J(x_k)x_{k+1}^p = b - F[x_k] + J(x_k)x_k$$

    using CGLS and GCV stopping criterion.
  - 3. Calculate the perturbation $\delta x = x_{k+1}^p - x_k$

    and the new misfit ${\phi_d}^{new} = ||b - F[x_{k+1}^p]||^2$
  - 4. If ${\phi_d}^{new} < \phi_d$, set $x_{k+1} = x_{k+1}^p$ go to 1.
  - 5. Elseif ${\phi_d}^{new} > \phi_d$ set

    $$x_{k+1}^p = x_k + \omega\delta x \quad 0.1 < \omega < 0.5$$

    go to 3.

– 6. If

$$\frac{||x_{k+1} - x_k||}{max(||x_{k+1}||, ||x_k||)} < \delta$$

terminate the process

In the implementation of this process we chose $\omega = 0.5$ and the stopping criterion was $\delta = 10^{-3}$. Notice that this algorithm does not get to a minimum of a functional like the Tichonov regularization and in principle, the solution of this method is different from Tichonov solution. However as we saw in the linear case, the solutions are very similar to the full space regularization and the cost of this solution is substantially less than a full space solution.

8.2.3 Hybrid Methods For Nonlinear Inversion

The subspace implementation which we discussed in the last subsection might give solutions which are different from the Tichonov solution. One way to obtain a more Tichonov-like solution is to use a hybrid method. Again, the main purpose is to avoid inverting directly the matrix $J(x)^T J(x) + \beta W^T W$. The problem can be presented as:

$$minimize \quad ||F[x] - b||^2 + \beta ||Wx||^2 \tag{8.23}$$

$$subject\ to \quad x \in \mathcal{K}\ (J_W(x), r(x), n)$$

where $r(x)$, $J_W(x)$ are the same as in 8.19. Linearizing and formulating the objective function we get:

$$minimize \quad ||J(x_k)x_{k+1} - b + F[x_k] - J(x_k)x_k||^2 + \beta ||Wx||^2 \tag{8.24}$$

$$subject\ to \quad x_{k+1} \in \mathcal{K}\ (J_W(x_k), r(x_k), n)$$

Again this is equivalent to the solution of the system:

$$J(x_k)x_{k+1} = b - F[x_k] + J(x_k)x_k$$

using hybrid Krylov space methods. The algorithm can be summarize as follows:

Krylov Hybrid Subspace - Damped Gauss-Newton Method

- Choose an initial model x_0. Calculate the misfit $\phi_d = ||b - F[x_0]||^2$.
- For $k = 1, 2...$
  - 1. Calculate $J(x_k)$
  - 2. Solve:

$$J(x_k)x_{k+1}^p = b - F[x_k] + J(x_k)x_k$$

    using hybrid LSQR with weighting matrix W and GCV criterion for the regularization parameter β_k.
  - 3. Calculate the perturbation $\delta x = x_{k+1}^p - x_k$,

    the new misfit ${\phi_d}^{new} = ||b - F[x_{k+1}^p]||^2$

    and the new model norm ${\phi_m}^{new} = ||W x_{k+1}^p||^2$
  - 4. If ${\phi_d}^{new} + \beta_k {\phi_m}^{new} \leq \phi_d + \beta_k \phi_m$, set $x_{k+1} = x_{k+1}^p$ go to 1.
  - 5. Elseif ${\phi_d}^{new} + \beta_k {\phi_m}^{new} > \phi_d + \beta_k \phi_m$, set

$$x_{k+1}^p = x_k + \omega \delta x \quad 0.1 < \omega < 0.5$$

    go to 3.
  - 6. If

$$\frac{||x_{k+1} - x_k||}{max(||x_{k+1}||, ||x_k||)} < \delta$$

    terminate the process

In the implementation of this process we chose $\omega = 0.5$ and the stopping criterion was $\delta = 10^{-3}$. This algorithm, although it does not get to a minimum of a functional

like the Tichonov regularization, gives a very similar solution especially if the subspace is chosen such that most of the vectors which are associated with the large singular values have converged.

8.3 Summary

In this chapter we developed two algorithms for solving nonlinear ill-posed problems. Both algorithms are based on understanding the two different processes which we face when solving a nonlinear ill-posed problem. The first process is noise estimation. This process is global in nature, and therefore one has to look for global type techniques in order to estimate the regularization method. Here we suggested the cooling strategy, which is a "safe" method to descend from an estimate of high noise level to low noise level. Cooling does not use the characteristics of random uncorrelated noise and therefore is not very efficient. We therefore suggested the use of GCV as a method to estimate the global noise. We use three variants of the GCV which have been developed in Chapters 3, 4 and 5.

The second process which has to be addressed is the estimation of the step length. If the step length is too large then the linearization does not hold and the iteration might not converge. This process is local in its nature and therefore needs a different type of regularization than the noise estimation process. We have suggested to use two common methods for this process, the damped Gauss-Newton or trust regions. In this way we ensure that the step which we choose is not only regularized against the non-uniqueness of the problem, but we also make sure we descend in each nonlinear step.

Chapter 9

Approximate Fréchet Kernels

In the last chapter we presented a methodology for large scale nonlinear inverse problems. While this methodology deals with the choice of regularization parameter and the matrix inversion, there are still two other bottle-necks for the inversion. The first bottle-neck is the forward modelling, which has to be calculated for each proposed model, and the second is the calculation of the Fréchet derivatives, that is, the sensitivities. While the forward modelling has to be accurate in order to estimate the data misfit, the sensitivities are needed in order to calculate the next step and to check for convergence. It was therefore suggested by Farquharson [1995], Farquharson and Oldenburg [1996], Li [1992], Ellis *et al.* [1993], to approximate the sensitivities. In this section we discuss some of the methods to approximate the sensitivities and the problems which rise from such approximations. We step through two common strategies - the cord or AIM update, (Kelley [1995]) and the Shamanskii update (Shamanskii [1967]). We propose a new methodology for the implementation of these strategies when applied to ill-posed problems. We then propose a secant type update for the nonlinear problem.

9.1 The Concept of Approximate Sensitivities

The concept of approximate sensitivities is not new. Newton suggested approximating the first derivative of a nonlinear minimization problem in one dimension by using information from previous iterations. His approach was extended to systems of equations by Broyden [1965]. Other simple approximations such as the cord and Shamanskii methods also were

suggested for nonlinear equations (see review by Kelly [1995]). However the application of these techniques to nonlinear ill-posed problems is not straight-forward. An ill-posed problem does not have a unique solution and therefore the solution is defined by some minimization problem where the desired model yields the smallest norm $||Wx||^2$ subject to fitting the data. This model minimizes the nonlinear objective function $\phi = \phi_d + \beta\phi_m$ and solves the nonlinear system of equations for the gradients:

$$g(x) = J(x)^T(F[x] - b) + \beta W^T W x = 0 \tag{9.1}$$

This system involves the sensitivities $J(x)$ and therefore if the sensitivities are approximated by a matrix B and are not calculated, we cannot check for convergence, or even solve the correct system of equations, i.e., the gradient system. It is therefore suggested by some authors (Dennis and Schnabel [1996]), to calculate the sensitivities and not to use such approximations. Although they are right if the exact Tichonov solution is needed, just like in the linear problem, we do not have to restrict ourselves to Tichonov solutions only, and we could obtain other reasonable solutions which fit the data using approximate sensitivities. However we should be aware that the solution is different than the Tichonov solution.

The common approach to the approximated sensitivities is local. This approach looks at the linearized approximation problem:

$$g(x + \delta x) \approx J(x)^T(F[x] + J(x)\delta x - b) + \beta W^T W(x + \delta x) = 0 \tag{9.2}$$

and suggests to use approximate sensitivities in order to solve 9.2. While this approach is valid for linear problems it is not necessarily valid for nonlinear problems. The reason is that in the linear case the local (linearized) problem and the global (not linearized) problems are identical. In the nonlinear case if we do not estimate the function $g(x)$ there is no way to know if the model x_k which was obtained by a series of k steps actually

minimizes the objective function. Furthermore, even if the iteration converges to some model $x^{\dagger}$ there is no way to know (without calculating sensitivities) how close it is to the Tichonov solution. But the worst thing is that since we solve a sequence of linear problems without defining a global function, we cannot be sure that the model obtained is not a product of the minimization process, i.e., if we start from different points or carry out the iterations using different strategies (DGN versus TR for example) we get to the same solution.

We therefore suggest a new and different formulation for approximate sensitivities which is global in nature. Our goal in this section is to quantify this approximation. In order to do that we define the Jacobian $H(x) = g'(x)$ and assume that it is bounded. Given the Jacobian the Newton iteration for the solution of the system 9.1 is:

$$\delta x = -H(x)^{-1} g(x) \tag{9.3}$$

We now review a theorem from non-exact minimization which discusses the relation between convergence and errors in the Jacobian of a nonlinear system of equations and errors in the function evaluation

Theorem 9.1: Error in Jacobian and function evaluation (Kelly [1995])

Let $g(x)$ be continuously differentiable and let $H(x) = g'(x)$. Let $\epsilon(x)$ be the error in the evaluation of $g(x)$. Then there are $K > 0$, $\delta > 0$ and $\delta_1 > 0$ such that given the exact solution x^* to the system $g(x) = 0$, if $||x_k - x^*|| < \delta$ and the perturbation, $\Delta(x_k)$, to H, such that $||\Delta(x_k)|| < \delta_1$ then the update:

$$x_{k+1} = x_k - (H(x_k) + \Delta(x_k))^{-1}(g(x_k) + \epsilon(x_k)) \tag{9.4}$$

satisfies:

$$||x_{k+1} - x^*|| \leq K(||x_k - x^*|| + ||\Delta(x_k)||\ ||x_k - x^*|| + ||\epsilon(x_k)||) \tag{9.5}$$

This theorem shows that while the minimization process is forgiving to inaccurate Jacobians $H(x)$, it is not very forgiving to inaccurate calculations of the function $g(x)$.

We have used this characteristic before without noticing. The Gauss-Newton formulation is such that the Jacobian is approximated such that the second derivatives of $F[x]$ with respect to x are not calculated. If we start from a point x_0 such that $||x_0 - x^*||$ is small enough, then using the wrong Jacobian would slow down the convergence rate, however errors in the function $g(x)$ would cause an error in the final result. Thus approximate sensitivities cannot give the solution of the Tichonov problem but rather solve a close problem to the gradients. We define the "near-by problem" to 9.1 as the following system of equations:

$$f(x) = \beta W^T W x + B^T(F[x] - b) = 0 \tag{9.6}$$

where the matrix B is hopefully close to the sensitivities $J(x)$.

We define the solution of 9.6 as $x^\dagger$. The Jacobian of the system 9.6 is given by:

$$G(x) = \beta W^T W + B^T J(x) \tag{9.7}$$

The Newton step, x^N_{k+1}, for the solution of the system 9.6 is then:

$$x^N_{k+1} = -G(x_k)^{-1} f(x_k) = -(\beta W^T W + B^T J(x_k))^{-1}(\beta W^T W x_k + B^T(F[x_k] - b)) \tag{9.8}$$

The Newton step, although it converges quickly, contains the term $J(x)$ which we try to avoid calculating. This is not a problem since the theorem of errors in the Jacobian and in function evaluations states that a small error in the Jacobian just slows the process and therefore we replace 9.8 by an approximate sensitivities step:

$$x_{k+1} = (\beta W^T W + B^T B)^{-1}(B^T(b - F[x_k]) - \beta W^T W x_k) \tag{9.9}$$

If the difference between the approximate Jacobian and the Jacobian of the Newton iteration of 9.8 is small enough, i.e., if:

$$||(\beta W^T W + B^T B) - (\beta W^T W + B^T J(x_k))|| = ||B^T(B - J(x_k))|| < \delta_1 \tag{9.10}$$

then theorem 9.1 about errors in the Jacobian holds and the iteration will converge to $x^\dagger$.

Now assume that we have found the solution of the near-by problem 9.6, $x^\dagger$. It is important to know how close this solution is to the real solution x^*. In order to determine this we notice that:

$$g(x^\dagger) - g(x^*) = g(x^\dagger) = \beta W^T W x^\dagger + J(x^\dagger)^T (F[x^\dagger] - b) \tag{9.11}$$

Adding and subtracting $B^T(F[x^\dagger] - b)$ gives:

$$\beta W^T W x^\dagger + B^T(F[x^\dagger] - b) + (J(x^\dagger)^T - B^T)(F[x^\dagger] - b) = \tag{9.12}$$

$$(J(x^\dagger)^T - B^T)(F[x^\dagger] - b)$$

Using the fundamental theorem of calculus we can also write:

$$g(x^\dagger) - g(x^*) = H(\xi)(x^\dagger - x^*) \tag{9.13}$$

where ξ is a point between x^* and $x^\dagger$, and $H(\xi)$ is the Jacobian. Assuming that the Jacobian $H(x)$ is invertible we can write:

$$H(\xi)^{-1}(g(x^\dagger) - g(x^*)) = x^\dagger - x^* \tag{9.14}$$

For simplicity assume that $W = I$. Differentiating 9.1, the Jacobian can be written as:

$$H(x) = \frac{\partial g}{\partial x} = \beta I + J(x)^T J(x) + \frac{\partial J}{\partial x}(F[x] - b) \tag{9.15}$$

For the analysis we assume that terms in the Jacobian which come from the misfit term, are positive (not necessary definite) and therefore its minimum eigenvalue is β (and $||H(\xi)^{-1}|| < \beta^{-1}$). If this is the case, we can substitute 9.12 in 9.14 and obtain:

$$||H(\xi)^{-1}(g(x^\dagger) - g(x^*))|| = ||H(\xi)^{-1}(J(x^\dagger)^T - B^T)(F[x^\dagger] - b)|| = ||x^\dagger - x^*|| \tag{9.16}$$

and therefore:

$$||x^{\dagger} - x^{*}|| \leq \beta^{-1} ||(J(x^{\dagger})^{T} - B^{T})|| \; ||(F[x^{\dagger}] - b)|| \tag{9.17}$$

This equation shows how bad the approximation can be. The approximation depends on two parts. First it depends on how well the matrix B approximates the sensitivities $J(x)$ and second it depends on the misfit and β. While the misfit is dictated from the problem, the approximation of B to $J(x)$ is in our control. In the next sections we discuss some of the possible approximations to the sensitivities.

The formulation of the near-by problem is satisfactory if the gradient direction is close to the near-by direction. In this case an actual reduction in the real nonlinear minimization function $\phi = \beta\phi_m + \phi_d$ can be obtained. However if these directions are not close then the direction of the near-by problem might not be a descent direction. In this case we know that the near-by problem is not so near-by, and it should be replaced by another problem or terminated. This is demonstrated in Figure 9.1.

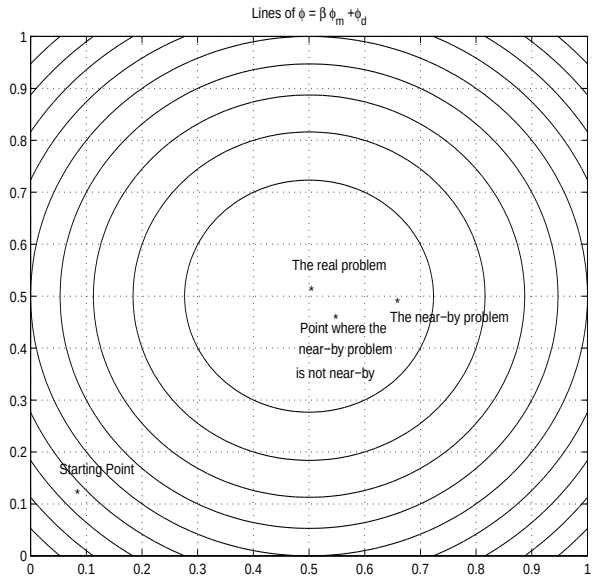

Figure 9.1: The real problem, the near-by problem, the starting point and the place that the near-by problem is not near-by any more.

Therefore after obtaining a direction δx we suggest to check if this direction actually decreases the value of the original objective function. If we start far from the solution, then at some stage, the update direction δx would not be satisfactory and even for a very small step length, a reduction in the function ϕ cannot be obtained. At this stage the near-by direction is not acceptable and the problem should be terminated.

Finally we write a general description of the approximate sensitivities algorithm:

Nonlinear Inversion Using Approximate Sensitivities

- Choose an initial model x_0 and approximate sensitivities B.
- Calculate $\phi^{old} = \beta||Wx_0||^2 + ||F[x_0] - b||^2$
- For $k = 1, 2...$ do:
  - Calculate a near-by step using 9.9.
  - Calculate $\phi^{new} = \beta||W(x + \delta x)||^2 + ||F[x + \delta x] - b||^2$
  - If $\phi^{new} \leq \phi^{old}$ set $x = x + \delta x$
  - Elseif $||\delta x|| < \delta$ terminate process.
  - Else $\delta x \rightarrow \delta x/2$
  - Check for convergence of the near-by problem.

Using the formulation of the near-by problem, we can use algorithms which were developed for nonlinear equations in order to solve 9.6. In the next sections we explore some of these methodologies.

9.2 The Cord and Shamanskii Updates

Maybe the most simple update is the cord method. For this method we assume that we can calculate $J(x_0)$ analytically or numerically. We therefore set $B = J(x_0)$ and solve

the system

$$f_c(x) = \beta W^T W x + J(x_0)^T (F[x] - b) = 0 \tag{9.18}$$

using the cord method, which means that the sensitivities in this case are kept constant.

$$x_{k+1} = (\beta W^T W + J(x_0)^T J(x_0))^{-1} (J(x_0)^T (b - F[x_k]) - \beta W^T W x_k) \tag{9.19}$$

Such an iteration was used by Li [1992] for the DC resistivity problem and Routh and Oldenburg [1996] for the nonlinear tomography problem. The difference between the application of the iteration suggested here and their work is that while their criterion for termination was based on the misfit, the criterion suggested here is based on the value of the global objective function.

The main problem of the cord method is that if we do not start very close to the solution, then the near-by problem is not so similar to the real problem and therefore after a few iterations the objective function, ϕ, can no longer be minimized. At this stage we face two options. If we are satisfied with the model and the misfit, we might terminate the iteration, however if we are not satisfied with the model, i.e., the misfit is too large or does not satisfy some criteria of convergence, then we can restart the process once again, and calculate a new matrix B based on the model which is obtained at the current iteration. This approach was first suggested by Shamanskii [1967] for the solution of nonlinear systems of equations. A similar strategy was proposed by Dosso [1990] for the solution of the MT problem. Thus, the cord process is repeated again, and therefore, the Shamanskii method could be viewed as a sequence of near-by problems such that each near-by problem is utilized to its full capacity. It is also possible to obtain the exact Tichonov solution using the Shamanskii method. In this case we continue the process until the function $f(x)$ is actually the gradient $g(x)$.

So far, the algorithms deal with the case of a constant regularization parameter. However in most cases we need an adaptive regularization parameter. Just like when the

sensitivities are known exactly, we suggest two methods to choose such a regularization parameter. The first method is the cooling method. In this case we follow the cooling algorithm in Chapter 8, however we substitute the approximate sensitivities in the place of the exact sensitivities. The second possibility is to use the GCV to find the regularization parameter. In this case we substitute the approximate sensitivities in place of the exact sensitivities in the nonlinear GCV algorithm.

9.3 Secant-Type Update

In the last section we discussed a constant near-by problem, i.e., the approximate sensitivities B are kept constant and the function $f(x)$ does not change until we cannot carry the nonlinear iteration any longer. However the near-by problem is not really the problem of interest and therefore, given new information, we might want to update the near-by problem. This is the idea behind the secant update for nonlinear ill-posed problems.

The application of secant updates to nonlinear systems of equations is not new (Dennis *et al.* [1989], Kelley [1995], Dennis and Schnabel [1996]). However applying secant updates to nonlinear inverse problems has been done only in a few cases (Zhang *et al.* [1995]). The main idea of the secant method is to obtain a better estimate for the sensitivities $J(x)$ by looking at the models which were calculated in the iteration process.

In order to explain the basic idea we first consider the one-dimensional case. Assume we have a function of one unknown $s(x)$ and we want to find the solution of $s(x) = 0$. In the first iteration we choose x_0 and calculate $s(x_0)$ and $s'(x_0)$. Then using Newton or some other iteration we calculate a point x_1 and the function at that point $s(x_1)$. In order to continue the iteration we would need the derivative $s'(x_1)$. Newton suggested approximating the derivative by:

$$s'(x_1) \approx \tilde{s}'(x) = \frac{s(x_1) - s(x_0)}{x_1 - x_0} \tag{9.20}$$

Thus the approximate derivatives obey the secant equation

$$\tilde{s}'(x)(x_1 - x_0) = \tilde{s}'(x)\delta x = s(x_1) - s(x_0) = \delta s \tag{9.21}$$

In the one-dimensional case, it has been proved that this iteration converges to the solution of $s(x) = 0$.

The extension of this iteration to more than one dimension is not straight-forward. Assume that we start with a model x_0. For this point we calculate the forward modelling $F[x_0]$ and the exact sensitivity $J(x_0)$. Using this sensitivity we proceed and calculate x_1 and $F(x_1)$. Our goal is to find an approximation to $J(x_1)$, call this approximation B_1. In parallel to the one-dimensional case we want the new sensitivity to obey the secant equation:

$$B_1(x_1 - x_0) = F[x_1] - F[x_0] \tag{9.22}$$

The matrix B_1 is of size $N \times M$, however we have only one vector equation to satisfy and therefore we have another $N - 1$ degrees of freedom. It was suggested by Broyden [1965] to choose a matrix B_1 such that it is as close as possible to the current sensitivities, $J(x_0)$. i.e. B_1 should minimize:

$$||B_1 - J(x_0)||_2^2 = ||\Delta J||_2^2 \tag{9.23}$$

$$subject\ to \quad B_1(x_1 - x_0) = F[x_1] - F[x_0]$$

Adding and subtracting $J(x_0)(x_1 - x_0)$ from the equation and substituting $\Delta J = B_1 - J(x_0)$, the constraints with ΔJ can be written as:

$$\Delta J(x_1 - x_0) - F[x_1] + F[x_0] + J(x_0)(x_1 - x_0) = 0$$

In order to find B_1 we take the route of constrained minimization which is different from Broyden's paper. We define the Lagrangian:

$$\mathcal{L} = \frac{1}{2}||\Delta J||_2^2 + \lambda^T(\Delta J(x_1 - x_0) - F[x_1] + F[x_0] + J(x_0)(x_1 - x_0)) \tag{9.24}$$

In order to find ΔJ we differentiate the Lagrangian and equate to zero. This yields:

$$\Delta J + \lambda(x_1 - x_0)^T = 0 \tag{9.25}$$

$$\Delta J(x_1 - x_0) - F[x_1] + F[x_0] + J(x_0)(x_1 - x_0) = 0$$

Multiplying the first equation by $(x_1 - x_0)$ gives:

$$0 = \Delta J(x_1 - x_0) + \lambda(x_1 - x_0)^T(x_1 - x_0) = F[x_1] - F[x_0] - J(x_0)(x_1 - x_0) + \lambda(x_1 - x_0)^T(x_1 - x_0)$$

where the last equality due to the second equation in 9.25. From this equation we can find that:

$$\lambda = \frac{F[x_1] - F[x_0] - J(x_0)(x_1 - x_0)}{(x_1 - x_0)^T(x_1 - x_0)}$$

Substituting back in the first equation of 9.25 gives:

$$\Delta J = \frac{F[x_1] - F[x_0] - J(x_0)(x_1 - x_0)}{(x_1 - x_0)^T(x_1 - x_0)}(x_1 - x_0)^T \tag{9.26}$$

Equation 9.26 is the famous Broyden's update to the sensitivities.

We can now combine this methodology with the near-by problem methodology which we introduced in the previous section. Using this update, at each iteration we can have an estimate of the sensitivities $J(x_k)$ and thus we define in each iteration a different near-by problem. We then carry one iteration of this near-by problem to obtain a new near-by problem.

There are a few potential problems with this method. First, since we do not have a global function to solve, because the near-by problem is constantly changing, we cannot say *a priori* to which solution the procedure will converge and to what distance from the Tichonov model. Hopefully the secant approximation yields a better approximation to the sensitivities at x^* than $J(x_0)$ and therefore the solution is a better one. Second, there is a potential danger of not converging at all. In this work, termination of the

secant iteration occurred if it had not converged to some model after a fixed number of iterations. These two problems emphasise the third problem which is in my opinion the biggest problem when using such algorithm. The end result depends on the minimization process itself, since there is no global objective function to minimize or an equation which we solve, the end solution is based on the path of minimization. It is therefore impossible to compare results of two secant algorithms, and it is hard to conclude if some features in the model are there due to the data and the objective function, or due to the minimization process.

Although the above problems are definitely significant, secant updates could still achieve a model which makes the objective function small and fits the data. The secant method may still give reasonable results where the constant approximate sensitivities may fail.

Chapter 10

Applications of Nonlinear Inverse Problems

In Chapters 7-9 we discussed the formulation and strategies for the solution of nonlinear inverse problems. In this chapter we compare the different strategies when applied to two generic type of problems, the nonlinear gravity problem and the magnetotellurics inverse problem. Unlike the linear case our measure of efficiency is not based on the number of floating point operations. Different problems have very different characteristics, while in one problem calculating the forward modelling is not a problem, in the other forward modelling is the most expensive part. We therefore compare between the efficiency of the methods by counting the number of forward modellings, sensitivity calculations and matrix inversions which needed in order to obtain convergence. The quality of the solution is judged by the model norm and the misfit (just like in the linear case).

Each example is tested using eight different algorithms with various noise levels. The methods we are going to use are:

- CGLS+GCV - Solving each linearized system for x_{k+1} using the CGLS and the GCV as a stopping criterion (Section 8.2.2).
- Hybrid+GCV - Solving each linearized system for x_{k+1} using hybrid Krylov method with GCV criterion (Section 8.2.3).
- Full-Space+GCV - Solving each linearized system for x_{k+1} using Tichonov regularization with GCV criterion (Section 8.2.1).

- Cooling - Starting with large regularization parameter and decreasing it slowly. Recall that this method must have a target misfit (Section 8.1).
- TSM - Using the two stage method for regularization parameter. Recall that this method must have a target misfit (section 7.4.2).
- Cord+GCV+Hybrid - Fixing the sensitivities and using the GCV+Hybrid method on each linearized iteration (Section 9.2).
- Shamanskii+GCV+Hybrid - Fixing the sensitivities for a number of iterations, and using the GCV+Hybrid method on each linearized iteration (Section 9.2).
- Secant+GCV+Hybrid - Updating the sensitivities using Broyden's method, and using the GCV+Hybrid method on each linearized iteration (Section 9.2).

Trust region methods are not explored here since, as explained in Section 7.3.3, the special structure of the problem is such that TR methods does not give substantial advantage on the DGN.

10.1 The Gravity Interface Problem

The gravity interface problem was described in Chapters 1 and 6, and we use the same discretization and linearization procedures which we described in Chapter 6. Recall that the forward modeling is given by:

$$\delta g_j = \int\int_{\mathcal{D}} \frac{1}{r_h(x,y,h(x,y);x_j,y_j)} - \frac{1}{r_m(x,y,m(x,y);x_j,y_j)} dxdy \qquad (10.1)$$

with:

$$r_h(x,y,h(x,y);x_j,y_j) = \sqrt{(x-x_j)^2+(y-y_j)^2+h^2}$$

and

$$r_m(x,y,m(x,y);x_j,y_j) = \sqrt{(x-x_j)^2+(y-y_j)^2+(h+m)^2}$$

and the sensitivities are:

$$J(m)(.) = \int\int_{\mathcal{D}} \frac{(h(x,y)+m(x,y))(.)dxdy}{((x-x_j)^2+(y-y_j)^2+(h(x,y)+m(x,y))^2)^{\frac{3}{2}}} \tag{10.2}$$

The integrals are discretized using the midpoint rule. The $100m \times 100m$ square domain is divided into 49×49 grid points and we assume we have 30×30 data points. We add 5% and 10% Gaussian noise to the data. The model and the data are plotted in Figure 10.1.

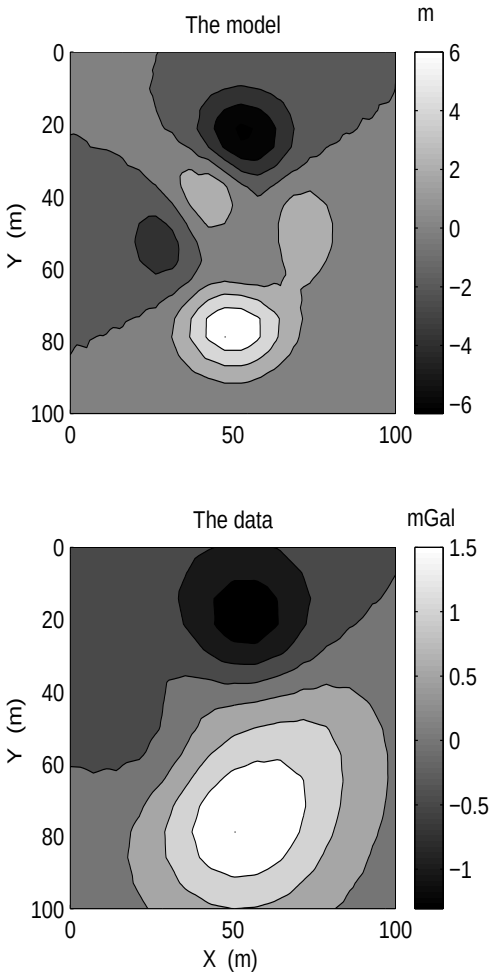

Figure 10.1: The model and data used for nonlinear inversion

Our goal is to test the different algorithms using different conditions. We therefore

Method	$\|\|F[x]-b\|\|$	ϕ_m	FM	Sens	Matrix Inversions
CGLS+GCV	6.47E-2	34.43	8	5	5
Hybrid+GCV	6.61E-2	32.99	8	4	4
Full-space+GCV	6.75E-2	31.45	7	4	4
Cooling	6.73E-2	31.48	72	63	63
TSM	6.73E-2	32.21	20	5	5
Cord+GCV+Hybrid	6.53E-2	35.45	11	1	7
Shamanskii+GCV+Hybrid	6.65E-2	32.87	13	3	9
Secant+GCV+Hybrid	6.69E-3	35.21	13	1	8

Table 10.1: Experiment one: Comparison between different methods for 5% noise. True misfit is $6.72E-2$, real model norm 36.87.

design three tests. In the first test we use as a background model a half space which has the value of the true background (20 meters). This choice gives an initial misfit of 1.41 when the final misfit is 0.067 in the 5% noise case. In this case the initial model is relatively close to the end result. In the second case we assume we have a wrong background and therefore start with a half-space which is far from the initial model (30 meters instead of the 20). In this case the initial misfit is 9.56, which is obviously far from the end result. In the third experiment we use the correct reference model, but start with a far starting model which is a flat half-space at depth 10 meters lower than the true reference. In order to carry the inversion we choose a weighting function $W = -0.01\nabla^2 + I$, where the ∇^2 operator does not contain boundary conditions.

We start with the first experiment and use the different methods to invert the system. A comparison of these methods is in Table 10.1 and 10.2. The final models which were obtained using these methods in the 10% noise case are plotted in Figures 10.2 and 10.3.

In Figures 10.4, 10.5 and 10.6 we view the iteration progress for the GCV+CGLS case, the GCV+full space case and the TSM. Note that in the different variants of the GCV, the model norm is monotonically increasing and the misfit is monotonically decreasing

Method	$\|\|F[x]-b\|\|$	ϕ_m	FM	Sens	Matrix Inversions
CGLS+GCV	1.38E-1	36.43	15	10	10
Hybrid+GCV	1.55E-1	31.45	8	6	6
Full-space+GCV	1.55E-1	31.44	7	6	6
Cooling	1.55E-1	31.44	69	58	58
TSM	1.55E-1	32.31	20	5	5
Cord+GCV+Hybrid	1.54E-1	32.29	13	1	5
Shamanskii+GCV+Hybrid	1.55E-1	31.46	17	3	17
Secant+GCV+Hybrid	1.55E-1	31.46	17	5	17

Table 10.2: Experiment one: Comparison between different methods for 10% noise. True misfit is $1.53E-1$, real model norm 36.87.

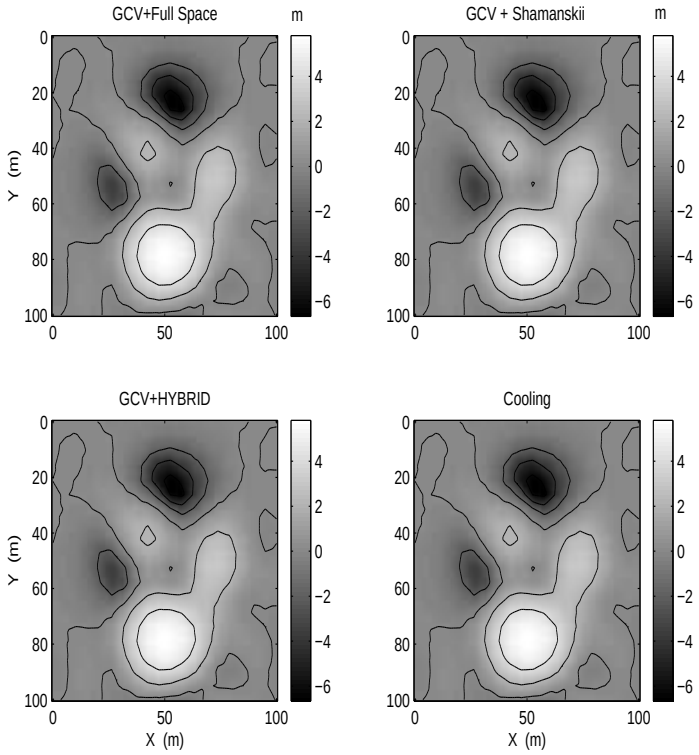

Figure 10.2: Experiment one: results of Full-space+GCV, Hybrid+GCV, Cooling and Shamanskii inversions. Noise level 10%.

to the final value while in the TSM the model norm first increased above its final value and then decreased back to its final value. This behaviour of the TSM is predicted when looking in Figure 7.3. Note also that hybrid GCV, full-space GCV, cooling and Shamanskii methods give basically the same results (Figures 10.3 and 10.3. This result is expected since the same quantity is minimized. The results of the GCV+CGLS, TSM, cord and secant methods are somewhat different. This result is also expected since the minimization problem is different for each of these problems.

In comparing the different inversions we notice the following observations from this first experiment:

- Noise was estimated quite accurately using the three variants of GCV (full, hybrid

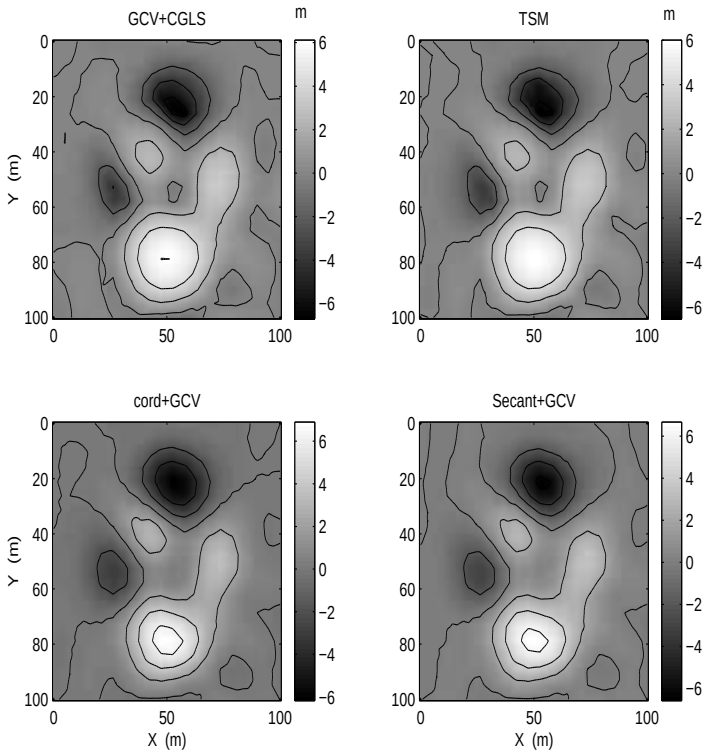

Figure 10.3: Experiment one: results of TSM, CGLS+GCV, Cord+GCV+Hybrid, Cord+GCV+Hybrid inversions. Noise level 10%.

and subspace).

- Noise was estimated quite accurately using approximate sensitivities. Basically the approximate sensitivities did not influence noise estimation.
- The quality of the solution in terms of model norm versus misfit is similar in the cooling and the GCV algorithms and the Shamanskii method. However the models which are obtained using cord and secant methods are somewhat different and have higher model norm.
- Although the cord and secant methods give different results, the models can be considered to be reasonable.
- TSM gives reasonable models but with model norm slightly higher than cooling or GCV for the same misfit.
- The results emphasise that the use of global algorithms which minimize a Tichonov

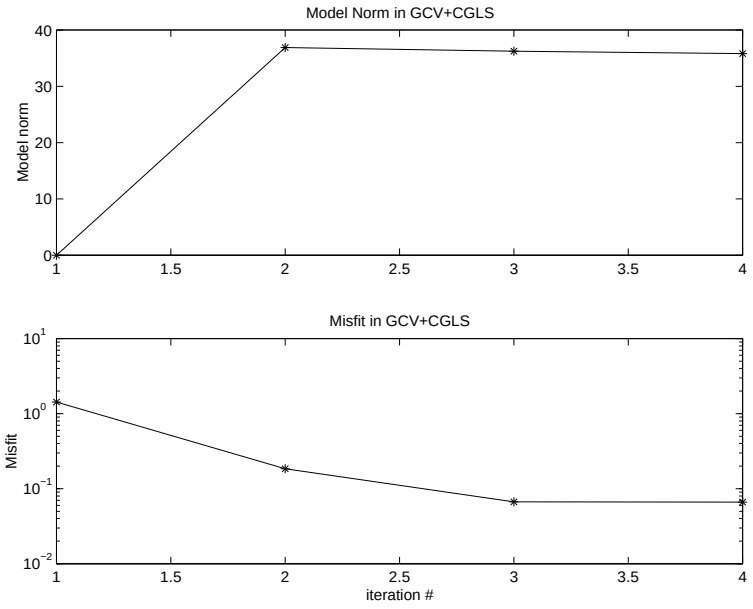

Figure 10.4: Model norm and misfit as a function of iteration in the CGLS+GCV

objective function, have an advantage over the inexact algorithms such as the cord and secant, where the minimization result depends on the path of minimization.

In the first experiment we have performed, most methods did well. The experiment is somewhat easy since the initial model is close to the final model. In order to demonstrate, we calculate the difference between the final sensitivities of the full space GCV model and the initial model is:

$$||J(x_0) - J(x^*)||_2 = 0.0098$$

Therefore it is not a surprise that all the methods did so well. In the next experiment we take the wrong base model and try to recover the model using this reference model. The base model is a model which has a mean depth of 30 meters instead of 20 meters.

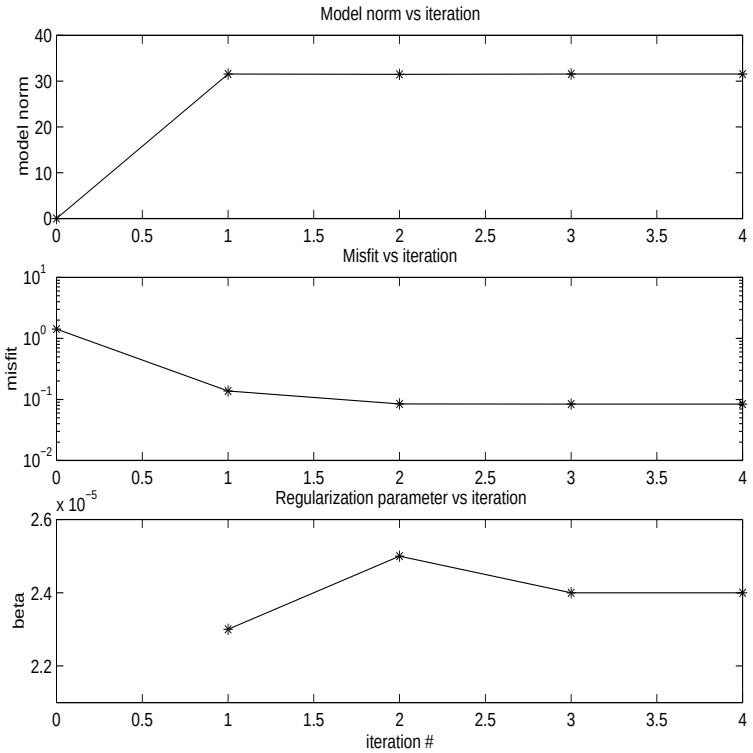

Figure 10.5: Model norm misfit and the regularization parameter as a function of iteration in the full space GCV

For this case:

$$||J(x_0) - J(x^*)||_2 = 0.8$$

The experiment is done for the 5% noise and the results are shown in Table 10.3, and the models obtained using different methods are plotted in Figures 10.7 and 10.8.

All the algorithms have converged, and the models are satisfactory from an interpretation point of view. From a mathematical point of view, this example shows that the algorithms proposed in Chapter 8 and 9 are robust and work well even when the reference model is notoriously bad.

As a last example we try a more difficult test. We start from a model which is not the reference model. The model is a surface at depth 10 meters lower than the real reference,

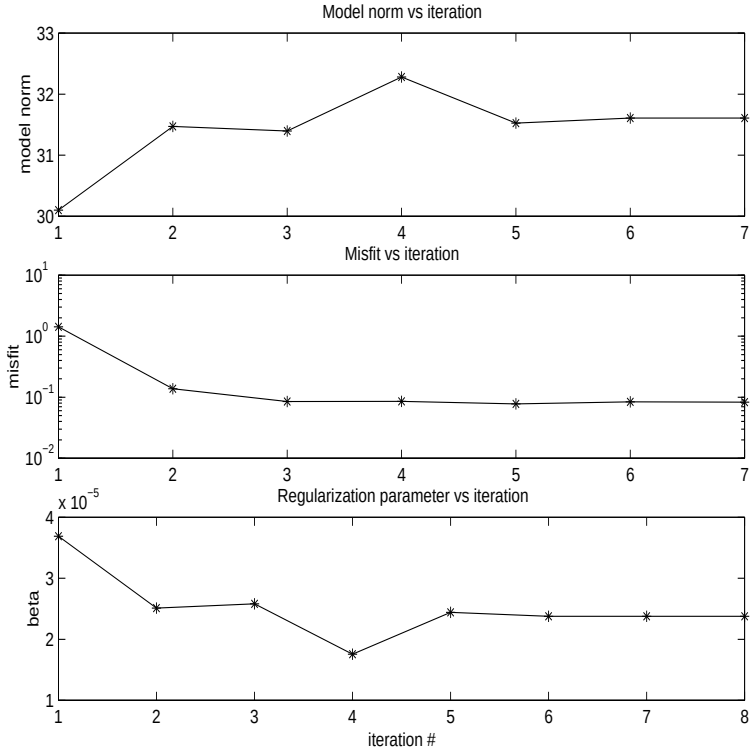

Figure 10.6: Model norm misfit and the regularization parameter as a function of iteration in the TSM

Method	$\|\|F[x]-b\|\|$	ϕ_m	FM	Sens	Matrix Inversions
CGLS+GCV	3.6E-1	118.3	6	4	4
Hybrid+GCV	3.8E-1	108.5	5	3	3
Full space+GCV	3.8E-1	108.0	6	4	4
Cooling	3.8E-1	108.0	38	31	31
TSM	3.8E-1	108.6	20	4	4
Cord+GCV+Hybrid	3.8E-1	110.1	12	1	5
Shamanskii+GCV+Hybrid	3.8E-1	108.0	9	2	7
Secant+GCV+Hybrid	3.8E-1	108.4	8	1	5

Table 10.3: Experiment two: Comparison between different methods for 5% noise. Reference model is far from the real model. True misfit is $3.7E-1$, true model norm 113.5.

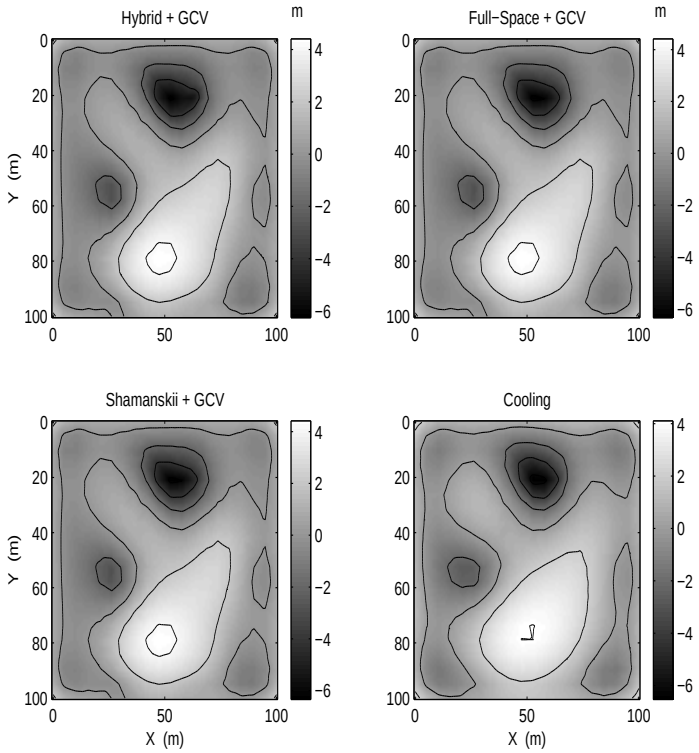

Figure 10.7: Experiment two: 2-D Nonlinear Gravity Inversion. The reference model is far from the true reference.

i.e., $x_0 = 10$. The noise level is 3%. In this case the difference between the sensitivity of the true model and the starting model is:

$$||J(x_0) - J(x)|| = 0.3$$

This experiment has no geophysical meaning, since, as stated in Chapter 7, if the *a priori* information states that the model is similar to a model x_1 other than the reference model x_0, we should reset the reference model x_0 to x_1, which means, in our example, to set the reference depth to 30 meters instead of the original 20 (just as in the experiment two). However from a mathematical prespective it is a good example to test the stability of the algorithms. The results are shown in Table 10.4 and in Figures 10.9 and 10.10.

We summarize the results of the last two experiments:

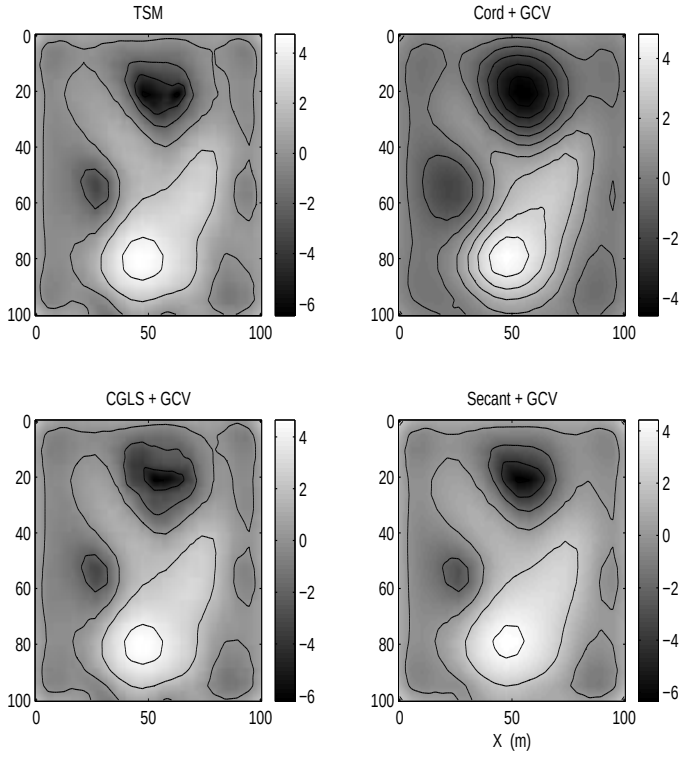

Figure 10.8: Experiment two: 2-D Nonlinear Gravity Inversion. The reference model is far from the true reference.

Method	$\|\|F[x]-b\|\|$	ϕ_m	FM	Sens	Matrix Inversions
CGLS+GCV	5.0E-2	33.3	21	16	16
Hybrid+GCV	5.6E-2	31.1	10	8	8
Full space+GCV	5.6E-2	31.0	9	7	7
Cooling	5.6E-2	31.0	41	35	35
TSM	5.6E-2	31.4	29	9	9
Cord+GCV+Hybrid	9.3E-2	30.8	15	1	10
Shamanskii+GCV+Hybrid	5.7E-2	31.1	14	3	10
Secant+GCV+Hybrid	9.1E-2	31.1	12	1	9

Table 10.4: Experiment three: Comparison between different methods for 3% noise. Starting model is far from the real model. True misfit is $5.5E-2$, true model norm 36.87.

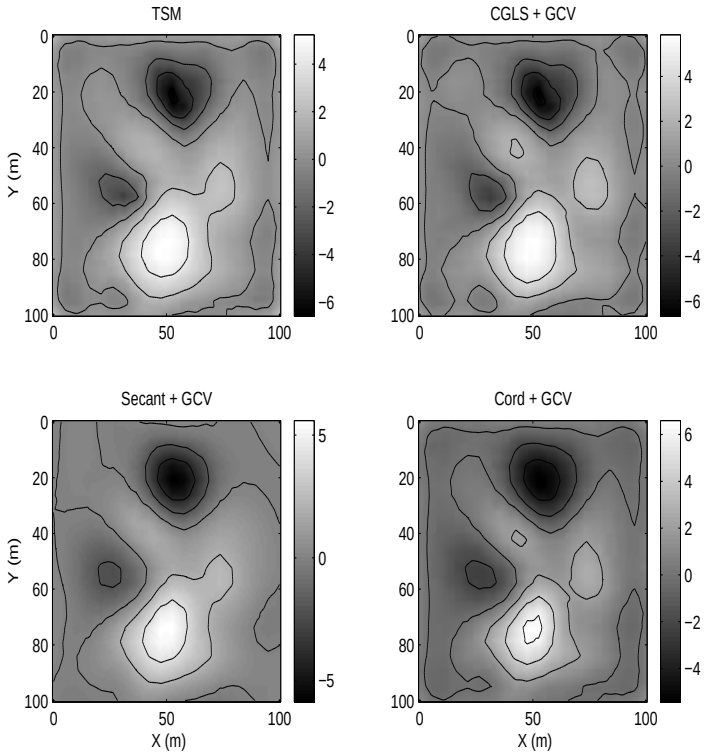

Figure 10.9: Experiment three: 2-D Nonlinear Gravity Inversion, starting model is far from the reference model

- All methods did well when the reference model is far from the true reference.
- Cooling, GCV in full space, hybrid GCV, TSM and Shamanskii methods did well even when starting model is not reasonable.
- Cooling, GCV in full space, hybrid GCV and Shamanskii methods produce very similar results.
- Methods which depend on the path of minimization such as cord, secant and CGLS+GCV, obtained a reasonable model but did not predict the misfit very well.

We summarize the nonlinear gravity problem with a discussion about the efficiency

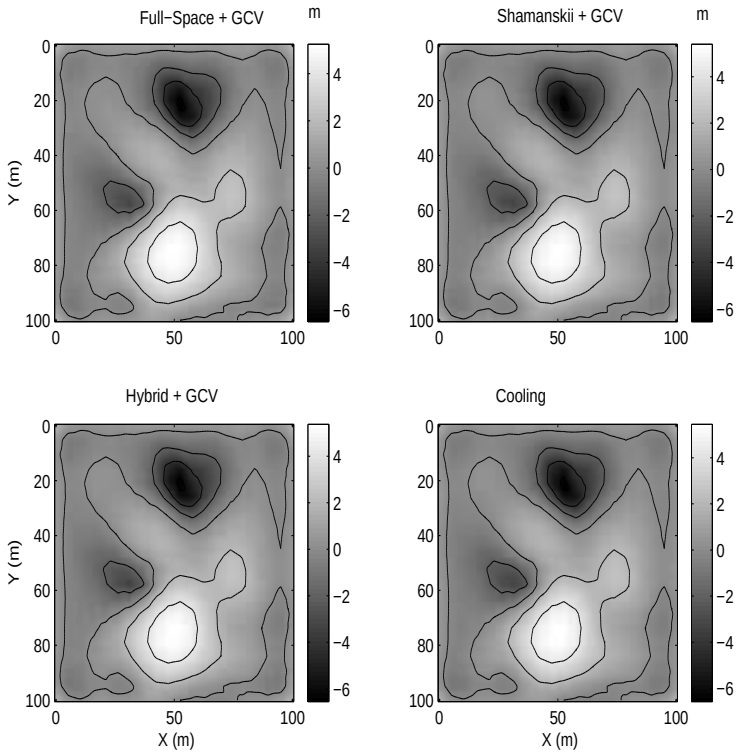

Figure 10.10: Experiment three: 2-D Nonlinear Gravity Inversion, starting model is far from the reference model

of the different methods. In order to determine which of the algorithms is the most efficient for this problem a comparison of the different possible bottle-necks is needed. For this size of problem, the forward modeling takes around 50 seconds (using SPARC 10 workstation), the sensitivities take around 180 seconds and inverting a matrix of that size using CGLS takes around 700 seconds. It is clear from this rough count that we would prefer an algorithm which performs fewer matrix inversions and sensitivities and does more forward modeling. In this case the GCV with CGLS is significantly more efficient, since for every direction only one matrix inversion is performed and only one sensitivity matrix is calculated.

10.2 The Magnetotelluric Problem

10.2.1 Equations and Synthetic Example

A very different and generic type of electromagnetic inverse problem arises from the magnetotelluric (MT) experiments. In this experiment we measure the response of the earth due to a plane wave impinging upon the earth's surface. Recall from Chapter 1 that assuming that the earth is layered, the electric field, E, is given by the equations:

$$\frac{d^2E}{dz^2} = i\omega\mu_0\sigma(z)E \tag{10.3}$$

With boundary conditions:

$$E(\infty) = 0$$

$$E(0) = 1$$

Where ω is the angular frequency, $\sigma(z)$ is the one-dimensional conductivity structure and μ_0 is the magnetic permeability, which is assumed to be constant. Equation 10.3 is the governing equation for the MT experiment. The data for this experiment are given by:

$$c_0(\omega, \sigma(z)) = -\frac{E(z=0,\omega)}{\partial_z E(z=0,\omega)} \tag{10.4}$$

Our goal is to recover the conductivity profile $\sigma(z)$ from the complex measurements c_0. It is common to present the complex data, $c(\omega)$, by its phase and by the apparent conductivity which is defined as:

$$\sigma_a(\omega) = \mu_0\omega|c(\omega)|^2 \tag{10.5}$$

Although the sensitivities for this problem can be found analytically (Oldenburg [1979]), in this work we have used numerical differentiation in order to obtain the sensitivities. Thus the only thing needed for the solution of this problem is a robust forward modelling program. The forward modelling is performed through the propagator matrix

formulation (Ward and Hohmann [1988]). The one-dimensional earth is divided into M layers with constant conductivity in each layer. The differential equation 10.3 is solved in each layer, with the conditions of continuity of the field E and its derivative E' at each layer interface. This leads to the recursive relation for $C(z,\omega) = -E(z,\omega)/E'(z,\omega)$ which constitutes the forward modelling:

$$C(z_{j-1},\omega) = \frac{1}{k_j}\frac{tanh(k_j h_j) + k_j C(z_j,\omega)}{1 + k_j C(z_j,\omega) tanh(k_j h_j)} \qquad j = M...1 \tag{10.6}$$

where

$$k_j = \frac{1}{\sqrt{2}}(1+i)\sqrt{\omega\mu_0\sigma_j}$$

h_j is the thickness of the j^{th} layer and σ_j is the conductivity of the j^{th} layer. The data are given simply by $c_0(\omega) = C(0,\omega)$.

In order to carry out forward modelling and inversion we pick a model made from 64 layers. The thickness of the layers increases quadratically with depth such that $z_i = \zeta_i^2$ and ζ is linearly spaced. The conductivity model is taken from the book by Whittall and Oldenburg [1992]. The model is plotted in Figure 10.11. The data in terms of phase and apparent conductivity are plotted in Figure 10.12. In the first stage we add 5% noise to the data. In order to carry out the inversion we need to pick a reference model and

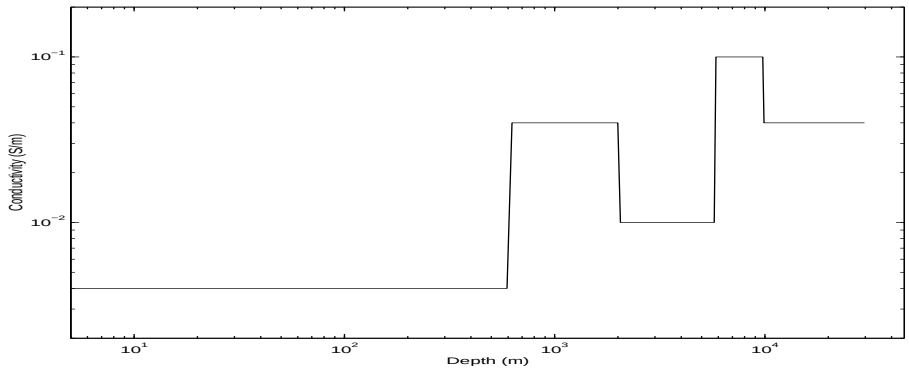

Figure 10.11: The conductivity model used for the MT experiment

a weighting matrix. In this work we use the operator $0.001I - \nabla^2$ where ∇^2 does not contain boundary conditions, and we set the reference model to have the value of the background, $x_0 = 0.04 \quad S/m$. The model tend to span many order of magnitude and therefore we do not invert for the conductivity but rather for the log of the conductivity. The results of this inversion are summarized in Table 10.5. In order to demonstrate robustness of our algorithm we repeat the same experiment, but this time with noise level set to 0.5%. The results of this experiment are shown in Table 10.6 and plotted in Figures 10.13 and 10.14.

From these two experiments we conclude:

- All the methods which do not use approximate sensitivities predict the misfit and obtained reasonable models.
- All the methods which do not use approximate sensitivities work well even for very

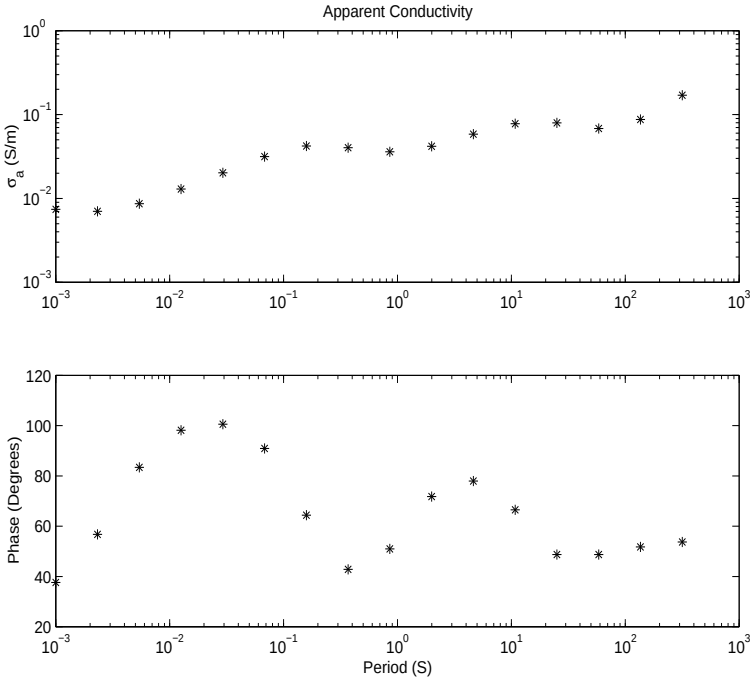

Figure 10.12: The MT data for the given conductivity model.

Method	$\|\|F[x]-b\|\|$	ϕ_m	FM	Sens	Matrix Inversions
Full space+GCV	3.0E-1	6.7E-1	6	4	22
CGLS+GCV	2.9E-1	7.4E-1	9	7	7
Hybrid+GCV	3.3E-1	5.9e-1	6	4	4
Cooling	2.9E-1	7.2E-1	17	13	13
TSM	3.2E-1	6.0E-1	20	4	4
Cord+GCV+Full-space	6.3E-1	1.4	25	1	19
Shamanskii+GCV+Full-Space	3.0E-1	6.7E-1	10	3	7
Secant+GCV+Hybrid	9.1E-1	1.3	8	1	5

Table 10.5: Inversion of MT data for the 5% noise case. True misfit is $3.2E-1$, true model norm is 1.58.

Method	$\|\|F[x]-b\|\|$	ϕ_m	FM	Sens	Matrix Inversions
Full space+GCV	4.2E-2	9.3E-1	10	6	76
CGLS+GCV	3.9E-2	1.01	10	8	8
Hybrid+GCV	4.1E-2	9.3E-1	9	5	5
Cooling	4.3E-2	9.2E-1	63	45	45
TSM	4.2E-2	1.1	35	9	9
Cord+GCV	3.1E-1	2.1	9	1	6
Shamanskii+GCV	4.1E-2	9.3E-1	19	3	10
Secant+GCV	4.1E-1	1.8	6	1	4

Table 10.6: Inversion of MT data for the 0.5% noise case. The true misfit was 0.04 and the true model norm is 1.58.

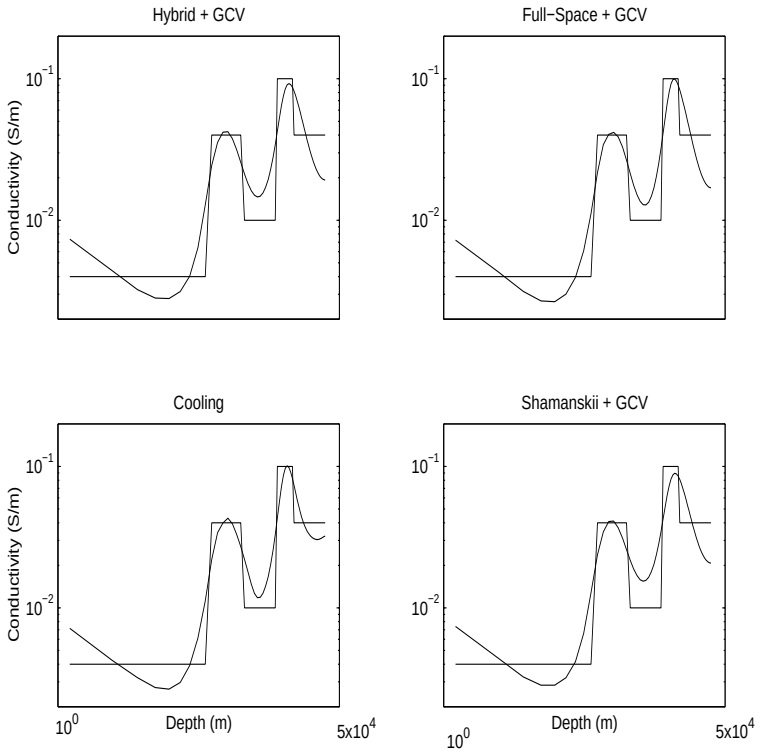

Figure 10.13: Result of full-space GCV, Shamanskii, hybrid GCV and cooling MT inversions.

low noise levels, where the nonlinear part of the objective function is dominant.

- Models obtained by approximate sensitivities did not fit the data to the right extent and have a larger model norm. However the inverted models have the "right flavour" to them, and they simulate the correctly inverted models.

Notice that the approximate sensitivities fail to find a model which predict the data. The main reason is revealed when observing the norm of the difference between the sensitivities of the half-space and the sensitivities of the final model. For this case:

$$||J(x_0) - J(x^*)|| = 1.13$$

therefore we cannot expect the models from the approximate sensitivities to converge to the true model.

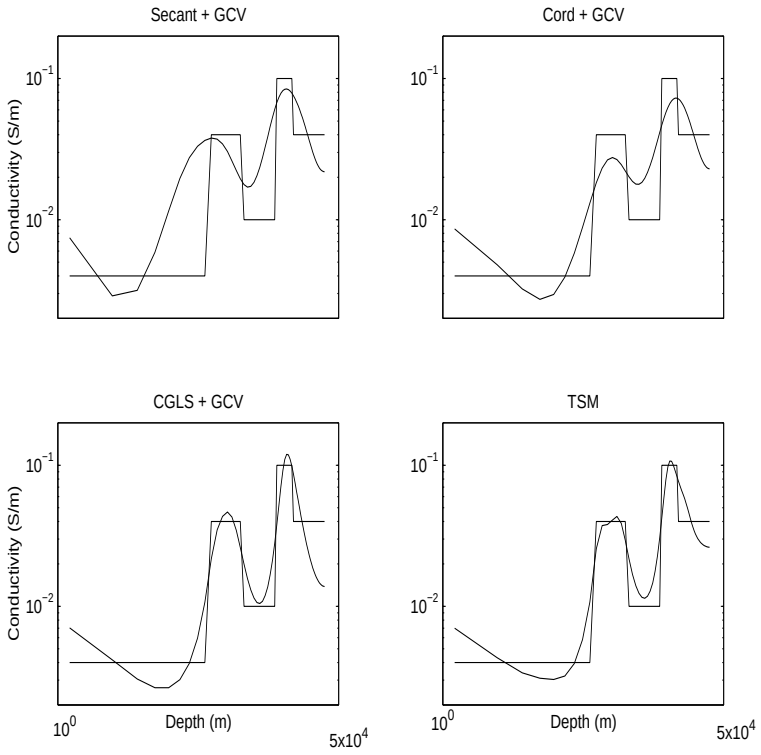

Figure 10.14: Result of secant GCV, cord GCV, TSM and CGLS GCV MT inversions.

As a last experiment with the synthetic data, we pick a starting model which has sensitivities which are close to the final ones:

$$||J(x_0) - J(x^*)|| = 0.12$$

The model is plotted in Figure 10.15 (top left). We now carry out three inversions. Two using the cord and the secant method and one with the full-space GCV for comparison. The results of the inversions are plotted in Figure 10.15 and the numbers are in Table 10.7. The results emphasize the importance of a good starting point for the use of approximate sensitivities.

Method	$\|\|F[x]-b\|\|$	ϕ_m	FM	Sens	Matrix Inversions
Full space+GCV	2.7E-1	1.7E-1	6	3	3
Cord+GCV+Full-Space	2.5E-1	3.1E-1	8	1	6
Secant+GCV+Full-Space	2.7E-1	2.6E-1	8	1	5

Table 10.7: Inversion of MT data with approximate sensitivities. The starting model is close to the final model. The true misfit is 0.27.

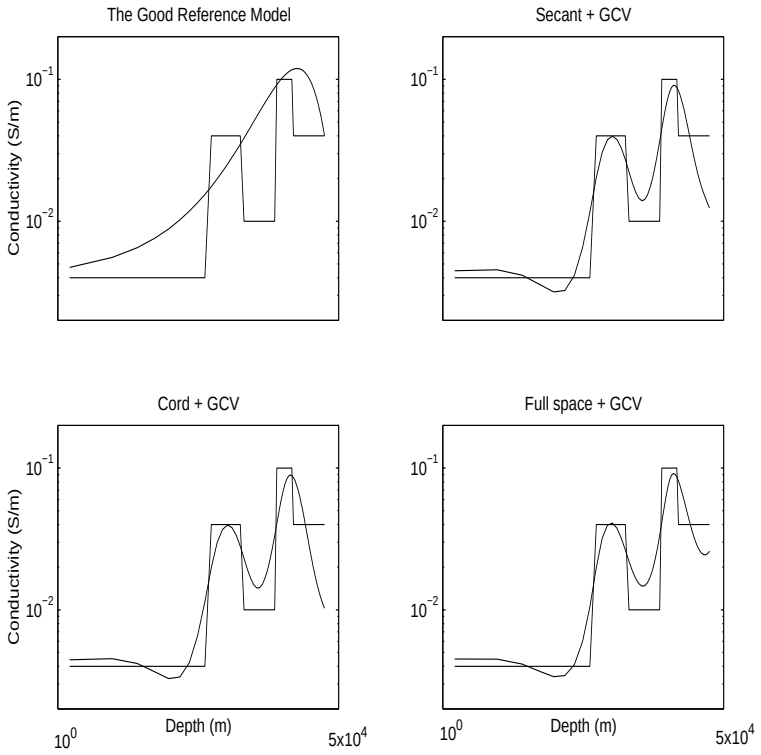

Figure 10.15: Result of approximate sensitivities MT inversions when starting model is close to the final model.

10.2.2 Field Example and Conclusions

As a final experiment, we invert field data taken from Young *et al* [1988]. The data are plotted in Figure 10.16.

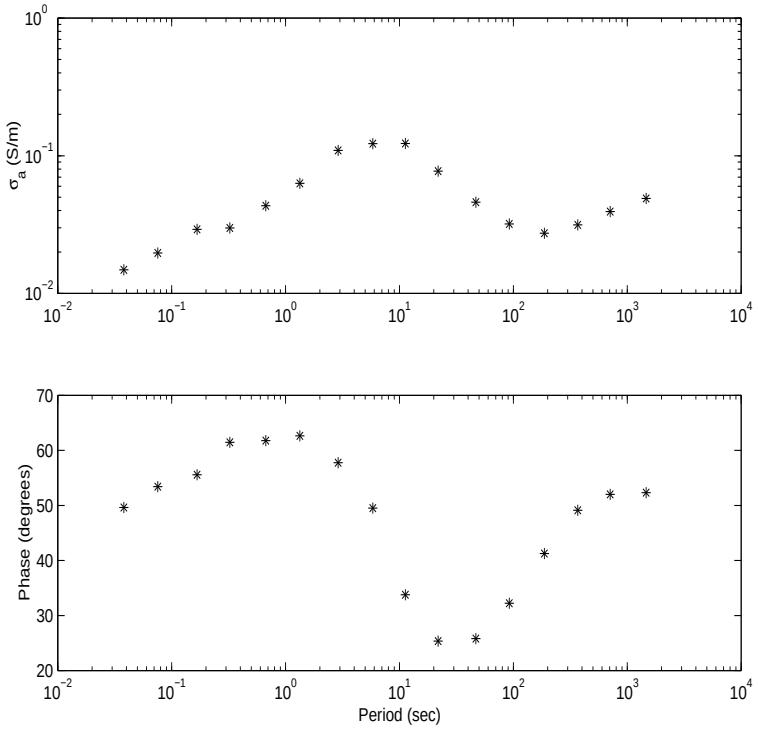

Figure 10.16: Field MT data.

Although the standard deviations of the data are given, we ignore them for the moment and invert these data using the GCV first. We use the same discretization, objective function and reference model as the synthetic example. We repeat the inversion of the same data using the cooling method (the target misfit is calculated using the standard deviations) and the TSM, and compare between the results. The results of the inversion are plotted in Figure 10.17 and summarized in Table 10.8. Again we see that GCV predicted the right misfit well and produced reasonable models which matched the cooling method and the TSM. The TSM gave a model with a slightly higher model norm

Method	$\|\|F[x]-b\|\|$	ϕ_m	FM	Sens	Matrix Inversions
Full space+GCV	3.8E-1	5.5E-1	12	10	85
CGLS+GCV	3.9E-1	5.9E-1	10	8	8
TSM	3.8E-1	5.9E-1	40	10	10
Cooling	3.7E-1	5.6E-1	26	19	19

Table 10.8: Inversion of MT field data. Predicted misfit is $3.8E-1$.

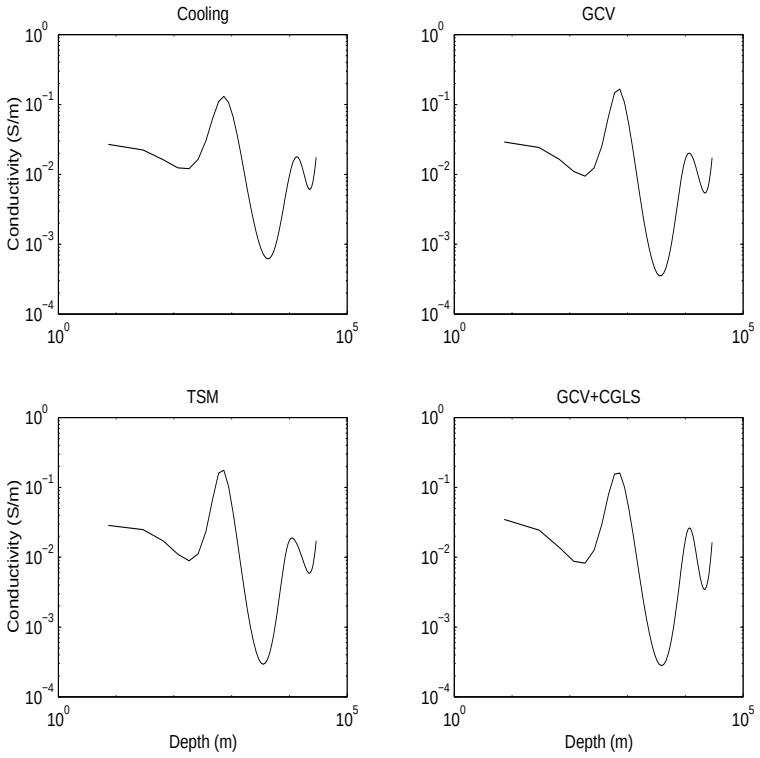

Figure 10.17: Inversion of the MT field data.

than the GCV for the same misfit, however the models look almost identical and from an interpretation point of view they are similar.

Finally in order to decide which method is the most efficient for the inversion, we checked the computational time of each of the stages in the inversion. For the 64 model parameters and 16 data points, it takes about 0.01 seconds to carry out full matrix

inversion or the SVD, 0.9 seconds for the forward modelling and 61 seconds for the calculation of the sensitivities. It is therefore clear that subspace and hybrid methods play no role in the improvement of computational time. The main hurdles are the sensitivities and the forward modelling. If approximate sensitivities yield good results then they would give the shortest computational time. However the problem is fairly nonlinear, and as shown before, in order to make the approximate sensitivities approach successful, we need a fairly good starting model. If such a starting model is not avaliable we need to take a method which yields good descent directions. The GCV based methods and the TSM seem to have such descent vectors. The advantage of the GCV methods is that they do not need a target misfit. When using the GCV, all three variants, full-space, hybrid and subspace, give good results. In numerical testing we found that in general the hybrid GCV is the most stable variant. I believe that the main reason is that when using the full-space GCV, it could happen that the "black box" minimization for the GCV function might fail in one specific iteration. In this case the model for this iteration might build structure which is hard to get rid of in later iterations. Hybrid methods, restrict the model to a "nicer" space and therefore even if in one iteration the GCV function is not fully minimized, the result of this iteration is still reasonable.

10.3 Summary

In this chapter we investigated nonlinear inversion methods for two types of problems, nonlinear gravity and magnetotellurics. As a conclusion to this section we summarize the selection of an inversion method for a nonlinear inverse problem, based on these two results. First for a new problem, where the behaviour of the nonlinearity is not known, I would start with the cooling method. The main reason is that the cooling method is a "safe". It is important to note once again that the starting model and the reference model

should be identical. If this is done then the starting model norm is zero and the problem is almost quadratic in the first iterations, which makes convergence easy. Observing the cooling method, we could learn something about the nonlinearity of the problem, the reference model that we picked, and the computational time. If the problem is highly nonlinear, I would work with the GCV and a hybrid method, limiting the space size and ensuring to work only with "nice" vectors. In cases that the computational time of the cooling method is not satisfactory, or when the noise level is not known, we should use one of the GCV variants. The variant depends on the computational time of each of the three bottle necks, i.e., the sensitivities, forward modelling and matrix inversion. In cases where the matrix inversion is the biggest problem, the combination of GCV and CGLS is the most efficient. If the forward modelling is the main problem then hybrid or full space GCV are the best options, since they obtain a better direction. Finally if the sensitivities impose great difficulty, then the Shamanskii method performs better. Using approximate sensitivities such as the cord and secant methods are possible in certain cases. Special care should be taken in these cases to start from a relatively near-by point. If such a point is not avaliable, then it is possible that the reconstructed model does not fit the data but hopefully has the "flavour" of the exact inverted model.

Chapter 11

Summary and Future Work

The goal of this thesis was to review and develop new techniques for solving linear and nonlinear inverse problems. In so doing, the computational tools for solving inverse problems have been comprehensively studied.

First, in Chapters 2-6, linear inverse theory was dealt with. Linear inverse theory was formulated in Chapter 2. Chapter 3 reviewed the commonly-used Tichonov regularization as well as providing noise estimation techniques and, a new explanation for the L-curve technique. This explanation suggests a new choice for the point on the L-curve required to solve a linear inverse problem. Tichonov regularization, although commonly used, cannot handle large scale applications within a reasonable amount of time and memory. Chapters 4 and 5 presented methods to obtain solutions, similar to the Tichonov solution, using subspace and hybrid techniques. These techniques require substantially less computing time and memory than Tichonov regularization. Chapter 4 reviewed Krylov space methods, Lanczos bidiagonalization, least-squares QR and conjugate gradient least-squares, and studied the behaviour of the Krylov filter. Other subspaces which were analysed in this chapter are a new multilevel formulation and a subspace formed from gradients. Finally, the chapter reviewed noise estimation criteria and extended the generalized cross validation criterion so it can be used effectively in subspace regularization. Chapter 5 introduced hybrid regularization methods. It further developed existing hybrid Krylov methods, reviewed gradient hybrid methods and developed a new iterated Krylov space method. Chapter 6 summarized Chapters 2-5 by using and testing the

techniques which were developed in these chapters on examples from gravity and tomography. This chapter showed that all variants of GCV (full-space, subspace and hybrid) are very robust and stable as noise estimation techniques. It is demonstrated that the GCV methods tend to detect uncorrelated noise and to ignore correlated noise. Chapter 6 also compared the different methods for solving linear inverse problems. Krylov space methods were the most efficient methods (for most cases), while multilevel methods were the slowest. The experiment with the nonlinear gravity problem demonstrated the ability to carry out one iteration in a nonlinear process. This experiment also showed that the different variants of the GCV tend to ignore the nonlinear terms in the right hand side, and produce a model which fits the linear iteration to within the predicted noise level.

Chapters 7-10 comprise the second part of this thesis, and made extensive use of the linear algebra and the noise estimation methods which were developed in the first part of the thesis. Chapter 7 reviewed the major aspects of the formulation of nonlinear inverse problems, nonlinear minimization techniques and other commonly used algorithms for the solution of nonlinear inverse problems. This chapter also reviewed the commonly used two stage method and demonstrated that this method might fail. A new explanation is suggested for this failure. Chapter 8 discusses new techniques for solving nonlinear inverse problems. First the cooling method was developed. This method is an improvement of a method presented in Chapter 7. Then a new method for solving nonlinear inverse problems was developed. This method is based on separation of the regularization into local regularization (to overcome the nonlinearity of the problem) and global regularization (to overcome the non-uniqueness of the problem). The method uses the GCV for the global regularization and damped Gauss-Newton for the local regularization. In this chapter it was also shown how to calculate an approximation to the solution for large scale problems using Krylov spaces and hybrid methods. Chapter 9 discussed approximate sensitivities. The chapter presented a new formulation of approximate sensitivities

and reviewed a few of the common methods to obtain them. The new formulation allows one to compute a bound on the distance between the approximate sensitivities solutions and the Tichonov solution. Chapter 10 summarized the ideas and concepts of Chapters 7-9 by applying them to two generic examples: the gravity interface problem and the magnetotelluric problem. Experiments with both examples showed that the techniques can estimate the noise accurately and obtain satisfactory solutions. In the comparison of the methods, hybrid solutions tend to be the most stable.

Future work to be done is to tackle other computational challenges which evolve from inverse problems. These includes: combining the methods presented here with other constraints such as bounds on the unknowns, using differential equations in order to solve ill-posed problems, estimating resolution and inference of a nonlinear problem and general improvement of the algorithms by parallelizing them.

References

[1] Altman, J. 1987. Smoothing Data with Correlated Errors. *Tech. Report 280. Dept. of Stat., Stanford.*

[2] Armijo, L., 1966. Minimization of a Function Having Lipschitz-Continuous First Partial Derivatives. *Pacific J. Math.*, 16, 1-3.

[3] Backus, G.E. and Gilbert, J.F., 1968. The Resolving Power of Gross Earth Data. *Geophys. J. Roy. Astr. Soc.*, 16, 169-205.

[4] Backus, G.E. and Gilbert, J.F., 1970. Uniqueness in the Inversion of Inaccurate Gross Earth Data. *Phil. Trans. Roy. Soc.*, 266A, 123-150.

[5] Bakushinsky, A., Goncharsky, A., 1994. Ill-Posed Problems: Theory and Applications. *Kluwer Academic Publishing.*

[6] Biggs, M. C., Hernandez, M. de F. G., 1995. Using the KKT Matrix in an Augmented Lagrangian SQP Method for Sparse Constrained Optimization. *Journal of Optimization Theory and Applications*, 85, 1, 201-209.

[7] Björk, A., 1990. Error Analysis of Modified Gram Schmidt Method for Solving Underdetermined Linear Systems. *Linkoping University Report*, LiTH-MAT-R-1990-06.

[8] Björk, A., Strakos Z., 1995. Stability of Conjugate Gradient Type Methods for Linear Least Squares Problems. *Linkoping University Report*, LiTH-MAT-R-1995-26.

[9] Björk, A., 1996. Numerical Methods for Least Squares Problems, *SIAM, Philadelphia*

[10] Blankely, R.J., 1995. Potential Theory in Gravity and Magnetic Applications. *Cambridge University Press.*

[11] Born, M., 1975. Principles of Optics. *Pergamon Press*, 5th ed.

[12] Briggs, W., 1987. A Multigrid Tutorial. *SIAM, Philadelphia.*

[13] Broyden, C.G., 1965. A Class of Methods for Solving Nonlinear Simultaneous Equations. *Math. Comput.*, 19, 577-593.

[14] Brown, P., Saad, Y., 1990. Hybrid Krylov Methods for Nonlinear Systems of Equations *SIAM J Sci Stat. Comp.* , 11, 3 450-481

[15] Brown, P., Saad, Y., 1994. Convergence Theory of Nonlinear Newton-Krylov Algorithms. *SIAM J Opt.* , 4, 2 297-330

[16] Chew, W.C., 1990. Waves and fields in inhomogeneous media *New York : Van Nostrand Reinhold.*

[17] Claerbout, J. 1985 Imaging the Earth's Interior. *Blackwell Scientific Publications.*

[18] Constable, S.C., Parker, R.L. and Constable, C.G., 1987. Occam's Inversion: A Practical Algorithm for Generating Smooth Models from Electromagnetic Sounding Data. *Geophysics*, 52, 289-300.

[19] Cost, J.R., 1983. Nonlinear Regression Least-Squares Method for Determining Relaxation Time Spectra for Processes with First-Order Kinetics. *J. Appl. Phys.*, 54, 2137-2146.

[20] Dembo, R., Steihaug, T., 1983. Inexact Newton Algorithms for Large Scale Optimization. *Math. Programming*, 26, 190-212.

[21] Dendy. J.E., 1982 Black Box Multigrid. *J. Computational Physics*, 48, 366-386.

[22] Dennis, J.E., Martinez, H.J., Tapia, R.A., 1989. Convergence Theory for the Structured BFGS Scant Method with an Application to Nonlinear Least Squares. *J. Optim. Applic.*, 61, 2, 161-178.

[23] Dennis, J.E., Song-bai, S., Phuong, A.V., 1988, A Memoryless Augmented Gauss-Newton Method for Nonlinear Least-Squares Problems. *J. Comp. Math.*, 6, 4, 355-363.

[24] Dennis, J.E., Schnabel, R.B., 1996. Numerical Methods for Unconstrained Optimization and Nonlinear Equations. *SIAM Philadelphia.*

[25] Devaney, A. J. 1989. The Limited-View Problem in Diffraction Tomography. *Inverse problems.* 5, 4, 510-510.

[26] Dosso, S.E. 1990. Inversion of 1-D Magnetotelluric Data. *Ph.D Thesis, The University of British Columbia.*

[27] Eisenstat, S.C., Walker, H.F., 1994. Choosing the Forcing Terms in Inexact Newton Method. *Report 6/94/75, Mathematics and Statistics Dept. Utah State University.*

[28] Elden, L., 1977. Algorithms for the Regularization of Ill-Conditioned Least Squares Problems. *BIT*, 17, 134-145.

[29] Elden, L., 1982. A Weighted Pseudo-inverse, Generalized Singular Values, and Constrained Least Squares Problems. *BIT*, 22, 487-502.

[30] Elden, L., 1990. Algorithms for the Computation of Functions defined on the Solution of a Discrete ill-posed problem. *BIT*, 30, 466-483.

[31] Ellis, R.G., Farquharson, C.G., Oldenburg, D.W., 1993. Approximate Inverse Mapping Inversion of COPROD2 Data. *J. Geomag. Geoelectr.*, 45, 1001-1012.

[32] Elster, C. Neumaier, A. 1997. A Method of Trust Region Type for Minimizing Noisy functions. *Computing : archiv fur informatik und numerik*, 58, 1, 31-38.

[33] Engl, H.W., Hanke, M., Neubaur, A., 1996. Regularization of Inverse Problems *Kluwer Academic Publishers.*

[34] Farquharson, C.G., Oldenburg, D.W., 1996. Approximate Sensitivities for the Electromagnetic Inverse Problem. *Geophys. J. Int.*, 126, 235-252.

[35] Farquharson, C.G.,1995. Approximate Sensitivities for the Multi-Dimensional Electromagnetic Inverse Problem. *Ph.D Thesis, The University of British Columbia.*

[36] Fraley, C., 1989. Computational Behavior of Gauss-Newton Methods. *SIAM J. Sci. Stat. Comp.*, 10, 3, 515-523.

[37] Gauss, C.F., 1801. (Translation Stewart G.W.), Theory of the Combination of Observations Least Subject to Errors. *SIAM Philadelphia.*

[38] Gill, P.E., Murray, W., Wright, M.H., 1981. Practical Optimization. *New York Academic Press.*

[39] Golub, G.H., Heath, M. and Wahba, G., 1979. Generalized Cross-Validation as a Method for Choosing a Good Ridge Parameter. *Technometrics*, 21, 215-223.

[40] Golub, G.H., Van Loan, C.F., 1996. Matrix Computations *The John Hopkins University Press, Second Edition*

[41] Golub, G.H., Von Matt. U, 1991. Quadratically Constrained Least Squares and Quadratic Problems. *Numer. Math.*, 59, 561-580.

[42] Golub, G.H., Von Matt. U, 1996. Generalized Cross-Validation for Large Scale Problems. *Submitted to SIAM.*

[43] Green, P.J. 1990. Bayesian Reconstruction from Emission Tomography Data Using a Modified EM Algorithm. *IEEE in Medical Imaging*, 9, 84-93.

[44] Gulliksson, M., Soderkvist, I., Wedin, P.A., 1997. Algorithms for Constrained and Weighted Nonlinear Least Squares. *SIAM J. Optim.*, 7, 1, 208-219.

[45] Haber, E. Oldenburg, D. 1996. Joint Inversion: A structural approach *Inverse Theory,* 3, 253-315.

[46] Haber, E. Oldenburg, D. Celler, A. 1996. Direct Estimation of Kinetic Parameters In Dynamic SPECT. *IEEE on Nuclear Science*, 3, 253-315.

[47] Hackbusch, W., 1985. Multigrid Methods and Applications. *Springer Verlag New-York.*

[48] Hadamard, J., 1923. Lectures on Cauchy's Problem in Linear Partial Differential Equations. *Yale University Press, New Haven.*

[49] (a) Hanke, M., 1997. A Regularizing Levenberg-Marquardt Scheme, with Applications to Inverse Groundwater Filtration Problems. *Inverse Problems*, 13, 79-95.

[50] (b) Hanke, M., 1997. Regularizing Properties of A Truncated Newton CG Algorithm for Nonlinear Inverse Problems. *Submitted to: Inverse Problems.*

[51] (c) Hanke, M., 1997. Limitations of the L-Curve Method in Ill-Posed Problems. *Submitted to BIT.*

[52] Hanke, M., Hansen, P.C, 1993. Regularization methods for large scale problems *Survey on Mathematics for Industry*, 3, 253-315.

[53] (a) Hansen, P.C., 1992. Analysis of Discrete Ill-Posed Problems by Means of the L-Curve. *SIAM Review*, 34, 561-580.

[54] (b) Hansen, P.C., 1992. Numerical Tools for Analysis and Solution of Fredholm Integral Equations of the First Kind. *Inverse Problems*, 8, 849-872.

[55] Hansen, P.C., 1995. Rank Deficient and Discrete Ill-Posed Problems. *Ph.D Thesis, Technical University of Denmark.*

[56] Hestenes, M., Stiefel, E., 1952. Methods of Conjugate Gradients for Solving Linear Systems. *Cons. Reas. Nat. Bur. Stand.*, 49, 6, 409-436.

[57] Hestenes, M. 1980. Conjugate Direction Methods in Optimization Springer-Verlag.

[58] Jerri, A.J., 1985. Introduction to Integral Equations with Applications, *Marcel Deker Inc.*

[59] Jingsheng, L., Elsworth, D., 1995. A Modified Gauss-Newton Method for Aquifer Parameter Identification. *Ground Water; Journal of the National Water Well*, 33, 4, 662-671.

[60] Kelley, C.T., 1995. Iterative Methods for Linear and Nonlinear Equations. *SIAM Philadelphia.*

[61] Kennett, B. L. N., Williamson, P.R., 1988. Eds - Vlaar, N.J,. Nolet, G., Wortel, M.J.R., Cloetingh, S.A.P.L., Reidel, D., Dordrecht. Subspace Methods for Large Scale Nonlinear Inversion. *in: Mathematical Geophysics: a Survey of Recent Developments in Seismology and Geodynamics,* 139-154.

[62] Knoth, O., 1996. A Globalization Scheme for the Generalized Gauss-Newton Method. *Num. Math.*, 56, 2, 591-603.

[63] Lanczos, C., 1961. Linear Differential Operators. D. Van Nostrand Co., London.

[64] Landweber, L. 1951. An Iteration Formula for Fredholm Integral Equations of the first kind. *Amer. J. Math.* 73, 615-624.

[65] Lange K. Carson R. 1984. EM reconstruction algorithm for emission and transmission tomography. *J. Comput. Assist. Tomogr.*, 8, 306-316.

[66] Lawson, C.L. and Hanson, R.J., 1974. Solving Least Squares Problems. Prentice–Hall, Englewood Cliffs, New Jersey.

[67] Li, Y. Oldenburg, D.W., 1996. 3-D Inversion of Magnetic Data. *Geophysics* 61, 2, 394-408.

[68] Li, Y. 1992. Inversion of 3-D DC Resistivity Data. *Ph.D Thesis, The University of British Columbia.*

[69] Li, Y. 1996. 3-D Inversion of Gravity Data. *Submitted to Geophysics.*

[70] Luenberger, D.G., 1969. Optimization by Vector Space Methods, *New-York: John Wiley and Sons.*

[71] Marquardt, D.W., 1963. An Algorithm for Least Squares Estimation of Non-linear Parameters. *J. Soc. Ind. Appl. Math.*, 11, 431-441.

[72] McGaughey J., 1994 The Applicability of Radio Wave Tomography to Mapping Out the Louvicourt Copper Deposit. *Minigtek report*, Reference number Y5114/63/94.

[73] McGillivray, P.R., 1992. Forward Modelling and Inversion of DC Resistivity and MMR Data. *Ph.D Thesis, The University of British Columbia.*

[74] Menke, W., 1984. Geophysical Data Analysis: Discrete Inverse Theory. *Academic Press, Inc., Orlando.*

[75] Mirzaei, M., 1996. Inversion of Potential Field Data. *Ph.D Thesis, Utrecht University.*

[76] Molton. D.J., Morel. J.E., Ascher. U.M, 1997. Approximate Schur Complement Preconditioning of Lowest Order Nodal Discretization. *Submitted to SIAM.*

[77] Morse, P.M., Feshbach, H., 1953. Methods of Theoretical Physics. *McGraw-Hill, New-York.*

[78] Nagi. D., 1966. The Gravitational Attraction of a Right Rectangular Prism *Geophysics* 31, 2, 362-371.

[79] Nolet, G., Snieder, R., 1990. Solving Large Inverse Problems by Projection *Geophys. J. Int.*, 103, 565-568.

[80] Nunez, J., Llacer, J. 1990. A Fast Bayesian reconstruction algorithm for Emission Tomography with Entropy Prior Converging to Feasible Images. *IEEE on Medical Imaging.* 9, 159-171.

[81] Nychka, D., Wahba, G.,Goldfarb, S., Pugh, T. 1984. Cross-Validation Spline Methods for the Estimation of Three-Dimensional Tumor Size Distribution from Observations on Two Dimensional Cross Section. *J. Amer. Stat. Assoc.*, 79, 832-846.

[82] Oldenburg, D.W., 1979, One-Dimensional Inversion of Natural Source Magnetotelluric Measurements. *Geophysics*, 44, 1218-1244.

[83] Oldenburg, D.W., 1983. Funnel Functions in Linear and Nonlinear Appraisal. *J. Geophys. Res.*, 88, 7387-7398.

[84] Oldenburg, D.W., 1984. An Introduction to Linear Inverse Theory. *IEEE Trans. Geosci. Remote Sensing*, GE-22, 665-674.

[85] Oldenburg, D.W., Ellis, R.G., 1991. Inversion of Geophysical Data Using an Approximate Inverse Mapping. *Geophys. J. Int.*, 105, 325-353.

[86] Oldenburg, D.W., McGillvary, P.R., Ellis, R.G., 1993 Generalized Subspace Methods for Large Scale Inverse Problems. *Geophys. J. Int.*, 114, 12-20.

[87] Oldenburg, D.W., Li, Y. 1994. Subspace Linear Inverse Method. *Inverse Problems.* 10, 915-935.

[88] O'Leary, D.P., Simmons, J.A., 1981. A Bidiagonalization-Regularization Procedure for Large Scale Regularization of Ill-Posed Problems. *SIAM J. Sci. Stat. Comut.*, 2, 474-489.

[89] Paige, C.C., Saunders, M.A., 1982. LSQR: an Algorithm for Sparse Linear Equations and Sparse Least Squares. *ACM Trans. Math. Software,* 8, 195-209.

[90] (a) Parker, R.L., 1977. Understanding Inverse Theory. *Ann. Rev. Earth. Plan. Sci.*, 5, 35-64.

[91] (b) Parker, R.L., 1977. The Frechet Derivative for the Electromagnetic Induction Problem. *Geophys. J. Roy. Astr. Soc.*, 68, 165-170.

[92] Parker, R.L., Whaler, K.A, 1981. Numerical Methods for Establishing Solutions to Inverse Problem of Electromagnetic Induction. *J. Geophys. Res.*, 86, 9574-9584.

[93] Parker, R.L., Shure, L., Hildebrand, J.A., 1987. The Application of Inverse Theory to Seamount Data. *Rev Geophys.,* 25, 17-40.

[94] Parker, R.L., 1994. Geophysical Inverse Theory. *Princeton University Press, Princeton, New Jersey.*

[95] Parker, R.L. and McNutt, M.K., 1980. Statistics for the One-Norm Misfit Measure. *J. Geophys. Res.*, 85, 4429-4430.

[96] Parlett, B.N, 1980. The Symmetric Eigenvalue Problem. *Prentice-Hall.*

[97] Ros, D., Falcon, C., Pavia, J., 1996. The influence of a relaxation parameter on SPECT iterative reconstruction algorithms. *Physics in Medicine and Biology,* 41, 5, 925.

[98] Routh, P., Oldenburg, D.W., 1996. AIM for Borehole Tomography. *JACI Report 1996.*

[99] Saad, Y., 1987. Least Squares Polynomials in the Complex Plane and Their Use for Solving Non-symmetric Linear Systems *SIAM J. Numer. Anal.*, 24, 1, 155-169.

[100] Saad, Y., 1996. Iterative Methods for Sparse Linear Systems. *PWS Publishing Company.*

[101] Scales, J.A., 1987. Tomographic Inversion via the Conjugate Gradient Method. *Geophysics*, 52, 179-185.

[102] Scales, J.A., Smith, M.L., 1994. Introductory Geophysical Inverse Theory. *Samizdat Press.*

[103] Scales, J.A., Docherty, P., Gersztenkorn, A., 1990. Regularization of Nonlinear Inverse Problems: Imaging the Near Surface Weathering Layer. *Inverse Problems*, 6, 115-131.

[104] Shamanskii, V.E., 1967. A Modified Newton Method. *Ukran. Mat. Zh.*, 19, 133-138.

[105] Shepp, L.A., Vardi, Y., 1982. Maximum Likelihood Reconstruction for Emission Tomography. *IEEE on Medical Imaging*, 1, 113-122.

[106] Smith, J.T., Booker, J.R., 1991 Rapid Inversion of Two and Three Dimensional Magnetotelluric Data. *J. Geophys. Res.*, 96, 3905-3922.

[107] Smith,, M.F., Floyd, C.E., Jaszczak, R.J., Coleman, R.E. 1992. Reconstruction of SPECT images Using Generalized Matrix Inverses. *IEEE on Medical Imaging*, 12, 165-175.

[108] Stirling, D.S., Porter, D., 1990. Integral Equations. *Cambridge Texts in Applied Mathematics.*

[109] Tarantola, A., Valette, B., 1982. Generalized Nonlinear Inverse Problems Solved Using the Least Squares Criterion. *Rev. Geophys. Space Phys.*, 20, 2, 219-232.

[110] Tarantola, A. Inverse Problem Theory. 1987 *Elsevier Scientific Publishing Company, Amsterdam.*

[111] Thompson, D.R., Rodi, W., Toksoz, M.N., 1994. Nonlinear Seismic Diffraction Tomography Using Minimum Structure Constraints. *J. Acoust. Soc. Am.*, 95, 1, 324-330.

[112] Tichonov, A.N. 1963. Solutions of incorrectly formulated problems and the regularization method, *J. Soviet Math. Dokl.*, 4, 1035-1038.

[113] Tichonov, A.N., Arsenin, V.Y., 1977. Solutions of Ill-Posed Problems *John Wiley and Sons, Inc.*

[114] Twomey, S., 1977. Introduction to the Mathematics of Inversion in Remote Sensing and Indirect Measurements. Elsevier Scientific Publishing Company, Amsterdam.

[115] Van Huffel, S., Vandewalle, J. 1991. The Total Least Square Problem. *SIAM* Philadelphia.

[116] Varah, J.M., 1983, Pitfalls in Numerical Solutions of Linear Ill-Posed Problems. *Siam J. Sci. Comp.*, 4 164-176.

[117] Wahba, G. 1977. Practical Approximate Solutions to Linear Operator Equations When The Data are Noisy. *SIAM J. Numer Anal.*, 14, 651-667.

[118] Wahba, G. 1990. Spline Models for Observational Data. *SIAM Philadelphia.*

[119] Ward, S.H., Hohmann, G.W., 1988. Electromagnetic theory for geophysical Applications, in *Electromagnetic Methods in Applied Geophysics.*, 1 131-311. Soc. Expl. Geophys.

[120] Whittall, K.P., Oldenburg, D.W., 1992. Inversion of Magnetotelluric Data for a One Dimensional Conductivity. *SEG. Monograph*, 5.

[121] Young, C.T., Booker, J.R., Stodt, J., Waff, H.S., Wannamaker, P.E., 1988. Verification of Five Magnetotelluric Systems in the mini-EMSLAB experiment. *Geophysics*, 53, 553-557.

[122] Zhang, J. Mackie, R.L., Madden, T.R., 1995. 3-D Resistivity Forward Modeling and Inversion Using Conjugate Gradients. *Geophysics*, 60, 5, 1313-1325.