Inversion of 2D Magnetotelluric and Radiomagnetotelluric Data with Non-Linear Conjugate Gradient Techniques

Inversion av magnetotelluriska och radiomagnetotelluriska data med den icke-linjärt konjugerade gradientmetoden i två dimensioner

Dominik Zbinden
Inversion of 2D Magnetotelluric and Radiomagnetotelluric Data with Non-Linear Conjugate Gradient Techniques

Inversion av magnetotelluriska och radiomagnetotelluriska data med den icke-linjärt konjugerade gradientmetoden i två dimensioner

Dominik Zbinden
Abstract

Inversion of 2D Magnetotelluric and Radiomagnetotelluric Data with Non-Linear Conjugate Gradient Techniques
Dominik Zbinden

I implemented and tested the method of Non-Linear Conjugate Gradients (NLCG) to invert magnetotelluric (MT) and radiomagnetotelluric (RMT) data in two dimensions. The forward problem and the objective function gradients were computed using finite-difference methods. The NLCG algorithm was applied to three field data sets to test the performance of the code. It was then compared to the inversion techniques of Occam and damped Occam considering the quality of the output resistivity models and the computation times. The implemented code was further investigated by testing two line search techniques to reduce the objective function along a given search direction. The first line search procedure was constrained to the first Wolfe condition, leading to a rather inexact line search. The second, more thorough line search, was additionally constrained to the second Wolfe condition. Three preconditioners were applied to the NLCG algorithm and their performance was analysed. The first preconditioner was set to the diagonal of the approximate Hessian matrix and updated every 20-th iteration. Preconditioners two and three were updated with the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm using the identity matrix and the diagonal of the approximate Hessian matrix as start preconditioners, respectively.

The tests showed that the method of NLCG is more efficient pertaining to computation times compared to the Gauss-Newton (GN) based techniques (Occam and damped Occam). For the two smaller data sets that were inverted, the NLCG inversion was two to four times faster than Occam and damped Occam. For the larger data set, the NLCG inversion converged more than one order of magnitude faster than the GN based inversion techniques. This is because GN methods require to evaluate the entire sensitivity matrix to update the model, whereas NLCG only needs to compute a matrix-vector product of the Jacobian. Moreover, expensive operations such as matrix products and direct inversions of linearised systems are avoided by NLCG. A limitation of the NLCG algorithm is that it is prone to converge to local minima due to the fixed Lagrange multiplier that is used in the penalty function. Occam inversion, which determines the optimal Lagrange multiplier as part of the inversion, did not show such problems. The line search tests of the NLCG algorithm showed that an inexact line search yields higher convergence per CPU time than a more exact line search. In accordance to previous studies, preconditioning accelerated the convergence of the NLCG algorithm considerably. The preconditioners updated with the BFGS algorithm achieved highest convergence. Choosing the identity matrix as a start preconditioner led to fast but unstable convergence. The reasons for that could not be determined completely. Taking the diagonal of the approximate Hessian as a start preconditioner instead of the identity matrix led to slower convergence for most of the inversion tests, but convergence could be stabilised.

All the tests performed within this project led to a robust implementation of the NLCG algorithm. A default set-up pertaining to line search and preconditioning could be established. However, the NLCG set-up can be adjusted by the user to improve convergence for a specific data set. This makes the algorithm implemented in this thesis more flexible than previously introduced NLCG codes. Preconditioning can certainly still be improved with further tests. Moreover, a future project will be to extend the 2D code to 3D, where NLCG should perform especially well, because the number of model parameters is usually higher in 3D.

Keywords: Numerical solutions, inverse theory, non-linear conjugate gradients, preconditioning, electromagnetic theory, magnetotellurics

Degree Project E in Geophysics, 1GE029, 30 credits
Supervisors: Thomas Kalscheuer and Hansruedi Maurer
Department of Earth Sciences, Uppsala University, Villavägen 16, SE-752 36 Uppsala (www.geo.uu.se)

ISSN 1650-6553, Examensarbete vid Institutionen för geovetenskaper, No. 329, 2015

The whole document is available at www.diva-portal.org


Nyckelord: Numeriska lösningar, inversionsteori, icke-linjärt konjugerade gradienter, förkonditioner, elektromagnetisk teori, magnetotelluriska mätningar

Examensarbete E i geofysik, 1GE029, 30 hp
Handledare: Thomas Kalscheuer och Hansruedi Maurer
Institutionen för geovetenskaper, Uppsala universitet, Villavägen 16, 752 36 Uppsala (www.geo.uu.se)

ISSN 1650-6553, Examensarbete vid Institutionen för geovetenskaper, Nr 329, 2015

Hela publikationen finns tillgänglig på www.diva-portal.org
List of Figures

2.1 The geometry of Marquardt-Levenberg, Gauss-Newton and Steepest Descent search directions. ................................. 6
3.1 The first and second Wolfe condition. ......................................................... 20
3.2 The line search procedure when only the first Wolfe condition is used. .................... 21
4.1 Solution convergence for the TE data from Smørgrav for three different Lagrange multipliers. .................. 28
4.2 Resistivity model obtained by NLCG and Occam inversion for the TE data from Smørgrav. .................. 29
4.3 Solution convergence for the TM data from Smørgrav for four different Lagrange multipliers. .................. 30
4.4 Resistivity model obtained by NLCG and Occam inversion for the TM data from Smørgrav. .................. 31
4.5 Resistivity model obtained by NLCG, Occam and damped Occam inversion for the joint inversion of TE and TM data from Smørgrav. .................................................. 32
4.6 Comparison of convergence between Occam, damped Occam and NLCG inversion for the joint inversion of TE and TM data from Smørgrav. .................................................. 33
4.7 Resistivity model obtained by Occam, damped Occam and NLCG inversion for the DET data from Skediga. .................................................. 36
4.8 Comparison of convergence between Occam, damped Occam and NLCG inversion for the DET data from Skediga. .................................................. 37
4.9 Comparison of convergence between different preconditioners for the DET data from Skediga. .................................................. 38
4.10 Resistivity model obtained by Occam and NLCG inversion for the DET data from Åvrö. .................................................. 40
4.11 Comparison of convergence between Occam and NLCG inversion for the DET data from Åvrö. .................................................. 41
4.12 Comparison of convergence between different preconditioners for the DET data from Åvrö. .................................................. 42
4.13 Solution convergence for the TE data from Smørgrav for five different search direction update techniques. .................................................. 43
4.14 Solution convergence for the TE data from Smørgrav for five different trial step techniques. .................................................. 43
4.15 Solution convergence for the DET data from Skediga for different line search techniques. .................................................. 44
A.1 Discretisation of the finite-difference mesh around node (i, j). .................................................. 56
D.1 Comparison of convergence between Occam and NLCG inversion for the TE data from Smørgrav. .................................................. 67
D.2 Solution convergence for different preconditioners for the TE data from Smørgrav. .................................................. 68
D.3 Resistivity models obtained for different values of $\lambda$ for the TE data from Smørgrav. .................................................. 68
D.4 Resistivity models obtained in different stages of the NLCG algorithm for the TE data from Smørgrav. .................................................. 69
D.5 Comparison of convergence between Occam and NLCG inversion for the TM data from Smørgrav. .................................................. 70
D.6 Solution convergence for different preconditioners for the TM data from Smørgrav. .................................................. 70
D.7 Resistivity models obtained for different values of $\lambda$ for the TM data from Smørgrav. .................................................. 71
D.8 Resistivity models obtained in different stages of the NLCG algorithm for the TM data from Smørgrav. .................................................. 72
D.9 Solution convergence for different Lagrange multipliers for the joint inversion of TE and TM data from Smørgrav. .................................................. 73
D.10 Solution convergence for different preconditioners for the joint inversion of TE and TM data from Smørgrav. .................................................. 73
D.11 Resistivity models obtained for different values of $\lambda$ for the joint inversion of TE and TM data from Smørgrav. .................................................. 74
D.12 Resistivity models obtained in different stages of the NLCG algorithm for the joint inversion of TE and TM data from Smørgrav. .................................................. 75
D.13 Solution convergence for different Lagrange multipliers for the DET data from Skediga. .................................................. 76
D.14 Solution convergence for different preconditioners for the DET data from Skediga. .................................................. 76
D.15 Resistivity models obtained by NLCG inversion for different values of $\lambda$ for the DET data from Skediga. .................................................. 77

D.16 Resistivity model obtained by Occam, damped Occam and NLCG inversion for the joint inversion of TE and TM data from Smørgrav. .................................................. 32
D.17 Solution convergence for the TE data from Smørgrav for three different Lagrange multipliers. .................................................. 28
D.18 Resistivity model obtained by NLCG and Occam inversion for the TE data from Smørgrav. .................................................. 29
D.19 Solution convergence for the TM data from Smørgrav for four different Lagrange multipliers. .................................................. 30
D.20 Resistivity model obtained by NLCG and Occam inversion for the TM data from Smørgrav. .................................................. 31
D.21 Resistivity model obtained by NLCG, Occam and damped Occam inversion for the joint inversion of TE and TM data from Smørgrav. .................................................. 32
D.22 Comparison of convergence between Occam, damped Occam and NLCG inversion for the joint inversion of TE and TM data from Smørgrav. .................................................. 33
D.23 Resistivity model obtained by Occam, damped Occam and NLCG inversion for the DET data from Skediga. .................................................. 36
D.24 Comparison of convergence between Occam, damped Occam and NLCG inversion for the DET data from Skediga. .................................................. 37
D.25 Comparison of convergence between different preconditioners for the DET data from Skediga. .................................................. 38
D.26 Resistivity model obtained by Occam and NLCG inversion for the DET data from Åvrö. .................................................. 40
D.27 Comparison of convergence between Occam and NLCG inversion for the DET data from Åvrö. .................................................. 41
D.28 Comparison of convergence between different preconditioners for the DET data from Åvrö. .................................................. 42
D.29 Solution convergence for the TE data from Smørgrav for five different search direction update techniques. .................................................. 43
D.30 Solution convergence for the TE data from Smørgrav for five different trial step techniques. .................................................. 43
D.31 Solution convergence for the DET data from Skediga for different line search techniques. .................................................. 44
D.32 Discretisation of the finite-difference mesh around node (i, j). .................................................. 56
D.33 Comparison of convergence between Occam and NLCG inversion for the TE data from Smørgrav. .................................................. 67
D.34 Solution convergence for different preconditioners for the TE data from Smørgrav. .................................................. 68
D.35 Resistivity models obtained for different values of $\lambda$ for the TE data from Smørgrav. .................................................. 68
D.36 Resistivity models obtained in different stages of the NLCG algorithm for the TE data from Smørgrav. .................................................. 69
D.37 Comparison of convergence between Occam and NLCG inversion for the TM data from Smørgrav. .................................................. 70
D.38 Solution convergence for different preconditioners for the TM data from Smørgrav. .................................................. 70
D.39 Resistivity models obtained for different values of $\lambda$ for the TM data from Smørgrav. .................................................. 71
D.40 Resistivity models obtained in different stages of the NLCG algorithm for the TM data from Smørgrav. .................................................. 72
D.41 Solution convergence for different Lagrange multipliers for the joint inversion of TE and TM data from Smørgrav. .................................................. 73
D.42 Solution convergence for different preconditioners for the joint inversion of TE and TM data from Smørgrav. .................................................. 73
D.43 Resistivity models obtained for different values of $\lambda$ for the joint inversion of TE and TM data from Smørgrav. .................................................. 74
D.44 Resistivity models obtained in different stages of the NLCG algorithm for the joint inversion of TE and TM data from Smørgrav. .................................................. 75
D.45 Solution convergence for different Lagrange multipliers for the DET data from Skediga. .................................................. 76
D.46 Solution convergence for different preconditioners for the DET data from Skediga. .................................................. 76
D.47 Resistivity models obtained by NLCG inversion for different values of $\lambda$ for the DET data from Skediga. .................................................. 77
List of Tables

3.1 Important discrepancies between MT and RMT forward modelling. .......................... 14
4.1 Input parameters for NLCG inversion. ........................................................................ 27
4.2 Input parameters for Occam and damped Occam inversion. ...................................... 27
4.3 Specifications for data and model from Smørgrav. ....................................................... 28
4.4 Comparison of objective function and RMS obtained after 100 iterations for different preconditioners for the data from Smørgrav. .......................................................... 33
4.5 Specifications for data and model from Skediga. ......................................................... 35
4.6 Specifications for data and model from Ávró. ............................................................. 39
C.1 NLCG input parameters in EMILIA. ......................................................................... 64
# Table of Contents

1 **Introduction** .......................................................... 1
   1.1 Electromagnetic modelling and inversion .......................... 1
   1.2 Aim of my thesis and outline ...................................... 2

2 **Background** ............................................................ 3
   2.1 Forward modelling of electromagnetic data ......................... 3
   2.2 Inversion of electromagnetic data ................................ 4
      2.2.1 Non-linear inversion techniques ............................. 4
      2.2.2 Inversion with Conjugate Gradient techniques .............. 7

3 **Method** ............................................................... 10
   3.1 Electromagnetic theory ............................................. 10
      3.1.1 Electromagnetic field propagation ............................ 10
      3.1.2 Forward problem .............................................. 13
      3.1.3 Sensitivity matrix ........................................... 15
   3.2 Inversion with Non-Linear Conjugate Gradients .................. 15
      3.2.1 Linear Conjugate Gradients ................................. 15
      3.2.2 Non-Linear Conjugate Gradients ........................... 17
      3.2.3 Line search ................................................. 19
      3.2.4 Preconditioning ............................................. 23

4 **Results** ............................................................... 26
   4.1 TE and TM data from Smørgrav, Norway .......................... 27
   4.2 Determinant data from Skediga, Sweden ......................... 34
   4.3 Determinant data from Ávrö, Sweden ............................ 38
   4.4 Convergence analysis of NLCG input parameters ................ 41

5 **Discussion** ........................................................... 45

6 **Conclusions** ........................................................ 49

7 **Acknowledgements** ................................................ 50

8 **Bibliography** ....................................................... 51

**Appendices** ................................................................ 55

A **Finite-difference approach** ........................................ 56

B **Sensitivity matrix-vector products** ............................... 60
   B.1 Sensitivity matrix computation .................................... 60
   B.2 Sensitivity matrix-vector product computation ................ 62

C **Input parameters for NLCG algorithm** ............................ 64

D **Additional results** .................................................. 67
   D.1 TE data from Smørgrav, Norway .................................. 67
   D.2 TM data from Smørgrav, Norway .................................. 70
   D.3 Joint inversion of TE and TM data from Smørgrav, Norway  .... 73
   D.4 DET data from Skediga, Sweden .................................. 76
   D.5 DET data from Ávrö, Sweden ...................................... 79
   D.6 Convergence analysis of NLCG input parameters ............... 82
1 Introduction

1.1 Electromagnetic modelling and inversion

Electromagnetic (EM) sounding is a non-destructive technique that is applied to retrieve electrical properties of the subsurface. It has wide applications in mineral and ground water exploration as well as environmental tasks such as landfill surveys, geothermal investigations and permafrost mapping (Reynolds, 2011). The main aim is to model subsurface resistivity structures. This can be used to characterise rocks and fluids that are contained in the subsurface. EM methods are often applied together with borehole information and other data like seismic reflection profiles to obtain detailed and realistic models of the ground. Different methods exist that can be applied according to the objective and scale of the problem. They can be divided into methods using external sources and controlled-source EM methods (CSEM). For CSEM, electric bipoles, loops or coils are used as transmitters. External sources can be natural phenomena such as interactions between solar winds and the magnetosphere (magnetotelluric (MT)) or electric fields generated by lightning discharges (audiomagnetotelluric (AMT)). Non-natural external sources are for instance signals from remote radio transmitters (radiomagnetotelluric (RMT)).

The response of the subsurface to the generated EM fields is then measured at a receiver, e.g. with an electric bipole, coil or a magnetometer. The difference of MT, AMT and RMT is mainly the frequency range of the signals. Frequencies vary between 0.001 and 10 Hz for MT, 10 Hz and 10 kHz for AMT (Reynolds, 2011) and between 10 kHz and 300 kHz for RMT (Kalscheuer et al., 2008). MT and RMT methods will be explained in detail in Chapter 3.1.

My thesis is focused on modelling and inversion of MT and RMT data. The task here is to find a subsurface resistivity model that best reproduces the data that are measured at the site. Data are electrical impedances or apparent resistivities and phases. It is possible to calculate data from a given model through forward calculation. The forward problem relates resistivities to electrical impedances and is based on Maxwell’s equations. These laws describe the interaction between electric and magnetic fields and how they are affected by electrical properties of the ground (cf. Chapter 3.1). In applied geophysics, the challenge is to compute a model out of field data, which is known as inversion. This technique has many advantages compared to trial-and-error methods, where data are calculated repeatedly for different trial models until the measured data are reproduced sufficiently. Inversion is a process, where the best model is computed automatically. Important information about the quality of the model like resolution and model covariance can be obtained. Therefore, geophysicists have focused their research on developing and improving inversion techniques during the last thirty years. Thanks to numerous field studies, more data has become available and the demand for robust and efficient forward modelling and inversion codes has increased. To account for complex shaped geometries and bodies, it is necessary to model in two or even three dimensions. Whereas most EM forward problems in 1D can be solved semi-analytically, 2D and 3D problems have to be solved numerically. In multiple dimensions, the number of data and model parameters can get very high and hence, the computation times can get immense. The inversion process implies multiple forward calculations and is even more time-consuming than a single forward problem. For this reason, research has been focused on finding fast and low-cost algorithms to invert EM data. This is either done by approximating forward solutions or by reducing the number of forward problems for the inversion process. An overview of different inversion techniques is given in
the background in Chapter 2. Nowadays, numerous EM inversion programs exist, such as the famous REDuced Basis OCCam’s inversion (REBOCC) code for MT data in two dimensions by Siripunvaraporn and Egbert (2000). REBOCC has served as a base for more advanced inversion programs, such as ElectroMagnetic Inversion with Least Intricate Algorithms (EMILIA) by Kalscheuer (2014). This program can be used to invert MT and RMT data (among other methods) in one or two dimensions. Different inversion schemes are implemented in the program and can be applied to the data. Within the scope of this thesis, I will extend EMILIA by implementing the method of Non-Linear Conjugate Gradients (NLCG), an especially fast and time-efficient inversion algorithm.

NLCG has been used to invert EM data in both two and three dimensions during the last fifteen years. Rodi and Mackie (2001) and Newman and Alumbaugh (2000) introduced two versions of NLCG algorithms and tested them with both synthetic data and field data. Subsequently, these NLCG algorithms were applied to a number of field studies (e.g. Mackie et al., 2007, Newman et al., 2010, Ghaedrahmati et al., 2014, Kamm and Pedersen, 2014). However, almost no additional tests of the performance of the proposed NLCG algorithms were done. For further use of the method of NLCG for EM problems, it is important to know the advantages and limitations of the method. Furthermore, the original NLCG algorithms can potentially be improved by performing additional tests. Hence, besides the implementation of the method of NLCG, this project will also be used to enhance the performance of the NLCG algorithm.

1.2 Aim of my thesis and outline

The inversion program EMILIA contains both Occam and damped Occam inversion (cf. Chapter 2.2). These techniques require high computation times to solve large inverse problems. The objective of my thesis is therefore to implement the method of NLCG, which should be significantly faster than the already implemented techniques. This will be done for MT and RMT data in two dimensions. Convergence of NLCG compared to Occam and damped Occam inversion will be analysed. Furthermore, the resistivity models obtained by NLCG will be compared to those computed by Occam and damped Occam inversion. These comparisons will depict the advantages and limitations of the method of NLCG. The discussion of Rodi and Mackie (2001) and Newman and Alumbaugh (2000) pertaining to the key points of the NLCG algorithm, namely the line search procedure (cf. Chapter 3.2.3) and preconditioning (cf. Chapter 3.2.4), have not been conclusive. Therefore, the convergence of the NLCG code will be further investigated by comparing different line search approaches and preconditioners. In the end, the user of EMILIA should have the possibility to use a fast and robust NLCG algorithm that produces high quality results. Since there is no perfect code that works for every data set, the user should be able to easily adjust parameters of the algorithm to improve the results for individual data sets.

In Chapter 2, a review of both forward modelling and inversion techniques is provided. This is followed by a theoretical part (Chapter 3), where basic EM theory, finite-difference forward modelling for MT and RMT in 2D and the NLCG inversion method are explained. Subsequently, the results are presented (Chapter 4) and discussed (Chapter 5). In Chapter 6, overall conclusions are drawn and an outlook for improvements and future developments is presented.
2 Background

This chapter starts with a short review of EM forward modelling techniques (Chapter 2.1). Since this thesis is focused on inversion of EM data, a detailed overview of inversion methods is given in Chapter 2.2.

2.1 Forward modelling of electromagnetic data

There are three numerical forward modelling techniques that are usually applied by the EM community. These are finite-difference (FD), finite-element (FE) and integral equation (IE) methods. They are summarised in the review of Pankratov and Kuvshinov (2015). The most straight-forward strategy is to implement finite-differences. This has been done in many studies both in 2D (Brewitt-Taylor and Weaver, 1976, Siripunvaraporn and Egbert, 2000, Rodi and Mackie, 2001, Kalscheuer et al., 2008, among others) and 3D (Mackie and Madden, 1993, Newman and Alumbaugh, 2000, among others). Using an FD approach, the model is split into different cells. This discretisation of the model allows for numerically solving the forward problem by approximating the derivatives with first and second order differences. This leads to a linear system of equations of the form $A_{FD}x = b$. The sparse and complex symmetric system matrix $A_{FD}$ can be solved by LU solvers\textsuperscript{1} or iterative methods such as BiCG\textsuperscript{2}. A more detailed explanation of the FD method in two dimensions with illustrations and derivations is given in Appendix A.

Another approach is to discretise the region of interest into sub domains, so-called finite-elements (FE), in which the EM field is approximated by mathematical functions. The linear system of equations is sparse like in the FD approach, but the implementation of FE in an algorithm is rather demanding. However, recent published papers have introduced robust FE solvers with flexible meshes (Farquharson and Miensopust, 2011, Ren et al., 2013, Grayver and Bürg, 2014, among others). The advantage of FE compared to FD is the greater flexibility of the mesh to account for complex geometries in the subsurface. Attempts have been made to allow for topography even for FD (e.g. Aprea et al., 1997). Nevertheless, the greater flexibility of FE remains, because refined meshes can be used in regions with high resistivity contrasts, which improves the accuracy of the final result.

A third common approach to solve EM forward problems is the integral equation method (IE) (e.g. Kamm and Pedersen, 2014, Kelbert et al., 2014). In this approach, the EM field is divided into a normal (for a layered background medium) and an anomalous field. The advantage hereby is that the discretisation only has to be made in the region of the anomalies, which is much smaller than the entire model area. This leads to a small-sized system of linear equations. However, the system matrix for the IE approach is dense and hence, the solution of the forward problem is more difficult to achieve. Furthermore, the IE approach needs the computation of Green’s tensors that are quite difficult to implement.

REBOCC and EMILIA compute their forward solutions with the FD approach. As described above, the simplicity of FD and the stability of the results are the reasons why FD is preferred compared to FE and IE. This is especially important for programs such as EMILIA, where forward modelling algorithms are implemented for different MT transfer functions and in different frequency ranges (cf. Chapter 3.1).

\textsuperscript{1}The LU decomposition divides the system matrix (complex symmetric) in a lower and upper triangular matrix. The inversion can then be solved by forward and backward substitution.

\textsuperscript{2}The Biconjugate gradient (BiCG) is used instead of CG (cf. Chapter 2.2.2) when the system matrix is not self-adjoint, meaning that $A$ is not equal to its conjugate transpose $A^\dagger$.  

3
2.2 Inversion of electromagnetic data

Many different inversion schemes for electromagnetic data have been introduced in the last decades. This section provides an overview of the most important methods that have been developed to invert EM data. The methods of Conjugate Gradients (CG) and Non-Linear Conjugate Gradients are introduced in Chapter 2.2.2 and described in full detail in Chapter 3.2, as they were the methods implemented for this thesis.

2.2.1 Non-linear inversion techniques

The aim of data inversion is to minimize the data misfit, i.e. the difference between the observed data $d^{obs}$ and the data returned by the model $m$, denoted as $F(m)$. Both $d^{obs}$ and $F(m)$ are vectors of length $N$ (number of data) holding the different data entries, whereas vector $m$ contains the model parameters and has length $M$ (number of model parameters). The total data misfit is often measured in Root Mean Square (RMS), written as

$$\text{RMS} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left( \frac{d^{obs}_i - F_i(m)}{\sigma_d^i} \right)^2}, \quad (2.1)$$

where $\sigma_d^i$ is the standard deviation of the $i$-th datum. Since field data always contain some noise, geophysicists do not want to fit the data completely, but rather compute a model that both fits the data sufficiently (e.g. in the range of the noise level) and preserves some model constraints at the same time. Reproducing all the data and noise exactly would lead to very rough models. By adding a regularisation term, the model can be smoothed to get a more realistic resistivity structure. The objective function that has to be minimized can then be expressed as

$$\Phi(m) = (d^{obs} - F(m))^T C_d^{-1} (d^{obs} - F(m)) + \lambda (m - m_{ref})^T L^T L (m - m_{ref}). \quad (2.2)$$

where $C_m$ is the model covariance matrix, $C_d$ is the data covariance matrix, $m_{ref}$ is a vector holding the reference (a priori) model parameters and $T$ is the transpose operator. The strength of the model regularisation can be adjusted by introducing a Lagrange multiplier $\lambda$. The inverse model covariance $C_m^{-1}$ is often chosen as a first or second order difference operator and will be denoted as $L^T L$ henceforth. This is also known as Tikhonov regularisation (Tikhonov and Arsenin, 1977). Thus, the magnitude of $\lambda$ determines how smooth the model will be. Equation (2.2) is often referred to as the penalty function. It can be extended with a target misfit $X_2^2$ to which the inversion should converge. Some inversion techniques adjust the parameter $\lambda$ during the inversion process. The objective function can then be formulated as a function of both the model parameters and $\lambda$:

$$\Psi(m, \lambda) = \left\{ (d^{obs} - F(m))^T C_d^{-1} (d^{obs} - F(m)) - X_2^2 \right\} + \lambda (m - m_{ref})^T L^T L (m - m_{ref}). \quad (2.3)$$

The predicted data vector $F(m)$ is non-linear for EM problems. This means that the inversion cannot be solved in one step. Instead, $F(m)$ is linearised and the problem is solved in multiple iterations. The linearisation of non-linear inverse problems is usually done with a first order Taylor expansion of $F(m)$,
written as

\[ F(m_{k+1}) \approx F(m_k) + \left[ \frac{\partial F_i(m)}{\partial m_j} \right]_{m=m_k} (m_{k+1} - m_k) \]

\[ = F(m_k) + J(m_{k+1} - m_k), \quad (2.4) \]

where \( J \) is the Jacobian matrix or sensitivity matrix containing the partial derivatives of the \( i \)-th datum with respect to (w.r.t.) the \( j \)-th model parameter and subscript \( k \) designates the iteration number, meaning that the left-hand side of Equation (2.4) is the predicted data vector of the updated model. By inserting Equation (2.4) into Equations (2.2) or (2.3), an objective function that is quadratic in \( m_{k+1} \) is obtained. It is then possible to calculate the model \( m_{k+1} \) that minimises this quadratic function.

The Gauss-Newton (GN) method is derived when the linearisation of \( F(m) \) is applied to Equation (2.2). It is strongly related to Newton’s method to find stationary points of non-linear functions. The difference of GN to Newton’s method is that GN approximates the Hessian (i.e. the second derivative of Equation (2.2)), whereas Newton’s method computes the full Hessian (it is based on a second order Taylor expansion of Equation (2.2)). As mentioned earlier, the non-linear problem is solved successively by linear approximation. For each step, the GN inversion model is updated by solving

\[ \left[ J^T C_d^{-1} J + \lambda L^T L \right] \Delta m = J^T C_d^{-1} (d_{obs} - F(m_k)) - \lambda L^T L (m_k - m_{ref}), \quad (2.5) \]

where \( \Delta m \) is the model update step, i.e. the difference between the new model \( m_{k+1} \) and the previous model \( m_k \). The Newton and GN methods have been tested in a number of studies, which are summarized in Avdeev (2005) and Siripunvaraporn (2012). Newman and Hoversten (2000) applied Newton’s method with the full Hessian and presented solutions on how to solve Newton iterations for two- and three-dimensional inverse problems efficiently. Different concepts on how to apply Newton’s method and GN for EM inversion have been proposed and compared by Haber et al. (2000).

A Gauss-Newton based technique that is often used to invert EM data is Occam’s inversion algorithm, introduced by Constable et al. (1987) in one dimension and extended to two dimensions by deGroot Hedlin and Constable (1990). As for GN, the objective function that has to be minimized consists of a data misfit term and a model regularisation term. The latter can take the form of first or second order Tikhonov regularisation (as in Equation (2.3)). The new model can then be expressed as

\[ m_{k+1} = \left[ J^T C_d^{-1} J + \lambda L^T L \right]^{-1} J^T C_d^{-1} \hat{d} + m_{ref}, \quad (2.6) \]

where \( m_{k+1} \) is the updated model and \( \hat{d} = d_{obs} - F(m_k) + J(m_k - m_{ref}) \). Note that Equation (2.6) is a simple modification of Equation (2.5). However, whereas the Lagrange multiplier is held constant for GN, it is changed during each iteration for Occam’s inversion. The aim is to find a value of \( \lambda \) so that both the desired data misfit is reached and the objective function is minimised. Hence, the inversion process is divided into two parts. First, the Lagrange multiplier is chosen so that the data misfit is minimised. When the desired misfit is reached, the algorithm tries to smooth the model without increasing the misfit of the data.

When first or second order Tikhonov regularisation is used, as it is the case for the GN based inversion methods, the objective function is not constrained to an unique minimum. The smoothness term \( L^T L \) defines a semi-norm, meaning that a set of solutions exist. This problem is solved by constraining the
inversions with Marquardt-Levenberg (ML) damping (e.g. Lines and Treitel, 1984), which enforces the inversions to converge. The ML term ensures that the model update step is small, i.e. that the model does not change too much in one iteration. By using both a smoothness and a Marquardt-Levenberg term, known as damped Occam inversion, the objective function can be rewritten as

$$\Psi_{damp}(m_{k+1}, \beta) = (d^{obs} - F(m_{k+1}))^T C_d^{-1}(d^{obs} - F(m_{k+1}))$$

$$+ \lambda (m_{k+1} - m_{ref})^T L^T L (m_{k+1} - m_{ref}) + \beta (m_{k+1} - m_k)^T I (m_{k+1} - m_k),$$

(2.7)

where $I$ is the identity matrix. Note that the objective function in Equation (2.7) is dependent on $\beta$ and not as in Occam's inversion on parameter $\lambda$. This means that $\lambda$ is kept fixed during the entire inversion process, whereas $\beta$ is adjusted in each iteration. By applying the first order Taylor expansion of $F(m)$, the model that minimises the quadratic approximation of Equation (2.7) can be computed by

$$m_{k+1} = [J^T C_d^{-1} J + \lambda L^T L + \beta I]^{-1} \{J^T C_d^{-1} \hat{d} + \beta (m_k - m_{ref})\} + m_{ref}.$$  

(2.8)

Damped Occam inversion was for instance applied to MT data by Kalscheuer et al. (2012). If $\beta$ is chosen to be large, Equation (2.8) reduces to the steepest descent step, which simply follows the negative gradient of $\Psi(m)$. This ensures certain convergence but is often very slow because it can take bad directions to reduce the objective function (Shewchuk, 1994). If $\beta$ is chosen to be small, Equation (2.8) approximates the GN method, which leads to rapid but uncertain convergence (Lines and Treitel, 1984, Aster et al., 2013) (see Figure 2.1).

![Figure 2.1: The geometry of Marquardt-Levenberg, Gauss-Newton and Steepest Descent search directions for two model parameters. The starting point is $(m_{1,start}, m_{2,start})$ and the minimum is located at $(m_{1,min}, m_{2,min})$ (adapted from Lines and Treitel, 1984).](image)

All the methods presented above contain three rather expensive operations. First, for a complete derivation of $J_{\min}(N,M)$ forward problems have to be solved (Rodi (1976) and Appendix B.1). Secondly, the matrix product $J^T C_d^{-1} J$ has to be computed, which requires high computation times, especially for
large $M$ and $N$. Additionally, linear systems of equations have to be solved to obtain the updated models, i.e. direct matrix inversions are involved. This is for instance done with the Cholesky factorisation\(^3\), which contributes strongly to the computation time. Therefore, efforts have been made to accelerate or circumvent these operations. One possibility is to approximate the sensitivity matrix. A famous technique to do this was introduced by Smith and Booker (1991). They performed separate inversions for each measurement site and updated the resistivity model by interpolating the different profiles. In many problems, especially for 2D and 3D, $M$ gets much higher than $N$. Thus, it is time-saving to perform the inversion in the data space rather than in the model space. Such a methodology is implemented in REBOCC. Occam’s inversion is transferred into data space and the system of equations is reduced from $M \times M$ to $N \times N$. This technique is referred to as data space Occam inversion ($\text{Dasocc}$). In REBOCC, further computation time is saved by calculating a subset sensitivity matrix instead of the entire Jacobian matrix. Siripunvaraporn and Egbert (2000) compute $J$ at a subset of the employed frequencies or stations and interpolate them to form an approximative sensitivity matrix. This is a reasonable approximation because MT data and their derivatives w.r.t. the model parameters are smooth in general (Siripunvaraporn and Egbert, 2000). During the last ten years, quasi-Newton (QN) techniques have been developed to solve multiple dimension large-scale inversions. These methods need less computation time than conventional Newton methods, because they do not require to invert the Hessian or approximate Hessian matrix, but instead update the inverse Hessian by using gradient information of the objective function from previous iterations. Furthermore, the Jacobian does not have to be calculated completely, because only column-wise information of $J$ is needed for the gradients (cf. Chapter 3.2.2 and Appendix B.2). Limited-memory QN techniques have been developed for EM inverse problems to further decrease computation times. For these methods, the Hessian is updated by using a subspace of the model only. With that, plenty of storage can be saved (Avdeev and Avdeeva, 2009, among others). Other time efficient techniques that circumvent direct matrix inversions and the computation of the Jacobian are the methods of Conjugate Gradients. They will be introduced in the next subsection.

### 2.2.2 Inversion with Conjugate Gradient techniques

A large set of inversion schemes belong to the Krylov subspace methods. They solve the linear system of equations

$$Ax = b$$

by inverting the system matrix $A$ in an iterative process. Krylov subspace methods use a practical property of $A$, namely that its inverse can be expressed as a linear combination of its powers (Shewchuk, 1994). The most often applied Krylov subspace technique for EM problems is the method of Conjugate Gradients (CG). This technique can be applied to invert symmetric and positive-definite matrices, such as the matrices that have to be inverted in Equations (2.5), (2.6) and (2.8). It is the iterative equivalent to the Cholesky factorisation, which inverts the matrix directly. Therefore, CG can be combined with other inversion techniques (e.g. GN based methods) to solve the system of linear equations for the model update step. CG was successfully applied to invert EM data in two and three dimensions (Mackie and Madden, 1993, Newman and Alumbaugh, 1997, Rodi and Mackie, 2001, Siripunvaraporn and Egbert, 2000).

\(^3\)The Cholesky factorisation or decomposition divides the matrix (Hermitian, positive-definite) that has to be inverted into a lower triangular matrix $L$ and its conjugate transpose $L^*$ so that $A = LL^*$. The matrix can then be inverted by forward and backward substitution.
2007). CG was first introduced by Hestenes and Stiefel (1952). A broad overview of the history and developments of CG is given by O’Leary (1998). Due to its wide applications, it is a very well studied method. Many mathematicians analysed the convergence properties of CG (Daniel, 1967, Gilbert and Nocedal, 1992, among others). Faber and Manteuffel (1984) studied the conditions of the system matrix \( A \) for which CG has solutions. An important property of CG is that it always converges to a minimum. It reaches the minimum after a maximum of \( n \) steps, where \( n \) is the length of vector \( x \). For EM inversion problems, \( n \) corresponds to \( M \), the number of model parameters. It is most often not necessary to solve the linear problem in Equation (2.9) completely. Rodi and Mackie (2001) showed that halting the CG inversion after three steps is already sufficient to update the GN model step. Therefore, the method of CG outperforms direct matrix inversion techniques such as the Cholesky factorisation for large-sized \( A \).

However, an important requirement for fast convergence of CG is that \( A \) has to be well-conditioned, meaning that its condition number \( \kappa = \frac{\lambda_{\text{max}}(A)}{\lambda_{\text{min}}(A)} \) is small, where \( \lambda_{\text{max}} \) and \( \lambda_{\text{min}} \) denote the maximum and minimum eigenvalues of matrix \( A \). For an ill-conditioned system matrix (e.g. large condition number), a pre-conditioner can be used to lower the condition number of \( A \). Preconditioning will be explained in Chapter 3.2.4.

Because of the non-linearity of EM inverse problems, the method of CG can only be used to minimise linear model update steps. The method of Non-Linear Conjugate Gradients (NLCG) can be applied to solve the entire inversion problem. The NLCG method was first introduced by Fletcher and Reeves (1964). They extended the linear CG method to non-linear problems by adjusting some elements of CG (cf. Chapter 3.2). Polak and Ribiere (1969) analysed the performance of the algorithm of Fletcher and Reeves (1964) and proposed a slightly different scheme, which showed improved convergence. The NLCG versions of Fletcher and Reeves (1964) and Polak and Ribiere (1969) will be further denoted as FR and PR, respectively. For EM problems, NLCG (with PR) was initially implemented to invert MT data in 2D by Rodi and Mackie (2001) and in 3D by Newman and Alumbaugh (2000). The algorithms of these authors served as a base for more recent applications of NLCG by the EM community, for instance to invert CSEM (Mackie et al., 2007, Newman et al., 2010), MT (Mackie et al., 2007) and very-low frequency (VLF) data (Kamm and Pedersen, 2014).

The main advantage of NLCG compared to other inversion techniques such as Occam and damped Occam is its convergence rate per time. It avoids the computation of the complete Jacobian matrix and does not require to invert the approximate Hessian and is therefore very fast. The only element that is needed to define the search direction is the gradient of the penalty function (Equation (2.2)), which can be expressed as

\[
g(m) = -2J^T C_d^{-1}(d_{\text{obs}} - F(m)) + 2\lambda L^T L m. \tag{2.10}
\]

Note that \( \lambda \) is normally not changed for the method of NLCG. It can be seen in Equation (2.10) that NLCG only needs to calculate a matrix-vector product of the Jacobian. Rodi (1976) presented a procedure how to compute the matrix-vector product directly and thereby avoiding the expensive computation of the entire sensitivity matrix. This reduces the number of forward problems from \( \min(N,M) \) to one per frequency. For high \( M \) and \( N \), this can strongly decrease computation times. The method of NLCG takes smaller steps than for instance GN and thus, it takes a lot more iterations to reach the minimum of the objective function. However, the different steps can be computed much faster. Rodi and Mackie (2001) have shown that NLCG only needs a fraction of the computation time of GN. As for the linear case, preconditioning can be applied to NLCG to further accelerate convergence. Tests have shown that
preconditioning strongly accelerates the inversion process (Newman and Alumbaugh, 2000, Rodi and Mackie, 2001). Detailed explanations to the implementation of NLCG are given in Chapter 3.2.
3 Method

3.1 Electromagnetic theory

This section will provide an overview of basic EM theory and modelling. Maxwell’s equations are presented and simple laws for electromagnetic field propagation are derived (Chapter 3.1.1). The descriptions are focused on MT and RMT methods in two dimensions, because I inverted such data for this project. An introduction to forward modelling and sensitivity computation is given in Chapter 3.1.2 and 3.1.3, whereas detailed derivations can be found in Appendices A and B, respectively. The explanations in this section are based on Ward and Hohmann (1988) and Kalscheuer et al. (2008).

3.1.1 Electromagnetic field propagation

The aim of electromagnetic prospecting is to measure the response of conductors and resistors in the subsurface to EM fields generated by controlled or external sources. We can derive expressions for the electric and magnetic fields as functions of the resistivity structure in the ground. In 2D, this can be solved numerically with finite-difference methods (cf. Appendix A). The basic set of equations that are needed to derive the relations of the subsurface resistivities with the EM fields are Maxwell’s equations. They consist of four laws that relate electric and magnetic fields to currents and charges. For this thesis, I will perform the derivations in frequency domain with harmonic time dependence $e^{i2\pi ft}$, with $f$ the frequency and $i = \sqrt{-1}$ the imaginary number. Maxwell’s equations can be written as

$$\nabla \times \mathbf{E} + \hat{z}\mathbf{H} = 0 \quad \text{(Faraday’s law)} \quad (3.1)$$

$$\nabla \times \mathbf{H} - \hat{y}\mathbf{E} = 0 \quad \text{(Ampere’s law)} \quad (3.2)$$

$$\nabla \cdot \varepsilon \mathbf{E} = q \quad \text{(Gauss’ law for electricity)} \quad (3.3)$$

$$\nabla \cdot \mathbf{H} = 0 \quad \text{(Gauss’ law for magnetism)} \quad (3.4)$$

where $\mathbf{E}$ is the electric field vector, $\mathbf{H}$ the magnetic field vector, $\hat{z} = i\mu \omega$ the impedivity, $\hat{y} = \sigma + i\varepsilon \omega$ the admittivity and $q$ the charge density. Parameter $\varepsilon$ is the dielectric permittivity, $\mu$ the magnetic permeability, $\sigma$ the electric conductivity and $\omega = 2\pi f$ the angular frequency. Note that the right-hand sides of Equations (3.1) and (3.2) are zero, meaning that we assume source-free media. Note further that the resistivity $\rho$, that is the actual model parameter with unit $\Omega m$, is the inverse of the conductivity $\sigma = \frac{1}{\rho}$.

Equations (3.1) to (3.4) are coupled through the so-called constitutive relations

$$\mathbf{D} = \varepsilon \mathbf{E} \quad \text{(Dielectric displacement)} \quad (3.5)$$

$$\mathbf{B} = \mu \mathbf{H} \quad \text{(Magnetic induction)} \quad (3.6)$$

$$\mathbf{J} = \sigma \mathbf{E} \quad \text{(Electric current density).} \quad (3.7)$$

In order to explain the propagation of EM fields, homogeneous media are assumed in the first instance. Furthermore a number of other assumptions are usually made for electromagnetic earth problems (Ward and Hohmann, 1988), such as

1. isotropic media with time-, temperature- and pressure-independent electrical properties, and
2. $\mu$ is the magnetic permeability of free space, i.e. $\mu = \mu_0 = 4\pi \times 10^{-7}Hm^{-1}$.

Taking these assumptions into account, a new set of equations can be derived from Equations (3.1) to (3.7), called Helmholtz equations. They can be expressed as

$$\nabla^2 E + k^2 E = 0 \quad (3.8)$$

and

$$\nabla^2 H + k^2 H = 0, \quad (3.9)$$

in which

$$k^2 = \mu_0 \varepsilon \omega^2 - i\mu_0 \sigma \omega = -\hat{z} \hat{y}. \quad (3.10)$$

Equations (3.8) and (3.9) are wave equations with wave number $k$ and describe how electric and magnetic fields propagate through space. For MT and RMT, the source is often far away from the receiver (far-field assumption) and hence, planar wave solutions of the Helmholtz equations can be considered. Frequency plays a crucial role in EM wave propagation. For MT problems with frequencies between 0.001 and 10 Hz, $\mu_0 \varepsilon \omega^2 \ll \mu_0 \sigma \omega$ and thus $k^2 \approx -i\mu_0 \sigma \omega$ or $\hat{y} \approx \hat{z}$. This is called the quasi-static approximation, i.e. displacement currents are ignored. However, for frequencies greater than 100 kHz, this assumption does no longer hold (Ward and Hohmann, 1988). Hence, in the upper frequency limit of RMT (from 100 to 300 kHz), displacement currents have to be taken into account (Kalscheuer et al., 2008).

For EM modelling, we can rotate the three-dimensional coordinate system $(x, y, z)$ such that the impinging wave is travelling in the $y$ and $z$ only. Thus, taking the $x$-direction as the geoelectrical strike direction, the plane waves are incident in the $y$-$z$ plane, meaning that the EM field components do not vary with $x$ (i.e. $\frac{\partial}{\partial x} = 0$). This allows to split the problem into two modes, namely the transverse electric (TE) and the transverse magnetic (TM) modes. The incident electric field in the TE mode and the incident magnetic field in the TM mode have $x$-components only. This reduces Maxwell’s equations to two sets of equations, written as

$$\begin{align*}
\frac{\partial H_x}{\partial y} - \frac{\partial H_y}{\partial z} &= \hat{y} E_x, \\
\frac{\partial E_y}{\partial z} &= -\hat{z} H_y, \\
\frac{\partial E_z}{\partial y} &= \hat{z} H_z
\end{align*}$$

TE mode \quad (3.11a) \quad (3.11b) \quad (3.11c)

and

$$\begin{align*}
\frac{\partial E_x}{\partial y} - \frac{\partial E_y}{\partial z} &= -\hat{z} H_x, \\
\frac{\partial H_x}{\partial z} &= \hat{y} E_y, \\
\frac{\partial H_y}{\partial y} &= -\hat{y} E_z
\end{align*}$$

TM mode \quad (3.12a) \quad (3.12b) \quad (3.12c)

For a homogeneous medium, the general solution of the Helmholtz equations for the TE mode is

$$E_x = (E_x^+ e^{-ikz} + E_x^- e^{ikz}) (e^{-iky} + e^{iky}) \quad (3.13)$$
and for the TM mode

\[ H_x = \left( H_x^+ e^{-ik_y y} + H_x^- e^{ik_y y} \right) \left( e^{-ik_z z} + e^{ik_z z} \right), \]  

(3.14)

where the superscripts + and − describe the up and down going wave, respectively, and \( k_y \) and \( k_z \) are the wave number components in the corresponding directions with \( k^2 = k_y^2 + k_z^2 \). As it can be readily seen in Equation (3.10), \( k \) is a complex quantity with a propagating and an attenuating part. Expressing the wave number as \( k = \alpha - i\beta \), we can derive

\[ \alpha = \omega \sqrt{\frac{\mu_0 \varepsilon_0}{2} \left( \sqrt{1 + \frac{\sigma^2}{\varepsilon_0^2 \omega^2}} - 1 \right)}, \]  

(3.15)

and

\[ \beta = \omega \sqrt{\frac{\mu_0 \varepsilon_0}{2} \left( \sqrt{1 + \frac{\sigma^2}{\varepsilon_0^2 \omega^2}} + 1 \right)}, \]  

(3.16)

where \( \alpha \) and \( \beta \) represent propagation and attenuation, respectively. In the quasi-static approximation, Equations (3.15) and (3.16) reduce to \( \alpha = \beta = \sqrt{\frac{\omega \mu_0 \sigma}{2}} \). The skin depth \( \delta \), where the amplitude of the incident EM field has been attenuated by a factor of \( 1/e \approx 0.37 \), can then be written as

\[ \delta = \sqrt{\frac{2}{\omega \mu_0 \sigma}} \approx 503 \sqrt{\frac{1}{f \sigma}}. \]  

(3.17)

Thus, the higher the frequency of the EM wave, the lower the penetration depth. Furthermore, good conductors attenuate the EM fields much more than resistive media.

For 2D modelling, where resistivities change in \( y \) and \( z \), the EM wave equations are not as simple as for homogeneous media. The electric field component \( E_x \) of the TE mode can be directly inserted into the homogeneous Helmholtz equations, because \( \hat{z} \) from Faraday’s law (Equation (3.1)) does not vary in space. Ignoring the \( \frac{\partial}{\partial z} \) terms (strike direction) this leads to

\[ \frac{\partial^2 E_x}{\partial y^2} + \frac{\partial^2 E_x}{\partial z^2} = 2\hat{y} E_x. \]  

(3.18)

However, for the TM mode, \( \hat{y} \) in Ampere’s law is not constant, because \( \sigma \) varies both in \( y \) and \( z \). Deriving the Helmholtz equation for the magnetic field component \( H_x \) yields therefore

\[ \frac{1}{\hat{y}} \left( \frac{\partial^2 H_x}{\partial y^2} + \frac{\partial^2 H_x}{\partial z^2} \right) + \frac{\partial}{\partial y} \left( \frac{1}{\hat{y}} \right) \cdot \frac{\partial H_x}{\partial y} + \frac{\partial}{\partial z} \left( \frac{1}{\hat{y}} \right) \cdot \frac{\partial H_x}{\partial z} = \hat{z} H_x. \]  

(3.19)

It is Equations (3.18) and (3.19) we want to solve with the FD approach. After obtaining \( E_x \) and \( H_x \), we can solve for the corresponding magnetic field component \( H_y \) with Equation (3.11b) for the TE mode and for the electric component \( E_y \) with Equation (3.12b) for the TM mode. What we then get are the 2D impedance tensor elements \( Z_{xy} \) and \( Z_{yx} \) from

\[ \begin{bmatrix} E_x \\ E_y \end{bmatrix} = \begin{bmatrix} 0 & Z_{xy} \\ Z_{yx} & 0 \end{bmatrix} \begin{bmatrix} H_x \\ H_y \end{bmatrix}. \]  

(3.20)
The impedances can be used to calculate apparent resistivities

\[
\rho_{\text{app}}^{xy} = \frac{1}{\omega \mu_0} |Z_{xy}|^2, \quad \rho_{\text{app}}^{yx} = \frac{1}{\omega \mu_0} |Z_{yx}|^2
\] (3.21)

and phases

\[
\phi^{xy} = \arg(Z_{xy}), \quad \phi^{yx} = \arg(Z_{yx}),
\] (3.22)

where the left-hand equations are for the TE mode and the right-hand equations for the TM mode. Thus, two sets of data are obtained. The reason of measuring both in TE and TM is that for a heterogeneous resistivity model, the impedances \(Z_{xy}\) and \(Z_{yx}\) do not match anymore. Actually, \(Z_{xy} = -Z_{yx}\) is only valid for a layered half-space in quasi-static approximation, because in this case the transmission angle of the EM wave is always vertical and independent on the angle of incidence (Kalscheuer et al., 2008). For 2D modelling however, the difference of \(Z_{xy}\) and \(Z_{yx}\) can get significant. Thus, to retrieve the most realistic model, TE and TM data are often inverted simultaneously in a joint inversion process. Another option is to invert the determinant of the 2D impedance tensor, expressed as

\[
Z_{\text{det}} = \sqrt{Z_{xx}Z_{yy} - Z_{xy}Z_{yx}}.
\] (3.23)

The subscript \(\text{det}\) stands for determinant mode (DET), which is rotationally invariant and less affected by possible 3D structures. Thus, the DET mode can provide more reliable results in two dimensions compared to TE and TM joint inversion (Pedersen and Engels, 2005). In EMILIA, TE and TM data as well as data from the determinant mode can be inverted. Furthermore, all these data can be inverted simultaneously by joint inversion.

### 3.1.2 Forward problem

Before we can numerically solve Equations (3.18) and (3.19), it is necessary to split the investigation area into a mesh with \((N_{zb} + N_{za}) \times N_y\) cells, where \(N_y\) is the number of cells in the horizontal direction and \(N_{zb}\) and \(N_{za}\) denote the number of cells in the vertical direction for the subsurface and the air, respectively. A resistivity value is assigned to each cell. For RMT, the dielectric permittivity could also be taken as a model parameter and consequently, this would double the total number of model parameters. However, Kalscheuer et al. (2008) have shown that the error that originates from assuming constant \(\varepsilon\) for all cells is often small. Within the scope of this thesis, the data is therefore inverted for resistivities only. For simplicity, one can choose the starting model to be homogeneous, i.e. all the cells in the subsurface have the same resistivity. The \(x\)-components of the EM fields are then calculated for the nodes of the mesh. We place the mesh so that the stations are located on the nodes. The task is then to evaluate \(\rho_{\text{app}}\) and \(\phi\) at the station nodes (denoted as \(F(m)\) in Chapter 2.2.1) from the given model. To get accurate results, it is important to scale the mesh appropriately. For instance, if some prior information about the resistivity structure is available, it is reasonable to reduce the size of the mesh cells in regions of high resistivity contrasts. Either way, the size of the mesh cells underneath the stations should be reduced, because data are most sensitive to these regions. In general, the smallest cells should be considerably smaller than \(\frac{1}{\alpha}\) (cf. Equation (3.15)) of the highest frequency (Kalscheuer et al., 2008). In addition, the total size of the mesh has to be deep and wide enough. A good measure of the mesh size is the skin depth (Equation
of the lowest measured frequency. Note however, in the RMT case, the attenuation is defined through Equation (3.16) and hence, the skin depth is larger than for the quasi-static case and the mesh has to be extended. A summary over the main differences between MT and RMT modelling is given in Table 3.1.

### Table 3.1: Important discrepancies between MT and RMT forward modelling.

<table>
<thead>
<tr>
<th></th>
<th>MT (quasi-static approximation)</th>
<th>RMT (with displacement currents)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angle of transmission</td>
<td>independent on angle of incidence because the total wavenumber ( k \approx k_z ), i.e. wave is transmitted vertically downward</td>
<td>theoretically dependent on angle of incidence because ( k ) has components in both ( y ) and ( z ), but assumption ( k \approx k_z ) holds when a conductive layer is present near the surface (Kalscheuer et al., 2008)</td>
</tr>
<tr>
<td>Mesh width and depth</td>
<td>chosen according to skin depth ( \delta = \sqrt{\frac{2}{\omega \mu_0 \sigma}} )</td>
<td>chosen according to ( \alpha ) from Equation (3.15) and ( \delta = \frac{1}{\beta} ) from Equation (3.16) (note that ( \delta_{RMT} \geq \delta_{MT} ))</td>
</tr>
<tr>
<td>Air half-space for mesh</td>
<td>only for TE mode, no conduction currents in the air</td>
<td>for TE and TM mode due to displacement currents in the air</td>
</tr>
<tr>
<td>Electric permittivity ( \varepsilon )</td>
<td>( \hat{\varepsilon} \approx \sigma ), thus ( \varepsilon ) is not included in currents</td>
<td>in EMILIA, ( \varepsilon ) can be varied arbitrarily over the cells. In this thesis, ( \varepsilon ) is chosen to be constant: ( \varepsilon_r = 5^1 )</td>
</tr>
</tbody>
</table>

The actual FD forward problem is a linear system of equations in the form

\[
Kx = s, \tag{3.24}
\]

with \( K \) the system matrix containing electrical and magnetic parameters \( \hat{y} \) and \( \hat{z} \) and the finite differences (cf. Appendix A for details), vector \( x \) the components \( E_x \) (in TE mode) or \( H_x \) (in TM mode) for the different mesh nodes and \( s \) the boundary values. For the TE mode, the mesh cells have to be extended into the air because small conductive currents are assumed in the air (\( \sigma_{air} = 10^{-10} \text{Sm}^{-1} \)). For the quasi-static case in the TM mode, there are no conduction currents in the air (i.e. from Equation (3.7) \( J = (0, \sigma E_y, \sigma E_z) \)), because we assume \( \sigma_{air} \approx 0 \text{Sm}^{-1} \). This leads to \( H_x = \text{const.} \) from Equations (3.12b) and (3.12c) and hence, the air half-space can be omitted. However, with displacement currents, \( \hat{y} \) is no longer zero and \( H_x \) is not constant anymore. This means that we have to extend the mesh into the air for the TM mode in the RMT case. At the top of the mesh, the particular components of vector \( s \) are set to unit amplitude and no phase (e.g. \( 1 + 0i \)). Further, at the two lateral boundaries, \( s \) is set to the \( E \) and \( H \) field values generated by the \( \hat{y} \)-values along the corresponding edge. The EM fields are assumed to have decayed completely at the lower bound of the mesh. In EMILIA, the forward problem in Equation (3.24) is solved by the method of preconditioned Conjugate Gradients (PCG) (cf. Subsections 3.2.1 and 3.2.4).

\[^1\varepsilon_r = \frac{\varepsilon}{\varepsilon_0} \text{ is the relative dielectric permittivity, with } \varepsilon_0 = 8.89 \times 10^{-12} \text{ Fm}^{-1}.\]
3.1.3 Sensitivity matrix

For the inversion process (cf. Chapter 3.2), the change of the data predicted by the model w.r.t. the model parameters is needed to define the updated model. This leads to the sensitivity matrix with \(N \times M\) entries written in matrix form as

\[
J = \begin{bmatrix}
\frac{\partial F_1(m)}{\partial m_1} & \frac{\partial F_1(m)}{\partial m_2} & \cdots & \frac{\partial F_1(m)}{\partial m_M} \\
\frac{\partial F_2(m)}{\partial m_1} & \frac{\partial F_2(m)}{\partial m_2} & \cdots & \frac{\partial F_2(m)}{\partial m_M} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial F_N(m)}{\partial m_1} & \frac{\partial F_N(m)}{\partial m_2} & \cdots & \frac{\partial F_N(m)}{\partial m_M}
\end{bmatrix}.
\]  

(3.25)

The elements of vector \(m\) are given in logarithmic resistivities to assure positive resistivity values for the entire model. The logarithms are defined to the base of 10 in EMILIA. Thus, the conversion \(\rho_j = \log_{10}(\rho_{j,linear}^{1\Omega m})\) is performed. The data vector \(F(m)\) is given partly in logarithmic form for the apparent resistivities, i.e. \(\rho_{app}^i = \log_{10}(\rho_{app,linear}^{1\Omega m})\), whereas the phase is treated linearly. The computation of \(J\) is done in logarithmic form, too. The detailed derivation of the sensitivity matrix elements as well as the computation of the sensitivity matrix-vector products to circumvent the full evaluation of \(J\) are given in Appendix B.

3.2 Inversion with Non-Linear Conjugate Gradients

In this section, the theory of NLCG will be explained and a flow-chart, which describes the implementation of NLCG in EMILIA, will be presented. The algorithm of linear CG is stated at first. The algorithm for NLCG is then derived from the linear procedure. The following explanations are based on Shewchuk (1994) and Nocedal and Wright (2006).

3.2.1 Linear Conjugate Gradients

The aim of CG is to solve Equation (2.9), in other words to reduce the residual vector

\[
r = Ax - b
\]

(3.26)
to a minimum. This is equivalent to minimise a quadratic function of the form

\[
\Phi_{quad}(x) = \frac{1}{2}x^TAx - b^Tx + c,
\]

(3.27)
because calculating the derivative of \(\Phi_{quad}\) w.r.t. \(x\) (i.e. the gradient) and setting it to zero leads to Equation (3.26). Thus, the residual is nothing else than the gradient of \(\Phi_{quad}\). CG minimises Equation (3.27) by solving the problem iteratively, i.e. by reducing the residual step by step. If the system matrix \(A\) has dimension \(n \times n\), CG reduces the residual dimension-wise until the solution is reached. The first requirement that CG can converge in \(n\) steps is that its residuals are linearly independent, i.e. mutually orthogonal:

\[
r_i^T r_j = 0, \text{ for all } i \neq j,
\]

(3.28)
where indices $i$ and $j$ denote the iteration number. The second requirement is that the search directions are conjugate w.r.t. $A$ (so-called A-orthogonality). This property can be expressed as

$$ p_i^T A p_j = 0, \text{ for all } i \neq j, \quad (3.29) $$

where $p$ is the search direction vector. Equation (3.29) implies that all $p$ are linearly independent. The third condition that has to be fulfilled is that the residuals need to be mutually orthogonal to all search directions, i.e.

$$ r_i^T p_j = 0, \text{ for all } i \neq j. \quad (3.30) $$

CG follows the set of linearly independent search direction vectors $(p_0, p_1, \ldots, p_{n-1})$ until Equation (3.26) is minimised. The solution $x$ is updated iteratively by setting

$$ x_{k+1} = x_k + \alpha_k p_k, \quad (3.31) $$

where $\alpha_k$ is the step size that has to be taken in each iteration to find the sub dimensional minimum of $x$. $\alpha_k$ can be derived from orthogonality properties of $p$ and $r$ (see Shewchuk (1994) for details) and can be expressed as

$$ \alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}. \quad (3.32) $$

The question remains how to determine the set of search directions. One possibility, known as the Gram-Schmidt approach, is to store all the old search directions and calculate new search directions by using the orthogonality property in Equation (3.29). The disadvantage of this approach is that all search directions have to be kept in mind. This is however very expensive with regard to the amount of computation, especially for large-sized systems of equations. In the method of CG, $p_k$ is a function of the residual and the previous search direction only and can be updated without knowing all the other search directions:

$$ p_{k+1} = -r_{k+1} + \beta_{k+1} p_k, \quad (3.33) $$

where $\beta$ is the step size needed to update $p_k$. $\beta$ can be derived as follows: Taking the A-orthogonality of all search directions into account, pre-multiplying Equation (3.33) by $p_k^T A$ leads to $\beta_{k+1} = \frac{r_{k+1}^T A p_k}{p_k^T A p_k}$. From Equation (3.31) we see that $\alpha_k A p_k = r_k - r_k$. By applying Equation (3.33) and the property from Equation (3.30), we get

$$ \beta_{k+1} = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}. \quad (3.34) $$

What is left is to determine a start direction $p_0$. This is done by taking the direction of steepest descent, i.e. the negative gradient or the residual, respectively. To define the new search direction (Equation (3.33)), the up-to-date residual is needed. As it can be seen in Equation (3.26), this involves the matrix-vector product $A x$. To avoid this rather expensive step, it is possible to combine Equations (3.26) and (3.31) to get

$$ r_{k+1} = r_k + \alpha_k A p_k \quad (3.35) $$

for the residual update. Note, that the matrix-vector product $A p_k$ has already been calculated and the residual can be updated at almost no additional cost. The method of CG can then be summarised in algorithmic form as follows:
Algorithm 1 CG

\( k \leftarrow 0 \)
choose a start model \( x_0 \)
set \( r_0 \leftarrow Ax_0 - b \)
set \( p_0 \leftarrow -r_0 \) (steepest descent)
\( \text{while } r_k^T r_k \geq tol \text{ and } k < k_{\text{max}} \) do

\[ q_k \leftarrow Ap_k \quad (3.36a) \]
\[ \alpha_k \leftarrow \frac{r_k^T r_k}{p_k^T q_k} \quad (3.36b) \]
\[ x_{k+1} \leftarrow x_k + \alpha_k p_k \quad (3.36c) \]
\[ r_{k+1} \leftarrow r_k + \alpha_k q_k \quad (3.36d) \]
\[ \beta_{k+1} \leftarrow \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k} \quad (3.36e) \]
\[ p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k \quad (3.36f) \]
\[ k \leftarrow k + 1 \quad (3.36g) \]
\( \text{end while} \)

If some predictions can be made about the final model, it is possible to choose \( x_0 \) so that it is close to the expected result. If not, a very easy choice is to set \( x_0 = 0 \). Some shortcuts can be taken to accelerate the algorithm. The scalar product \( r_k^T r_k \) has to be calculated several times. It can therefore be stored in an additional variable to save computation time. The residual is updated according to Equation (3.35). However, after many iterations, this update can deviate from the real residual due to accumulated floating point errors and hence, it is necessary to compute the residual by explicitly forming the matrix-vector product \( Ax_{k+1} \) (see Shewchuk (1994) for details). The CG algorithm runs as long as \( r_k^T r_k \) is higher than a certain tolerance \( tol \) or until it exceeds the maximum number of iterations \( k_{\text{max}} \). In practical problems, it is not reasonable to run the algorithm until \( r_k = 0 \), because there is often no exact solution and furthermore, it is usually not required to solve the linear system completely. It is hard to determine the best \( tol \) for specific problems. The algorithm above therefore also provides the possibility to halt the procedure after a certain number of steps \( k_{\text{max}} \). This can be very useful when a rough solution is sufficient, for example in combined GN and CG inversions (e.g. Rodi and Mackie, 2001). EMILIA solves the forward problem with \( k_{\text{max}} = 500 \) in a preconditioned form.

3.2.2 Non-Linear Conjugate Gradients

The method of NLCG differs from linear CG in two points. First, the step size \( \alpha_k \) cannot be defined by a general expression as in Equation (3.32), but instead a line search along the search direction has to be performed. Secondly, the residual will be replaced by the gradient of the corresponding non-linear function. If NLCG is applied to an EM inverse problem, the non-linear function is represented by the penalty function

\[
\Phi(m) = (d^{obs} - F(m))^T C_d^{-1} (d^{obs} - F(m)) + \lambda (m - m_{ref})^T L^T L (m - m_{ref}). \quad (3.37)
\]
This equation is equivalent to Equation (2.2) when $C_m^{-1} = L^T L$. Taking the derivative of Equation (3.37), we obtain the gradient

$$\nabla \Phi(m) = -2J^T C_d^{-1}(d_{\text{obs}} - F(m)) + 2\lambda L^T L m,$$

as already stated in Equation (2.10) in Chapter 2.2.2. The negative gradient is used as the start direction and is required to define the new search direction in each NLCG iteration. As most of the important steps were stated in the previous section for linear CG, I will proceed by presenting the algorithm of NLCG as it was implemented in EMILIA.

**Algorithm 2 NLCG**

$k \leftarrow 0$
choose a start model $m_0$
evalute $\nabla \Phi_0$ (the gradient)
set $p_0 \leftarrow -\nabla \Phi_0$ (steepest descent)
while $\text{RMS} \geq \text{tol}$ and $k < k_{\text{max}}$
do perform line search to get $\alpha_k$ (Chapter 3.2.3)
update model $m_{k+1} \leftarrow m_k + \alpha_k p_k$

evaluate $\nabla \Phi_{k+1}$, then

$$\beta_{k+1}^{\text{PR}} \leftarrow \max\left\{ \frac{\nabla \Phi_k^T (\nabla \Phi_{k+1} - \nabla \Phi_k)}{\nabla \Phi_k^T \nabla \Phi_k} ; 0 \right\}$$

update search direction $p_{k+1} \leftarrow -\nabla \Phi_{k+1} + \beta_{k+1}^{\text{PR}} p_k$

$k \leftarrow k + 1$

end while

Note that $\nabla \Phi_k$ is the short form of $\nabla \Phi(m_k)$ and will be used henceforth. The choice of start model $m_0$ is usually based on information from the site (e.g. borehole data) when applying Algorithm 2 to EM data. An easy and common way is to set all elements of $m_0$ to the same resistivity value, i.e. the start model is a half-space. The tolerance $\text{tol}$ that defines when the NLCG inversion is stopped is either an absolute or a relative error boundary. A convenient way is to halt the inversion when a certain absolute RMS value is reached (e.g. RMS=1.0). Another possibility is to stop the algorithm when the relative change of RMS between two subsequent iterations is below a certain value. The inversion can also be terminated after a maximum number of iterations $k_{\text{max}}$. The line search to define the step size $\alpha$ is an algorithm itself and will be described in the next subsection.

The parameter $\beta$ needs some further explanations. The algorithm stated above is written in the form of algorithm PR after Polak and Ribiere (1969). If $\beta$ were defined as for linear CG, the algorithm would be identical to the one suggested by Fletcher and Reeves (1964). Interchanging $\beta^{\text{PR}}$ with $\beta^{\text{FR}}$ would not affect convergence if the objective function were strongly convex and quadratic, but for arbitrary non-linear functions, numerical tests showed that PR converges faster than FR (Nocedal and Wright, 2006).
The disadvantage of PR is that negative $\beta$ can be obtained, which leads to a search direction that is no longer a descent direction. To ensure convergence for every iteration, negative $\beta$ can be reset to zero so that the algorithm is restarted in the direction of steepest descent. Many other techniques to define $\beta$ exist. An alternative to the PR algorithm with similar convergence properties is the method of Hestenes and Stiefel (HS) (Nocedal and Wright, 2006). A similar method compared to FR was introduced by (Dai and Yuan, 1999). Hager and Zhang (2005) suggested another technique to define $\beta$ with very promising results. Both the Dai-Yuan method (DY) and the Hager-Zhang method (HZ) ensure that $p_k$ is a descent direction (Nocedal and Wright, 2006). The different techniques to update $\beta$ can be expressed as

$$\beta_{k+1}^{FR} = \frac{\nabla \Phi_k^T \nabla \Phi_{k+1}}{\nabla \Phi_k^T \nabla \Phi_k},$$  

(3.40a) $$\beta_{k+1}^{DY} = \frac{\nabla \Phi_k^T \nabla \Phi_{k+1}}{\hat{v}_k^T p_k},$$  

(3.40b) $$\beta_{k+1}^{HS} = \frac{\nabla \Phi_k^T \hat{v}_k}{\hat{v}_k^T p_k},$$  

(3.40c) $$\beta_{k+1}^{HZ} = \left(\hat{v}_k - 2p_k \frac{\hat{v}_k^T \hat{v}_k}{\hat{v}_k^T p_k}\right) \nabla \Phi_{k+1}^T \frac{\hat{v}_k^T p_k}{\hat{v}_k^T p_k}.$$  

(3.40d)

where $\hat{v}_k = \nabla \Phi_{k+1} - \nabla \Phi_k$. All these techniques are implemented in the NLCG algorithm of EMILIA and can be chosen by the user.

### 3.2.3 Line search

The line search to find the best step size $\alpha$ is, besides preconditioning, the most important element in the NLCG algorithm. The aim of the line search is to define the minimum point of $\Phi(m)$ along the search direction $p_k$. It can be very time-consuming to find the exact minimum (i.e. $\nabla \Phi^T p_k = 0$), because for each calculation of the gradient, a forward and two additional pseudo-forward problems have to be solved (cf. Appendix B.2). Therefore, a trade-off has to be found between determining the minimum along $p_k$ and computing efficiently, i.e. solving a minimum of forward problems. It has been shown that a sufficient decrease of $\Phi(m)$ per NLCG iteration is enough to get fast convergence (Newman and Alumbaugh, 2000, Rodi and Mackie, 2001). By sufficient decrease, I mean that the first Wolfe condition is met, namely that

$$\Phi(m_k + \alpha p_k) \leq \Phi(m_k) + c_1 \alpha \nabla \Phi_k^T p_k.$$  

(3.41)

This implies that the reduction in $\Phi(m)$ should be proportional to the step length $\alpha$ and the scalar product of the gradient of $\Phi(m)$ with the search direction. To allow for a more precise line search, one can use the second Wolfe condition to add constraints to the gradient itself:

$$|\nabla \Phi(m_k + \alpha p_k)^T p_k| \leq c_2 |\nabla \Phi_k^T p_k|.$$  

(3.42)

The constant $c_2$ defines how much smaller the gradient at $m_k + \alpha p_k$ has to be compared to the gradient at the start point $m_k$. Of course, $c_2 < 1$ because we want a reduction of the gradient. In EMILIA, $c_1$ and $c_2$ can be defined by the user, but $c_1 = 10^{-4}$ and $c_2 = 0.1$ are recommended by the literature (Nocedal and Wright, 2006) and implemented as default. The constraints of the first and second Wolfe condition are illustrated in Figure 3.1. The main problem of the line search is to find a trial step $\alpha_{trial}$ that leads to the vicinity of the minimum, so that sufficient decrease is obtained immediately or only
Figure 3.1: The first (blue) and second Wolfe condition (red). The double arrows at the bottom of the figure indicate the acceptable ranges for the step size $\alpha$ (adapted from Nocedal and Wright, 2006).

after a few additional steps. Prominent suggestions have been made in the field of EM inversion. Rodi and Mackie (2001) proposed to define the trial step by the linear CG step (Equation (3.32)). This is a save step when the objective function is close to quadratic, but can be misleading for strongly non-linear functions. However, for EM inversions this is often a good choice (Rodi and Mackie, 2001). Newman and Alumbaugh (2000) suggested to define the trial step by imposing some constraints on the maximum change of the model parameters. They derive an expression for $\alpha_{\text{trial}}$ as a function of $m_{\text{max}}$, the model parameter that corresponds to the largest component in $p_k$. They then perform an additional step, even if the first Wolfe condition is met after the trial step. Using information of the objective function at zero and at the trial step distance as well as the gradient at zero (which is needed for the NLCG algorithm and thus already computed), they fit an unique parabola. Hence, if the parabolic fit works, they get an even lower objective function with almost no additional cost (see Figure 3.2). Newman and Alumbaugh’s code is generally faster compared to Rodi and Mackie’s algorithm, because only one gradient and therefore two pseudo-forward problems have to be solved per NLCG iteration (cf. Appendix B.2). Rodi and Mackie’s code requires the calculation of an additional pseudo-forward problem, because the computation of the linear CG step

$$
\alpha_k = -\frac{\nabla \Phi_k^T p_k}{p_k^T H_k p_k},
$$

(3.43)

includes another matrix-vector product of the Jacobian. $H_k$ denotes the approximate Hessian matrix of the objective function and can be expressed as

$$
H = 2J^T C_d^{-1} J + 2\lambda L^T L.
$$

(3.44)

Note that the term $-2 \sum_{i=1}^{N} C_d^{-1} (d_{ij}^{\text{obs}} - F_i(m)) B_i(m)$, where $B_i(m)$ is the second derivative of $F_i(m)$ w.r.t. the model parameters, is neglected. If the trial step fails to satisfy the first Wolfe condition, a backtracking strategy is invoked. This can be done by taking the objective function at the trial step plus the objective function and the gradient at point zero to fit a parabola. If the minimum point of the parabola still does not satisfy the first Wolfe condition, the process can be repeated until sufficient decrease is reached.

My line search for NLCG in EMILIA can be adjusted by the user. One option is similar to the method of Newman and Alumbaugh (2000). However, the trial steps are defined differently. By using Equation
(3.39), we can derive the expression $\alpha_{\text{trial}} = \frac{\Delta m_{\text{max}}(p_k)}{m_k+1 - m_k}$. We can then evaluate $\alpha$ by defining the maximum change of the model parameters $\Delta m$. This is only a good approach for the first NLCG iteration, because the model parameters will change less for higher iterations. Therefore, the trial step in the remaining iterations is set to the best step obtained in the previous iteration. The trial step mode can be changed to the linear CG step according to Rodi and Mackie (2001) (either for the very first or for all the trial steps). If these two options do not provide good results, the user can constrain the line search with the second Wolfe condition to get more accurate minima. The line search procedure is illustrated in Algorithm 3.

Note that $\Phi(\alpha)$ in Algorithm 3 is an abbreviation for $\Phi(m_k + \alpha p_k)$ and that the subscript $k$ for the NLCG iteration is not indicated (except for the search direction and the previous step size $\alpha_{k-1}$), because the line search algorithm is performed only once per NLCG iteration. By using the first Wolfe condition only, we do not have to evaluate the gradient of the objective function. $\nabla \Phi$ is calculated first when the best step size is found. Thus, there are no additional pseudo-forward calculations needed for the line search, because we need the gradient at the best $\alpha$ in any case for the NLCG iteration. However, we need $\nabla \Phi$ at all the points to decide whether the second Wolfe condition is fulfilled. Thus, the algorithm is more expensive when the second Wolfe condition is used. For the first Wolfe condition, the parabolic fit is only accepted if it leads to a lower objective function. Otherwise, the initial trial step will be accepted. Following Nocedal and Wright (2006), the step size $\alpha_{\text{parabolic}}$ is expressed by

$$\alpha_{\text{parabolic}} = -\frac{\nabla \Phi(0)^T p_k \alpha_{\text{trial}}^2}{2(\Phi(\alpha_{\text{trial}}) - \Phi(0) - \nabla \Phi(0)^T p_k \alpha_{\text{trial}})}.$$  \hfill (3.45)

There is a special case for the second Wolfe condition. If the first Wolfe condition is satisfied, but the second is not and the gradient is still negative, $\alpha_{\text{trial}}$ will be increased by an user-defined factor and the process will be repeated. If $\alpha_{\text{trial}}$ was too large (i.e. the minimum lies between zero and $\alpha_{\text{trial}}$), backtracking is performed. In the backtracking/zooming procedure (Algorithm 4), index $l$ denotes the number of line search iterations. When only the first Wolfe condition is used, $\alpha_{\text{low}} = \alpha_{k,0} = 0$ and $\alpha_{\text{high}} = \alpha_{k,1} = \alpha_{\text{trial}}$. When the second Wolfe condition is used as well, $l$ is dependent on how many times the step size had to be increased in the line search algorithm. The subscript $k$ is indicated in the
Algorithm 3: Line Search

\textbf{if} \( k = 0 \) \textbf{then}

\[ \alpha_{\text{trial}} \leftarrow \begin{cases} \frac{\Delta m_{\text{max}}(p_k)}{\nabla \Phi(0) p_0} & \text{Option 1} \\ -\frac{\nabla \Phi(0)^T p_0}{p_0^T H_0 p_0} & \text{Option 2} \end{cases} \]  

\begin{equation} \tag{3.46a} \end{equation}

\textbf{else}

\[ \alpha_{\text{trial}} \leftarrow \begin{cases} \alpha_{k-1} & \text{Option 1} \\ \frac{-\nabla \Phi(0)^T p_k}{p_k^T H_k p_k} & \text{Option 2} \end{cases} \]  

\begin{equation} \tag{3.46b} \end{equation}

\textbf{end if}

\textbf{if} \textit{first Wolfe condition} is used \textbf{then}

Evaluate \( \Phi(\alpha_{\text{trial}}) \)

\textbf{if} \( \Phi(\alpha_{\text{trial}}) \leq \Phi(0) + c_1 \alpha_{\text{trial}} \nabla \Phi(0)^T p_k \) \textbf{then}

use \( \Phi(0) \), \( \nabla \Phi(0) \) and \( \Phi(\alpha_{\text{trial}}) \) to fit parabola

Evaluate \( \Phi(\alpha_{\text{parabolic}}) \)

\textbf{if} \( \Phi(\alpha_{\text{parabolic}}) < \Phi(\alpha_{\text{trial}}) \) \textbf{then}

accept \( \alpha_{\text{parabolic}} \), evaluate \( \nabla \Phi(\alpha_{\text{parabolic}}) \), return to NLCG algorithm

\textbf{else}

accept \( \alpha_{\text{trial}} \), evaluate \( \nabla \Phi(\alpha_{\text{trial}}) \), return to NLCG algorithm

\textbf{end if}

\textbf{else if} \( \Phi(\alpha_{\text{trial}}) > \Phi(0) + c_1 \alpha_{\text{trial}} \nabla \Phi(0)^T p_k \) \textbf{then}

backtracking

\textbf{end if}

\textbf{else if} \textit{both first and second Wolfe condition} are used \textbf{then}

Evaluate \( \Phi(\alpha_{\text{trial}}) \)

\textbf{if} \( \Phi(\alpha_{\text{trial}}) > \Phi(0) + c_1 \alpha_{\text{trial}} \nabla \Phi(0)^T p_k \) \textbf{then}

backtracking

\textbf{end if}

Evaluate \( \nabla \Phi(\alpha_{\text{trial}}) \)

\textbf{if} \( |\nabla \Phi(\alpha_{\text{trial}})^T p_k| \leq c_2 |\nabla \Phi(0)^T p_k| \) \textbf{then}

accept \( \alpha_{\text{trial}} \), return to NLCG algorithm

\textbf{else if} \( |\nabla \Phi(\alpha_{\text{trial}})^T p_k| > c_2 |\nabla \Phi(0)^T p_k| \) \textbf{and} \( \nabla \Phi(\alpha_{\text{trial}})^T p_k > 0 \) \textbf{then}

backtracking

\textbf{else if} \( |\nabla \Phi(\alpha_{\text{trial}})^T p_k| > c_2 |\nabla \Phi(0)^T p_k| \) \textbf{and} \( \nabla \Phi(\alpha_{\text{trial}})^T p_k < 0 \) \textbf{then}

increase \( \alpha_{\text{trial}} \) and repeat

\textbf{end if}

\textbf{end if}
Algorithm 4 Zooning

\[ \text{Algorithm 4} \]

set \( \alpha_{k,l} \leftarrow \alpha_{\text{high}} \) and \( \alpha_{k,l-1} \leftarrow \alpha_{\text{low}} \)
use \( \Phi(\alpha_{\text{low}}), \nabla \Phi(\alpha_{\text{low}}) \) and \( \Phi(\alpha_{\text{high}}) \) to fit parabola

if first Wolfe condition is used then
  Evaluate \( \Phi(\alpha_{\text{parabolic}}) \)
  if \( \Phi(\alpha_{\text{parabolic}}) \leq \Phi(\alpha_{\text{low}}) + c_1 \alpha_{\text{parabolic}} \nabla \Phi(\alpha_{\text{low}})^T p_k \) then
    accept \( \alpha_{\text{parabolic}} \), evaluate \( \nabla \Phi(\alpha_{\text{parabolic}}) \), return to NLCG algorithm
  else
    set \( \alpha_{\text{high}} \leftarrow \alpha_{\text{parabolic}} \), repeat backtracking
  end if
else if both first and second Wolfe condition are used then
  Evaluate \( \nabla \Phi(\alpha_{\text{parabolic}}) \)
  if \( \Phi(\alpha_{\text{parabolic}}) > \Phi(\alpha_{\text{low}}) + c_1 \alpha_{\text{parabolic}} \nabla \Phi(\alpha_{\text{low}})^T p_k \) and \( \nabla \Phi(\alpha_{\text{parabolic}})^T p_k > 0 \) then
    set \( \alpha_{\text{high}} \leftarrow \alpha_{\text{parabolic}} \), repeat zooming
  else if \( |\nabla \Phi(\alpha_{\text{parabolic}})^T p_k| < c_2 |\nabla \Phi(\alpha_{\text{low}})^T p_k| \) and \( \nabla \Phi(\alpha_{\text{parabolic}})^T p_k < 0 \) then
    set \( \alpha_{\text{low}} \leftarrow \alpha_{\text{parabolic}} \), repeat zooming †
  end if
end if

Algorithm 4 is called Zooming, because \( \alpha \) is increased (marked by †) when the minimum lies at a larger step size. This can only be the case for the second Wolfe condition. When using the first Wolfe condition, Algorithm 4 is always a backtracking algorithm. The parabolic fit is performed similarly to Equation (3.45). However, the lower step size does not have to be zero. Hence, the expression is modified to

\[
\alpha_{\text{parabolic}} = \frac{2 \alpha_{\text{low}} (\Phi(\alpha_{\text{low}}) - \Phi(\alpha_{\text{high}})) - (\alpha_{\text{low}} - \alpha_{\text{high}})(\alpha_{\text{low}} + \alpha_{\text{high}}) \nabla \Phi(\alpha_{\text{low}})^T p_k}{2(\Phi(\alpha_{\text{high}}) - \Phi(\alpha_{\text{low}}) + \nabla \Phi(\alpha_{\text{low}})^T p_k (\alpha_{\text{low}} - \alpha_{\text{high}}))}. \tag{3.47}
\]

This backtracking or zooming procedure is repeated until the desired decrease in the objective function or the maximum number of backtracking steps (defined by user) is reached. Algorithm 4 and 3 are then terminated and the found step size is used for the model update step in Algorithm 2.

3.2.4 Preconditioning

Preconditioning is used to accelerate convergence of the NLCG algorithm. In case of linear CG, preconditioning transforms the quadratic function into a more spherical function. The system of linear equations that will be solved can then be written as

\[
M^{-1}A x = M^{-1} b, \tag{3.48}
\]
where \( M \) is the preconditioner matrix. Pre-multiplying the system matrix \( A \) with \( M^{-1} \) lowers the condition number\(^2\) of \( A \) when the preconditioner is chosen appropriately. The inverse of \( A \) would be the perfect preconditioner, because \( A^{-1}A = I \) and \( \kappa(A^{-1}A) = 1 \). In this case, CG would solve the system in one iteration. However, this preconditioner is a bad choice, because evaluating \( M^{-1} = A^{-1} \) would require to solve the equation \( Ax = b \), which is the actual problem we want to solve. The task is therefore to find a preconditioner that approximates \( A \) and whose inversion is simple.

In the non-linear case, the aim is to find a preconditioner that approximates the Hessian of the objective function. The NLCG scheme (Algorithm 2) can be changed by applying the preconditioner to \( p_k \) to obtain more favourable search directions.

### Algorithm 5 Preconditioned NLCG

\[
k \leftarrow 0
\]

choose a start model \( m_0 \)

evaluate \( \nabla \Phi_0 \) (the gradient)

set \( u_0 \leftarrow -M^{-1}\nabla \Phi_0 \)

\[\text{while } \text{RMS} \geq \text{tol} \text{ and } k < k_{\text{max}} \text{ do} \]

perform line search to get \( \alpha_k \) (Chapter 3.2.3)

update model

\[
m_{k+1} \leftarrow m_k + \alpha_k u_k \tag{3.49a}
\]

evaluate \( \nabla \Phi_{k+1} \), then

\[
\beta_{PR}^{k+1} \leftarrow \max \left\{ \frac{\nabla \Phi_k^T M^{-1} (\nabla \Phi_{k+1} - \nabla \Phi_k)}{\nabla \Phi_k^T M^{-1} \nabla \Phi_k}, 0 \right\} \tag{3.49b}
\]

update search direction

\[
u_{k+1} \leftarrow -M^{-1}\nabla \Phi_{k+1} + \beta_{PR}^{k+1} u_k \tag{3.49c}
\]

\[
k \leftarrow k + 1 \tag{3.49d}
\]

end while

The vector \( u_k \) designates the preconditioned search direction. Theoretically, the best preconditioner is the approximate Hessian in Equation (3.44). However, this preconditioner involves the matrix product of the Jacobian with itself. In the NLCG algorithm, \( J \) is never obtained directly and hence, \( H \) cannot be evaluated. Furthermore, the matrix product is very expensive. The complete approximate Hessian is therefore a bad choice. A possibility to circumvent these troubles is to compute the complete Jacobian matrix every \( k \)-th iteration. This may affect the quality of the preconditioner, but is less expensive than calculating \( J \) every iteration. Additionally, to avoid the matrix product of the Jacobian, the diagonal of \( J^T C_d^{-1} J \) can be taken instead of the entire matrix. Furthermore, by taking only the diagonal of the regularisation term of the approximate Hessian, the entire preconditioner gets diagonal. This has the advantage that inversion of the preconditioner gets trivial and thus less expensive. This preconditioner, denoted as \( M_1 \), can be written as

\[
M_1 = \text{diag}(2J^T C_d^{-1} J + 2\lambda L^T L). \tag{3.50}
\]

---

\(^2\)As explained in Chapter 2.2.2, the condition number is defined as \( \kappa = \frac{\lambda_{\text{max}}(A)}{\lambda_{\text{min}}(A)} \), where \( \lambda_{\text{min}} \) and \( \lambda_{\text{max}} \) are the minimum and maximum eigenvalues of \( A \), respectively.
$M_1$ needs the computation of the Jacobian at least once at the beginning of the NLCG inversion. Newman and Alumbaugh (2000) suggested a preconditioner that does not need any additional computations. It is based on the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (for more details see Nocedal and Wright, 2006, Dennis and Schnabel, 1996). The preconditioner $M_2$ is updated in each iteration with

$$M_2^{k+1} = M_2^k + \frac{\nabla \Phi(m_k)\nabla \Phi(m_k)^T}{\nabla \Phi(m_k)^Tp_k} + \frac{\hat{v}_k \hat{v}_k^T}{\alpha_k \hat{v}_k^Tp_k},$$  

(3.51)

where again $\hat{v}_k = \nabla \Phi_{k+1} - \nabla \Phi_k$ and $M_2^0 = I$. Note that only the diagonal is updated to simplify the inversion of the preconditioner. This preconditioner is very time-efficient, because all the information required to update $M_2$ is already at hand from the NLCG scheme. The drawback here is that the approximation of the Hessian is quite poor in early iterations. The preconditioner gets better after each update. The tests performed in this thesis showed that $M_2$ does not always converge (cf. Chapter 4). Therefore, a third preconditioner $M_3$ was introduced that is similar to $M_2$ with the only difference that the start preconditioner $M_3^0$ was set to $M_1 = \text{diag}(2J^TC_d^{-1}J + 2\lambda L^TL)$. This has the advantage that the start preconditioner is closer to the approximate Hessian for early iterations. The drawback is that the full Jacobian has to be evaluated. $M_3$ is then updated as given in Equation (3.51). All three preconditioners described in this section are implemented in EMILIA and can be chosen by the user.
4 Results

Three field data sets in the RMT frequency range were inverted to test the applicability of the NLCG algorithm. The first data set was inverted in the quasi-static approximation both separately for the TE and TM mode and simultaneously in a joint inversion process (Section 4.1). The second data set was used to test the determinant mode in the quasi-static approximation (Section 4.2). The third data set was inverted with displacement currents in the determinant mode and is described in Section 4.3. Convergence plots for different input parameters for the line search and the search direction in the NLCG algorithm were tested and the results are shown in Section 4.4. To evaluate the quality of the results from NLCG, the same data were inverted with Occam’s inversion (OC) technique that was already implemented in EMILIA. For the joint inversion of TE and TM data and the DET data in the quasi-static approximation, additional inversions with damped Occam (DO) were performed. Resistivity models obtained by OC, DO and NLCG are shown and compared. Different preconditioners were tested for the NLCG inversion and the resulting convergence plots are shown. The penalty function in Equation (2.2) was used as the initial objective function for the NLCG inversion. I tested different values of $\lambda$ and chose the one that provided a sufficiently smooth model with reasonable RMS. The Occam inversion was done with the objective function in Equation (2.3) and $\lambda$ was determined automatically by the inversion procedure. For DO, the same $\lambda$ as for the NLCG inversion was applied. The line search to find the values of $\lambda$ (for OC) and $\beta$ (for DO) that minimise the data misfit was always started with $\lambda = \beta = 100$. Different values of $\lambda$ and $\beta$ were then tested with an interval of $\Delta \log_{10}(\lambda) = \Delta \log_{10}(\beta) = 0.2$. The regularisation term used for all inversion techniques was a first-order finite-difference operator. The data were weighted by variances that were used for the inversions of the original field studies. All the results obtained with NLCG in this chapter are produced with the default set-up given in Appendix C. The most important input parameters are listed in Table 4.1. If other input parameters were used, it is indicated in the text. The input parameters for Occam and damped Occam inversion are given in Table 4.2. The results were computed on an Intel®Core™ i7-3770S processor with up to eight CPU (Central Processing Unit) cores with clock rates of 3.1 GHz. Parallel computing was applied to compute the LU decomposition of the system matrix of the forward problem (which was then used to solve the pseudo-forward problems) and the Cholesky factorisation needed to solve the linear system of equations for the model update step in Occam and damped Occam inversion. Four CPU cores were used for the first and the third data set and two cores were used for the second data set. The NLCG algorithm was executed with serial processing. In my thesis, analysis is focused on the performance of the NLCG algorithm. I refer to the original papers of the field studies for geological interpretations.
Table 4.1: Input parameters for NLCG inversion.

<table>
<thead>
<tr>
<th>Input parameter</th>
<th>NLCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of iterations</td>
<td>100*</td>
</tr>
<tr>
<td>Desired RMS value</td>
<td>1.0*</td>
</tr>
<tr>
<td>Method to define trial step in 1st iteration</td>
<td>linear CG step</td>
</tr>
<tr>
<td>Method to define trial steps in subsequent iterations</td>
<td>best $\alpha$ from previous iteration</td>
</tr>
<tr>
<td>Wolfe condition to constrain line search</td>
<td>first</td>
</tr>
<tr>
<td>Coefficient $c_1$ to define sufficient decrease</td>
<td>$10^{-4}$</td>
</tr>
<tr>
<td>Maximum number of backtracking iterations</td>
<td>5</td>
</tr>
<tr>
<td>Parabolic fit if sufficient decrease with trial step</td>
<td>yes</td>
</tr>
<tr>
<td>Technique to update $\beta$</td>
<td>Polak-Ribiere</td>
</tr>
<tr>
<td>Fixed Lagrange multiplier</td>
<td>data set dependent</td>
</tr>
<tr>
<td>Preconditioner</td>
<td>indicated in the text</td>
</tr>
</tbody>
</table>

*For the DET data set including displacement currents (Section 4.3), the maximum number of iterations was 50 and a difference convergence criterion was used, i.e. the inversion was stopped when the difference of RMS between two subsequent iterations was $< 0.001$.

Table 4.2: Input parameters for Occam and damped Occam inversion.

<table>
<thead>
<tr>
<th>Input parameter</th>
<th>Occam</th>
<th>damped Occam</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of iterations</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Desired RMS value</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>Starting value of $\lambda$</td>
<td>2.0</td>
<td>—</td>
</tr>
<tr>
<td>Starting value of $\beta$</td>
<td>—</td>
<td>2.0</td>
</tr>
<tr>
<td>Step size for $\lambda$ search</td>
<td>0.2</td>
<td>—</td>
</tr>
<tr>
<td>Step size for $\beta$ search</td>
<td>—</td>
<td>0.2</td>
</tr>
<tr>
<td>Fixed $\lambda$</td>
<td>—</td>
<td>data set dependent</td>
</tr>
</tbody>
</table>

4.1 TE and TM data from Smørgrav, Norway

A joint investigation with direct-current resistivity (DCR), RMT and controlled-source audiomagneto-telluric (CSAMT) data was performed by Kalscheuer et al. (2013) to map post-glacial clay sediments that were destabilised by fresh water (so-called quick clays). The site is located in Smørgrav in southern Norway, to the west of Oslo. Here, I will show inversion results for RMT data in TE mode, TM mode and joint inversion of TE and TM data. Occam inversion uses the GN step, which has a very high convergence rate per iteration (cf. Chapter 2.2.1). For NLCG inversion, the convergence per iteration is slower, but on the other hand, the computations are less expensive. Therefore, I ran OC in a maximum of 10 iterations and the NLCG inversion in a maximum of 100 iterations. For all inversions in this section, the horizontal elements of the first-order difference operator were weighted by a factor of four compared to the elements that belong to the vertical derivatives to benefit layered structures. The desired RMS value was set to 1.0. Nine frequencies ranging from 14 to 225 kHz were measured. Displacement currents were not taken into account. As a start model for the inversions, a half-space of 300 $\Omega$m was chosen. The model has a width of 2746 m (156 cells) and a depth of 2018 m (31 cells). The stations located at the surface are centred horizontally w.r.t. the model and separated by 10 m each. Two responses, both phase and apparent resistivity, were measured at 30 stations, resulting in 540 data points for each mode.
Table 4.3: Specifications for data and model from Smørgrav.

<table>
<thead>
<tr>
<th>Data set</th>
<th>TE data</th>
<th>TM data</th>
<th>Joint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of stations</td>
<td>30</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>Number of responses</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Number of frequencies</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Number of modes</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Total number of data</td>
<td>540</td>
<td>540</td>
<td>1080</td>
</tr>
<tr>
<td>Data used for inversion</td>
<td>432</td>
<td>432</td>
<td>864</td>
</tr>
<tr>
<td>Total number of model blocks</td>
<td>4836</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of horizontal blocks</td>
<td>156</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of vertical blocks</td>
<td>31</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The topography of the site was not flat, but for the inversion the assumption was made that all stations have the same height. For this reason, stations that differed strongly in height compared to other stations were weighted down in the inversion process (Kalscheuer et al., 2013). Furthermore, some data were removed because they were very noisy. Specifications of the data sets and the model are summarised in Table 4.3.

Figure 4.1: Solution convergence for the TE data from Smørgrav of the objective function (a) and RMS (b) for three different Lagrange multipliers: \( \lambda \approx 3.16, \lambda = 10 \) and \( \lambda \approx 31.6 \). The NLCG inversion was performed with preconditioner \( \mathbf{M}_2 \). The dashed line in (b) designates the desired RMS of 1.0.

For the TE mode, three different Lagrange multipliers (\( \lambda = 10^{0.5} \approx 3.16, 10 \) and \( 10^{1.5} \approx 31.6 \)) were tested for the NLCG inversion. The desired RMS was not reached for any \( \lambda \). However, the test showed that for \( \lambda \approx 3.16 \) the RMS could be reduced to a value of 1.167 within 100 iterations. \( \lambda = 10 \) and \( \lambda \approx 31.6 \) yielded significantly higher RMS values (see Figure 4.1). For the TE data set, it turned out that a change of 0.001 in RMS between two subsequent iterations is a good criterion to abort the NLCG inversion. If the RMS is changing this slowly, it can be assumed that the inversion is close to a minimum. For \( \lambda \approx 3.16 \) this was the case after 35 iterations (RMS = 1.185). After this, the model parameters did not change significantly.

Figure 4.2a shows the resistivity model obtained after 35 NLCG iterations with preconditioner \( \mathbf{M}_2 \) for the TE data (see end of this section for discussion of preconditioning). In Figure 4.2b, the result
for the Occam inversion after seven iterations is illustrated. OC did not reach the desired RMS of 1.0 either. Here, a model is presented that has a similar RMS value compared to the NLCG result. It can be seen that the two results are almost identical. The conductive part between 100 and 160 m along the profile is extended to greater depth in the model obtained by OC. On the other hand, the resistive part at 170 to 190 m is reaching further down for the NLCG inversion. These two differences can be explained by the different objective functions that are minimised. For the NLCG inversion, the Lagrange multiplier for smoothness constraints is fixed over all iterations, whereas the Lagrange multiplier of the Occam inversion is adjusted in each iteration and hence, the minimum is not stationary. However, the resemblance of the two models show that the inversions converge to a similar minimum.

As for the TE mode, different values of $\lambda$ were tested (see Figure 4.3) for the NLCG inversion of the TM data. With $\lambda \approx 3.16$, convergence was good and an RMS lower than 1.0 was achieved after 43 iterations using preconditioner $M_1$ with updates every 20-th iteration. However, the model was not smooth enough and therefore not realistic. To obtain a sufficiently smooth model, it was necessary to set $\log_{10}(\lambda) \geq 1.5$. The results for other values of $\lambda$ can be found in Appendix D.2.

Figure 4.4 shows the results for the TM data inversion. Only four iterations were needed for Occam’s inversion to reach the desired RMS of 1.0. Here, the resistivity model after three iterations is shown (RMS = 1.185). The NLCG inversion was performed over 200 iterations, because the data misfit after
Figure 4.3: Solution convergence for the TM data from Smørgrav of the objective function (a) and RMS (b) for four different Lagrange multipliers: $\lambda \approx 3.16, \lambda = 10, \lambda \approx 31.6$ and $\lambda = 100$. The NLCG inversion was performed with preconditioner $M_1$, updated every 20-th iteration. Inversion with $\lambda \approx 3.16$ reached the desired RMS (dashed line in (b)) after 43 iterations.

100 iterations was still quite high (RMS = 1.825) due to the strong model regularisation. The additional 100 iterations in the NLCG inversion changed the model in deeper regions and at the edges, but did not have strong influence on the data misfit (RMS = 1.775). The resistivity model from Occam’s inversion shows some near-surface anomalies. These changes in resistivity are not reproduced by the NLCG inversion because small-scale features are smoothed out due to the strong model regularisation. The flexibility of Occam’s inversion pertaining to $\lambda$ leads to a model that has a low data misfit and is still reasonably smooth at the same time. Occam’s inversion uses $\lambda \approx 10^3$ in the first iteration, i.e. an even higher regularisation term than NLCG. In the subsequent iterations however, the optimal $\lambda$ has a value below 10 and hence, a lower RMS can be achieved. As mentioned above, starting with $\lambda \leq 10$ forces the NLCG inversion to converge to a very rough model (see Appendix D.2), which is not in accordance with the results from the TE data inversion.

For the joint inversion, the Lagrange multiplier was chosen to be 10, lying between the optimal value from TE data inversion ($\lambda \approx 3.16$) and TM data inversion ($\lambda \approx 31.6$). This time, $M_3$ was chosen as a preconditioner. The NLCG inversion resulted in an RMS of 1.367 after 100 iterations. In Figure 4.5, the resistivity models obtained by NLCG, Occam (RMS = 1.211) and damped Occam inversion (RMS = 1.350) are shown. The three models are almost identical with some small differences occurring between the model obtained by Occam’s inversion and the other two models. For instance, between 100 and 160 m below 15 m depth, the model blocks from Occam’s inversion are more conductive. The model from OC is generally rougher because values of $\lambda < 10$ were used to constrain the model in late iterations. The RMS obtained by OC could also be reached by the method of NLCG and DO when using $\lambda \approx 3.16$. However, this did not change the resulting models significantly compared to the models obtained with $\lambda = 10$.

The convergence plot for OC, DO and NLCG inversion is shown in Figure 4.6. The NLCG inversion has strong convergence in the first iterations and flattens out after around 100 seconds of run time (43 iterations). OC is converging almost to the desired RMS of 1.0. After iteration seven however, the RMS starts to increase and the desired RMS is not reached. The objective function of DO is decreasing continuously. After five iterations, the curve flattens out similar to the curve of NLCG and the RMS
Figure 4.4: Resistivity model obtained by NLCG inversion (a) after 200 iterations (RMS = 1.775) and resistivity model obtained by Occam inversion (b) after 3 iterations (RMS = 1.185) for the TM data from Smørgrav. The Lagrange multiplier for the NLCG inversion was set to $\sim 31.6$ and preconditioner $M_1$, updated every 20-th iteration, was applied. Black triangles mark the station locations.

The absolute computation times are much higher for OC and DO than for the method of NLCG. More than 800 seconds are needed to reach the minimum RMS for Occam’s inversion, whereas NLCG terminates its last iteration at 250 seconds. DO needs over 850 seconds to reach a reasonable RMS. In each iteration of OC and DO, the model is updated several times to test different values of $\lambda$ and $\beta$, respectively. Full evaluations of the Jacobian are involved, i.e. 24 pseudo-forward problems\(^1\) have to be solved per frequency, mode and iteration (because the number of stations whose data were used for inversion is 24). Furthermore, for each model update, the forward data vector has to be computed and a linear system of equations (cf. Equation (2.6) and (2.8)) has to be solved. In EMILIA, this is done by Cholesky factorisation. This explains the high computation times for Occam’s inversion of 819 seconds to complete seven iterations (minimal RMS) and 647 seconds for five iterations (resistivity model in Figure 4.5). DO requires 898 seconds to reach an RMS < 1.40. NLCG is thus roughly three to four times faster than OC and DO. The algorithm implemented in EMILIA only needs to solve two pseudo-forward problems per frequency to evaluate the gradient in each iteration (see Appendix B.2) and two forward problems to determine the objective function at the trial step and at

\(^1\)The LU decomposition of the system matrix of the forward problem has to be computed only once per frequency, mode and iteration. Subsequently, 24 forward and backward substitutions have to be performed to solve the pseudo-forward problems.
Figure 4.5: Resistivity model obtained by NLCG inversion (a) with preconditioner $M_3$ after 100 iterations (RMS = 1.367), by Occam inversion (b) after 5 iterations (RMS = 1.211) and by damped Occam inversion (c) after 7 iterations (RMS = 1.350) for the joint inversion of TE and TM data from Smørgrav. The Lagrange multipliers for NLCG and DO were set to 10. Black triangles mark the station locations.
Figure 4.6: Comparison of convergence between Occam (10 iterations), damped Occam (10 iterations) and NLCG inversion (100 iterations) for the joint inversion of TE and TM data from Smørgrav. For all three inversion techniques, the desired RMS = 1.0 was not reached. The NLCG and DO inversion were regularised with $\lambda = 10$ and preconditioner $M_3$ was used for NLCG.

the second step obtained by parabolic interpolation. One additional pseudo-forward problem is solved in the very first iteration to compute the linear CG step. As described above, after 100 seconds the convergence gets very slow for NLCG and thus, the inversion could already be stopped at this point. Nevertheless, tests have shown that later iterations can improve the resistivity model in deeper regions where the stations are farther away. This does not improve the RMS considerably, but is still visible in the model (see Appendix D for these figures).

Table 4.4: Comparison of objective function and RMS obtained after 100 iterations for different preconditioners for the data from Smørgrav.

<table>
<thead>
<tr>
<th>Preconditioner</th>
<th>non</th>
<th>$M_1$</th>
<th>$M_2$</th>
<th>$M_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>TE Time (seconds)</td>
<td>183.59</td>
<td>203.75</td>
<td>178.19</td>
<td>180.30</td>
</tr>
<tr>
<td>TE Objective function</td>
<td>1441.8</td>
<td>1379.4</td>
<td>1218.4</td>
<td>1355.7</td>
</tr>
<tr>
<td>TE RMS</td>
<td>1.225</td>
<td>1.221</td>
<td>1.167</td>
<td>1.224</td>
</tr>
<tr>
<td>TM Time (seconds)</td>
<td>49.65</td>
<td>53.06</td>
<td>51.40</td>
<td>48.47</td>
</tr>
<tr>
<td>TM Objective function</td>
<td>9280.5</td>
<td>8544.3</td>
<td>8411.0</td>
<td>8399.8</td>
</tr>
<tr>
<td>TM RMS</td>
<td>2.795</td>
<td>2.721</td>
<td>2.723</td>
<td>2.702</td>
</tr>
<tr>
<td>Joint Time (seconds)</td>
<td>232.80</td>
<td>241.00</td>
<td>228.34</td>
<td>229.52</td>
</tr>
<tr>
<td>Joint Objective function</td>
<td>4451.1</td>
<td>4126.7</td>
<td>3871.7</td>
<td>4060.9</td>
</tr>
<tr>
<td>Joint RMS</td>
<td>1.411</td>
<td>1.368</td>
<td>1.351</td>
<td>1.367</td>
</tr>
</tbody>
</table>

Three different preconditioners were used for the data sets from Smørgrav. Resistivity models presented in this section were produced with preconditioner $M_2$ for the TE data inversion, $M_1$ (updated every 20-th iteration) for the TM data inversion and $M_3$ for the joint inversion of both TE and TM data. To compare the performance of the preconditioners, I inverted every data set with all three preconditioners
and without preconditioning. The obtained RMS and objective function values after 100 iterations are summarised in Table 4.4. Preconditioning did significantly reduce the values of the objective function. In some cases, it also lowered the RMS values. However, preconditioning is applied on the gradient of the entire objective function (cf. Chapter 3.2.4) and not just on the data misfit. Thus, the reduction of the RMS can be a side effect of preconditioning, but is not a necessity. Tests have also shown that preconditioning can indeed accelerate convergence of the NLCG inversion, but the minimum that the algorithm converges to is not affected. Hence, the resistivity models that were produced with different preconditioners were very similar, differing only slightly in the smoothness or the data misfit due to different values of the objective function. For the TE data, preconditioner $M_2$ provided the best results pertaining to the reduction of the objective function, followed by $M_3$ and $M_1$. In the inversions with preconditioner $M_1$, evaluations of the full Jacobian were done every 20-th iteration. Tests showed that this was a reasonable choice, because updating the preconditioner more frequently did not improve convergence (per iteration), but led to significantly higher computation times. On the other hand, fewer updates did deteriorate convergence so that the lower computation times per iteration could not compensate for that. For 100 iterations, preconditioner $M_1$ required 20 to 25 seconds more than inversion with the other two preconditioners and without preconditioning. Nevertheless, it could only slightly reduce the objective function. $M_1$ showed also slower convergence compared to the other preconditioners for the TM data and the joint inversion. Preconditioner $M_2$ lowered the objective function the most for the TE data inversion. For the TM data however, the NLCG inversion with $M_2$ converged poorly in the first few iterations. A higher objective function was found after two iterations, meaning that the search direction was no longer a descent direction and the inversion was stopped. I then repeated the inversion without stopping it for higher values of the objective function and the algorithm started to converge again after the second iteration. The problems that occurred with $M_2$ are discussed in detail in Chapter 5. The same observation was made for the joint inversion, where the objective function of preconditioner $M_2$ increased from iteration six to seven. After this, the inversion converged again strongly and achieved the lowest objective function after 100 iterations. Preconditioner $M_3$ converged always faster than $M_1$, but was slower than $M_2$ for the TE data and the joint inversion. However, the convergence with $M_3$ was stable for all data sets. Due to the results obtained by the data from Smørgrav pertaining to preconditioning, the remaining two data sets (Section 4.2 and 4.3) were inverted with preconditioners $M_2$ and $M_3$ only.

4.2 Determinant data from Skediga, Sweden

RMT and controlled-source magnetotelluric measurements were performed to map water-bearing formations below thick clay lenses in Skediga near Uppsala, Sweden (Pedersen et al., 2005). Within the scope of this thesis, I will consider inversion of RMT data only. 12 frequencies in a range from 4 to 180 kHz were used, but displacement currents were not taken into account. As for the data from Smørgrav, the inversions were run over a maximum of 10 iterations for OC and a maximum of 100 iterations for NLCG. Additionally, inversion with damped Occam was done for 10 iterations. For the results of NLCG, preconditioner $M_3$ was used. The components of the first-difference operator in the objective function were weighted equally. The start model was chosen to be a half-space with a resistivity of 1000 $\Omega m$. The model dimensions were 2940 m in width (94 model blocks) and 1555 m in depth (30 model blocks). The 22 stations were located at the surface, horizontally centred w.r.t. the model and separated by 10 m each. Apparent resistivities and phases were measured at each station, resulting in totally 528 data
points. The data were inverted in the determinant mode (cf. Chapter 3.1.1). All the data were included for the inversion. Data and model specifications are listed in Table 4.5.

**Table 4.5**: Specifications for data and model from Skediga.

<table>
<thead>
<tr>
<th>Data set</th>
<th>DET data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of stations</td>
<td>22</td>
</tr>
<tr>
<td>Number of responses</td>
<td>2</td>
</tr>
<tr>
<td>Number of frequencies</td>
<td>12</td>
</tr>
<tr>
<td>Number of modes</td>
<td>1</td>
</tr>
<tr>
<td>Total number of data</td>
<td>528</td>
</tr>
<tr>
<td>Data used for inversion</td>
<td>528</td>
</tr>
<tr>
<td>Number of model blocks</td>
<td>2820</td>
</tr>
<tr>
<td>Number of horizontal blocks</td>
<td>94</td>
</tr>
<tr>
<td>Number of vertical blocks</td>
<td>30</td>
</tr>
</tbody>
</table>

In Figure 4.7, resistivity models from NLCG, Occam and damped Occam inversion are shown. It can be seen that the models look similar underneath the stations. The largest differences occur on the left and right edges of the profile. Starting with the right edge, the models from DO and NLCG differ quite strongly compared to OC. The resistive structure (100 to 300 $\Omega \text{m}$) in the models obtained by NLCG and DO comes up to shallow depth, whereas for OC the entire right edge is rather conductive ($< 30 \Omega \text{m}$). It would be necessary to extend the profile in order to get more detailed information about the subsurface structure at the right edge. Differences in the models can also be observed on the left-hand side of the profiles. Occam inversion shows a band of cells with resistivities around 100 $\Omega \text{m}$, with a conductor underneath with resistivities of 10 to 30 $\Omega \text{m}$. The models obtained by DO and NLCG are resistive all the way down to 60 m. DO and NLCG produce generally smoother models. This is due to the higher Lagrange multipliers that were used for DO and NLCG ($\lambda \approx 31.6$). Especially in the last five iterations, OC used significantly lower values of $\lambda$ ($5 \leq \lambda \leq 15$). In the center of the profile, i.e. underneath the stations, the models look very similar. All three models exhibit a conductive region of 1 to 10 $\Omega \text{m}$ centred at 210 m along the profile and 15 m depth. Furthermore, the resistive cells $\rho > 100 \Omega \text{m}$ between 110 and 120 m along the profile coming up to the surface are present in all three models. Therefore, the parts of the model where the data are sensitive to are similar for all three inversion techniques.

The computation times of NLCG, Occam and damped Occam inversion are plotted in Figure 4.8. Although NLCG is clearly the fastest method due to the reasons mentioned in Chapter 4.1, OC and DO are not as far behind as for the results from Smørgrav. DO has similar convergence to OC, but its RMS is decreasing continuously until iteration 10, whereas the objective function of OC starts to oscillate after iteration four. The surprisingly good performances of the OC and DO are due to the relatively small number of data ($N = 528$) and model parameters ($M = 2820$). The Cholesky factorisation and the matrix product $J^T J$ can therefore be calculated much faster than in the Smørgrav example. For OC, the number of test model updates is reduced to four or five per iteration for the last seven iterations, which reduces the computation time additionally. Damped Occam inversion has a fixed Lagrange multiplier and searches only for the best damping factor $\beta$ to constrain the model update step (cf. Chapter 2.2.1). Thus, it generally requires less update steps per iteration than Occam inversion, especially in late iterations. On the other hand, the convergence is lower per iteration than for OC, because the step direction is
Figure 4.7: Resistivity model obtained by NLCG inversion after 100 iterations (RMS = 1.335) (a), resistivity model obtained by Occam inversion after 9 iterations (RMS = 1.488) (b) and resistivity model obtained by damped Occam inversion after 10 iterations (RMS = 1.400) (c) for the DET data from Skediga. The Lagrange multiplier for the NLCG inversion was set to $\sim 31.6$ and preconditioner $M_3$ was applied. The fixed $\lambda$ in damped Occam inversion was set to $\sim 31.6$, as well. Black triangles mark the station locations.
somewhere between the Gauss-Newton and steepest descent direction.

Preconditioning improved convergence for the method of NLCG. The objective function could be reduced from 2476.7 (without preconditioning) to 2280.4 with $M_3$ and 2260.5 with $M_2$ after 100 iterations (see Figure 4.9). Even though the data misfit is not directly affected by preconditioning because the preconditioners are applied on the entire objective function, the RMS could be reduced from 1.398 to 1.335 with $M_3$ and to 1.319 with $M_2$. The characteristics of preconditioners $M_2$ and $M_3$ become apparent by having a closer look at the convergence plot. It can be observed that the convergence is better without preconditioning during the first few iterations until 25 seconds of run time. After that, $M_3$ overtakes the non-preconditioned inversion and has a lower objective function for the remaining iterations. $M_2$ overtakes the non-preconditioned inversion after about 95 seconds. The convergence plot shows that preconditioner $M_2$ has some troubles to converge at early iterations. From iteration one to two\(^2\), the objective function even increased. In these cases, it is often not reasonable to take the step from the previous iteration as the trial step for the next iteration, because backtracking was performed without any success (the objective function still increased after five backtracking steps). Repeated backtracking leads to very small step sizes that are not a good measure for the next iterations. Thus, I performed the inversion with $M_2$ by taking the linear CG step as the trial step for all iterations. This involves one additional pseudo-forward problem per iteration. Therefore, the total run time with preconditioner $M_2$ is considerably higher than for the other two cases. The BFGS update technique that is used for both $M_2$ and $M_3$ implies that the preconditioners get better with increasing iteration numbers, because the Hessian approximation is improved in each iteration. Thus, it takes some time until the preconditioners get good enough to accelerate convergence. The overall run time for $M_3$ is even a bit faster compared to the inversion without preconditioning. This implies that less backtracking steps had to be performed.

\(^2\)Note that the first mark in Figure 4.9 designates the objective function obtained for the start model $m_0$. 

37
Figure 4.9: Solution convergence of objective function for NLCG inversion without preconditioner and with preconditioners $M_2$ and $M_3$ for the DET data from Skediga. Preconditioner $M_2$ was applied together with a different line search procedure. Instead of taking the best step from the previous iteration as the trial step, the linear CG step was taken for every iteration as the trial step. $\lambda$ was set to $\sim 31.6$. The enhanced symbols mark iteration 100.

Hence, no additional cost is generated.

4.3 Determinant data from Āvrö, Sweden

Fractured granite was investigated by Linde and Pedersen (2004) with RMT in the frequency range of 14 to 226 kHz. The site is located on the small island Āvrö, off the south-east coast of Sweden. Linde and Pedersen (2004) did not include frequencies over 56 kHz in their inversion in order to avoid influence by displacement currents, which they did not take into account in their forward modelling. Kalscheuer et al. (2008) developed a code in which displacement currents could be included. They obtained results that were in better agreement with reflectors from seismic studies (Juhlin and Palm, 1999) and borehole information. I inverted the data set with both OC and NLCG over the entire frequency range including displacement currents. The data was measured at 96 stations separated by 10 m. Both apparent resistivity and phase were measured over 9 frequencies in the determinant mode. This results in a total of 1728 data points. Some of the data were excluded due to low measurement accuracy (Kalscheuer et al., 2008). The model was chosen to be discretised rather finely with 150 cells horizontally and 74 cells vertically with a total width of 72.7 km and a depth of 38.4 km. In total, 1710 data points were inverted for 11100 model blocks, making this inverse problem significantly larger than the previous ones. Thus, the computation times are much higher. I therefore ran the NLCG inversion for only 50 iterations or until difference convergence was reached, meaning that the difference of the RMS values in two subsequent iterations is lower than 0.001. The Occam inversion was performed over 10 iterations as for the other data sets. The start model of OC was a 10000 $\Omega$ m half-space. This start model did direct the NLCG inversion into a local minimum where the data fit was very poor (RMS $\approx 7.5$). By changing the start model to 1000 $\Omega$ m, the NLCG inversion converged well and reached an even lower RMS than Occam’s inversion. Because
displacement currents were included, a constant relative dielectric permittivity \( \varepsilon_r = 5 \) was chosen for all the model blocks. Specifications of the data and the model are given in Table 4.6.

<table>
<thead>
<tr>
<th>Data set</th>
<th>DET data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of stations</td>
<td>96</td>
</tr>
<tr>
<td>Number of responses</td>
<td>2</td>
</tr>
<tr>
<td>Number of frequencies</td>
<td>9</td>
</tr>
<tr>
<td>Number of modes</td>
<td>1</td>
</tr>
<tr>
<td>Total number of data</td>
<td>1728</td>
</tr>
<tr>
<td>Data used for inversion</td>
<td>1710</td>
</tr>
<tr>
<td>Number of model blocks</td>
<td>11100</td>
</tr>
<tr>
<td>Number of horizontal blocks</td>
<td>150</td>
</tr>
<tr>
<td>Number of vertical blocks</td>
<td>74</td>
</tr>
</tbody>
</table>

The results from Occam and NLCG inversion are given in Figure 4.10. After some preliminary tests, the optimal Lagrange multiplier for the NLCG inversion was chosen to be 100. It can be seen that the overall shape of the resistive body in the center of the model is similar for both inversion techniques. Furthermore, both models correlate nicely with the seismic reflectors C and D detected by Juhlin and Palm (1999). The main differences lie in the details. The resistive body is reaching deeper for Occam’s inversion. In addition, the center of the resistor shows resistivities up to 100000 \( \Omega m \), whereas the model obtained by NLCG inversion has maximum resistivities in the range of 30000 to 60000 \( \Omega m \). At the left and right edge below 150 m depth, the model obtained by OC exhibits somewhat more conductive structures (\( \rho < 600 \Omega m \)). The method of NLCG produces a model where the resistivities outside the centrally located resistor do not change more than one order of magnitude (600 \( \leq \rho \leq 6000 \Omega m \)). Due to the fixed \( \lambda = 100 \) for the NLCG inversion, the model is generally smoother than Occam’s inversion model, because the latter technique used smaller values of \( \lambda \) for late iterations.

Despite the high model regularisation, NLCG inversion manages to attain a considerably lower data misfit than Occam’s inversion (RMS of 2.208 compared to 2.483, respectively), which can be seen in Figure 4.11. For this data set, OC needed 11754 seconds (3.27 hours) to complete 10 iterations, whereas NLCG inversion required 682 seconds (11.37 minutes) to run 20 iterations until the difference convergence criterion was fulfilled. The NLCG inversion is thus roughly 17 times faster than Occam, without loosing quality in the final resistivity model.

Preconditioning could again increase the convergence rate substantially (see Figure 4.12). Preconditioner \( M_3 \) is slow in the beginning, but overtakes the inversion without preconditioning after 600 seconds of run time and finishes at a significantly lower objective function. The convergence curve of preconditioner \( M_2 \) is steep from the start and reaches a value of the objective function after only 20 iterations that is considerably lower than the values obtained by \( M_3 \) or without preconditioning after 51 and 47 iterations, respectively. As mentioned earlier, a relative abortion criterion was used for these data and thus, the total number of iterations differ.
Figure 4.10: Resistivity model obtained by NLCG inversion (a) with preconditioner $M_2$ after 20 iterations (RMS = 2.208) and resistivity model obtained by Occam inversion (b) after 8 iterations (RMS = 2.483) for the DET data from Avrø with displacement currents. The Lagrange multiplier for the NLCG inversion was set to 100. The oblique lines denoted by C and D indicate seismic reflectors from Juhlin and Palm (1999). Black triangles mark the station locations.
Figure 4.11: Comparison of convergence between Occam (10 iterations) and NLCG inversion (20 iterations) for the DET data from Åvrø with displacement currents. For both inversion techniques, the desired RMS = 1.0 was not reached. The NLCG inversion was regularised with $\lambda = 100$ and preconditioner $M_2$ was used.

4.4 Convergence analysis of NLCG input parameters

In addition to the preconditioning tests, I performed inversion runs with different values of $\beta$ (i.e. different search direction update techniques, see Chapter 3.2.2) and different trial steps. Furthermore, I tested whether it is necessary to constrain the line search with both the first and the second Wolfe condition or if it is adequate to use the first Wolfe condition only. The tests were done with the TE and TM data sets from Smørgrav and the determinant data from Skediga. Figure 4.13 shows the convergence of both the objective function and RMS for the TE data from Smørgrav for different search direction update techniques, namely the method of Polak-Ribiere (PR), Fletcher-Reeves (FR), Hestenes-Stiefel (HS), Dai-Yuan (DY) and Hager-Zhang (HZ). It can be seen that PR, HS and HZ have the highest convergence rates in the first 40 seconds. After 100 iterations however, the other two methods FR and DY reach almost the same minima. If one wants to run the NLCG inversion for fewer iterations, the three former techniques are clearly favourable. Therefore, I chose PR as default in EMILIA. Similar results were found for the TM data, whereas the DET data inversion required similar computation times for all techniques (see Appendix D.6).

As described in Chapter 3.2.3, the trial steps can be defined in different ways. Three cases are considered here:
1. Set the trial step in the first NLCG iteration equal to the linear CG step. For subsequent iterations, take the best step from the previous iteration as the trial step.

2. Set the trial steps in all iterations equal to the linear CG step.

3. Define the trial step in the first NLCG iteration such that the model parameters do not change more than $\Delta m$. For subsequent iterations, take the best step from the previous iteration as the trial step.

Note that for the third case, $\Delta m = 1.0$ corresponds to a maximum allowed change of a factor of 10 for the model parameters, because $m$ is logarithmic. For the second case, tests showed that it is often not required to compute a parabolic fit when the first Wolfe condition is reached, because the linear CG trial step is already sufficiently close to the minimum. Therefore, it is possible to circumvent the calculation of an additional forward problem by starting the next NLCG iteration directly without computing the parabolic fit. In Figure 4.14, convergence of case one to three are compared. Case two is done with and without parabolic fit. Case three was tested for $\Delta m = 1.0$ and 2.0. The slowest convergence is achieved by case two including parabolic fit. It can be seen that a lot of time is saved by skipping the parabolic fit. This leads to the same final RMS, but is clearly faster (here more than 50 seconds). Defining the first trial step through the maximum allowed change of the model parameters only makes sense when the user has some idea on how much the model parameters should vary w.r.t. the start model. Option 3 was always faster than option 2 with parabolic fit, but never faster than option 1. A save option is therefore to take the trial step as a linear CG step in the first iteration (case 1). This is almost as fast as case 2 for the TE data and even faster for the TM data (see Appendix D.6). Convergence can get very slow for option 1 when the search direction is not a descent direction (as seen in some cases for preconditioner $M_2$). Then, backtracking is performed until the maximum of backtracking iterations is reached and a very small step size is obtained. This step size is then used as a trial step for the next iteration, which is in most cases far too small. On the other hand, option 2 has the disadvantage that the minimum may be very inexact if
the linear CG is a bad guess but still satisfies the *first Wolfe condition*. In those cases, a parabolic fit can reduce the objective function significantly. For the TE data, this was however not the case. For the DET data, similar results compared to the TE data inversion were obtained (see Appendix D.6). Thus, there is no clear evidence which option is best. However, option 1 and 2 are favourable.

The third test was performed to decide whether a more exact line search could improve convergence. In this context, inversions were performed by constraining the line search to the *first Wolfe condition* only or to both the *first* and the *second Wolfe condition*. The results in Figure 4.15 were obtained from the inversion of the determinant data from Skediga. Although using both *Wolfe conditions* increases convergence per iteration, it does not improve convergence per time, because for the *second Wolfe condi-*
additional evaluations of the gradient and thus additional pseudo-forward problems are required. It has to be mentioned that no parabolic fit was performed when the second Wolfe condition was used and sufficient decrease was reached with the trial step. Hence, for some iterations, the algorithm constrained to both Wolfe conditions was even faster than the algorithm using the first Wolfe condition only. This was for instance observed for the TE data set, where the algorithm with the more accurate line search completed 100 iterations in a slightly shorter time (see Appendix D.6). For the DET data shown here, the second Wolfe condition did not improve convergence. The algorithm using the first Wolfe condition was 45 seconds faster and achieved almost the same RMS. For the TM data, the algorithm with a more exact line search did reach the desired RMS of 1.0 with 25 iterations less. However, this was only 2 seconds faster (see Appendix D.6). Thus, it is generally sufficient to perform a rather inexact line search constrained by the first Wolfe condition only. For all data sets inverted in this project, the first Wolfe condition was sufficient to obtain good convergence.

Figure 4.15: Solution convergence for the DET data from Skediga of the objective function (a) and RMS (b) for the line search using the first Wolfe condition only and using both the first and the second Wolfe condition. The NLCG inversion was performed without preconditioner. $\lambda$ was set to $\sim 31.6$. 

![Graphs showing solution convergence](image-url)
5 Discussion

Three field data sets in the RMT frequency range were inverted by applying the existing inversion techniques of Occam and damped Occam and testing the newly implemented method of Non-Linear Conjugate Gradients. The data were inverted in three different modes, namely TE, TM and determinant mode. The performance of the NLCG algorithm was compared to the other two techniques considering computation times and quality of the output models. Three preconditioners and different line search schemes were implemented in the NLCG code and tested against each other to find the best possible set-up for the algorithm. The discussion is divided in three parts: First, the obtained resistivity models are analysed in detail and advantages and limitations of NLCG compared to Gauss-Newton based inversion methods, such as OC and DO, are described. Secondly, preconditioning for the NLCG inversion is discussed and thirdly, the results of the line search and search direction tests from Chapter 4.4 are interpreted.

For all the data sets, the resulting resistivity models of NLCG, OC and DO were very similar. There was no evidence that one of the methods would produce models with consistently higher or lower data misfits. For the data from Smørgrav, Occam’s inversion achieved the lowest RMS values. For the DET data from Skediga however, NLCG outperformed both DO and OC and reduced the data misfit the most. Looking at the actual resistivity structure of the models, it can be seen that the key factor affecting the final result is not the inversion method itself, but rather the objective function that is minimised. The results showed that the largest differences occurred between the models of OC and the other two methods (DO and NLCG). This can be explained by the fixed Lagrange multiplier used for both DO and NLCG, which is controlling the smoothness of the models. Occam’s inversion however has a non-stationary objective function, allowing the inversion process to be more flexible pertaining to the model regularisation. A main advantage of OC is that the inversion is guided superiorly to the minimum of the objective function. This was for instance the case for the TM data, where Occam’s inversion technique produced a smooth model, while the method of NLCG yielded a rough and unrealistic model when choosing a too small $\lambda$. For the TM data inversion, OC used high values of $\lambda$ in the first few iterations and smaller values of $\lambda$ in the last iterations to achieve a sufficiently smooth model with low data misfit. By applying a high regularisation term, the objective function gets more quadratic with respect to the model parameters. Thus, the non-linearity of the problem is damped and the risk that the inversion converges to a local minimum, where the desired RMS cannot be achieved, is small. Hence, OC is less susceptible to land in local minima in early iterations. NLCG and DO run the risk of being trapped in a local minimum when the objective function is not properly regularised. Thus, the resulting model will either be too rough (local minimum, small $\lambda$) or the data misfit too high (large $\lambda$). The DET data inversion from Ävrö also showed that OC is less dependent on the choice of the start model. While OC converged for both start models that were applied (10000 and 1000 $\Omega m$ half-spaces), NLCG only reached a reasonable RMS with the latter start model. This can again be explained with a local minimum, which NLCG converged to for the 10000 $\Omega m$ start model. The big advantage of NLCG is certainly the significantly lower computation times that are needed for an inversion run. NLCG needs to solve two pseudo-forward problems to obtain the gradient and two forward problems per iteration to evaluate the objective function. For the gradient, only column-wise information from the Jacobian matrix is needed. The element-wise computation of the sensitivity matrix is however required by OC and DO, once per iteration. Furthermore, matrix products of the Jacobian are involved to derive the linearised system of
equations for the model updates. This linear system is then solved with Cholesky factorisation for each model update. The larger the size of the inversion problem gets, the more expensive are these operations. Therefore, it is mainly a question of the problem size (i.e. number of data and model blocks) how much faster NLCG inverts the data. The convergence plots showed that for smaller problems with \( N < 1000 \) and \( M < 5000 \), NLCG was roughly two to four times faster. Of course, this also dependent on the number of model updates that have to be calculated for OC and DO and not just on the problem size. This in turn is dependent on the data set and on the input parameters. Nevertheless, with increasing problem size, the advantages of NLCG get more emphasised. Therefore, NLCG is for instance useful to invert large 2D RMT data sets where displacement currents are accounted for, because in this case, the mesh has to be discretised finely and the number of model parameters can get immense. Furthermore, in three dimensions \( M \) gets already large for smaller problems and hence, NLCG is beneficial for three-dimensional inversions. The tests of this thesis were focused on the convergence of the NLCG algorithm per computation time. The memory requirements of the method of NLCG, which are a limiting factor for large inverse problems, as well, were not evaluated and compared to DO and OC. Since the NLCG algorithm implemented in this thesis only uses matrix-vector products of the Jacobian and the applied preconditioners all were diagonal, the storage of large-sized matrices is avoided. Thus, it can be expected that the inversion methods of Occam and damped Occam require more memory than the method of NLCG. Further investigations could confirm this hypothesis.

Different studies have shown that preconditioning can significantly accelerate convergence of the NLCG inversion. Rodi and Mackie (2001) applied the approximated Hessian as a preconditioner. They used the Jacobian of a half-space to avoid expensive computations and held their preconditioner constant over the entire inversion procedure. In this thesis, a similar approach was chosen for preconditioner \( M_1 \), where the Jacobian was computed explicitly for the start model. However, only the diagonal of the Hessian approximation was used. The preconditioner was updated every \( k \)-th iteration to obtain better approximations. The best results were obtained with \( k = 20 \). Newman and Alumbaugh (2000) suggested another technique to compute the preconditioner. They used the BFGS algorithm to estimate the diagonal of the approximated Hessian. The identity matrix was chosen as a start preconditioner, which was then updated every iteration. The same preconditioner was used in this thesis and was denoted by \( M_2 \). Additionally, a combination of \( M_1 \) and \( M_2 \), denoted as \( M_3 \), was applied. This preconditioner used the approximated Hessian as a start preconditioner and was then updated by the BFGS algorithm. All three preconditioners were tested extensively in this thesis. The convergence analysis showed that preconditioner \( M_1 \) provided the poorest results. There, a trade-off has to be found between the number of updates and the expensiveness of the computations. The best preconditioner would be obtained by updating it every iteration. However, this would strongly increase computation times. On the other hand, calculating the approximate Hessian once in the beginning of the inversion would be very time efficient, but the preconditioner would get outdated rather quickly. This trade-off cannot be overcome. Therefore, the BFGS algorithm which only needs information that is already available from the NLCG algorithm (e.g. the gradient) and updates the preconditioner every iteration is a better choice. For three of the five tested data sets, \( M_2 \) converged the fastest. For the other two data sets, preconditioner \( M_3 \) won the race. For the largest data set (DET data from Ävrö), \( M_2 \) converged already after 20 iterations and was more than two times faster than \( M_3 \) that needed all 50 iterations. Furthermore, \( M_2 \) achieved a considerably lower RMS. Hence, if \( M_2 \) is working properly, it is clearly the best preconditioner. In the
case of $M_2$, the start preconditioner does not have any information about the curvature of the objective function. This can be advantageous, because too much information about the curvature at the start point can be misleading, since the overall shape of the objective function might differ strongly. For $M_3$, the detailed information of the approximate Hessian at the start point might not be a good representation of the Hessian of the entire objective function. Thus, this approach can require some time until the preconditioner starts to be effective, while $M_2$ converges strongly from the beginning, as seen for the data set from Åvrog. On the other hand, $M_2$ does not always converge. Especially in early iterations, observed both for the TM, joint and DET (Skediga) data inversion, the objective function started to increase for one iteration, meaning that the search direction was not a descent direction. In these cases, more detailed information about the local curvature would have improved the preconditioner, because for $M_3$, convergence was always stable. It is therefore strongly dependent on the data set whether $M_2$ or $M_3$ is the better choice. Interestingly, preconditioner $M_2$ did not show these convergence problems for the pure TE data set. The TE mode, where the electric field component is aligned with the strike direction, is not as sensitive to resistive structures as the TM mode. When resistive materials are present at shallow depth, the identity matrix as a start preconditioner seems to be a bad choice for the TM data inversion, since the sensitivity matrix entries for model cells at shallow depth can get large. The BFGS update is then too slow to compensate for the poor start preconditioner and convergence can fail. This interpretation is supported by additional tests that have been made. The TM data were inverted again, but the apparent resistivities were downgraded, so that the data set was dominated by the phases. The phase is less sensitive to near-surface anomalies. The test showed that the inversion converged perfectly when using preconditioner $M_2$. This means that resistive structures near the stations can indeed have influence on the performance of preconditioner $M_2$. Further support is given by synthetic tests, where data calculated from a simple model with a conductive background and a resistive block located at the center at shallow depth were inverted. The NLCG inversion with $M_2$ converged again perfectly for pure TE data, but failed to converge for TM data. Clearly, further tests have to be performed to confirm this hypothesis. Additionally, detailed analysis of the sensitivities of the TM data would help to understand how the search direction is influenced by them. For future users of my code, I recommend to start the inversion with preconditioner $M_2$. If any problems occur, the inversion can be repeated with $M_3$, which achieved stable convergence for all data sets.

The main aim of my thesis was to write a robust NLCG algorithm with the best possible set-up. In order to achieve this goal, different input parameters pertaining to the search direction and the line search were tested. Starting with the search direction, the results showed that the techniques of Polak-Ribiere, Hestenes-Stiefel and Hager-Zhang converged the best. The techniques of Fletcher-Reeves and Dai-Yuan converged slightly slower. Polak-Ribiere is the most often used technique for EM inversion, applied for instance by Newman and Alumbaugh (2000), Rodi and Mackie (2001) and Kamm and Pedersen (2014). The original study of Polak and Ribiere (1969) has shown that PR performs better than FR in most cases. DY has similar properties to FR (Nocedal and Wright, 2006), which could also be shown by the convergence tests in this thesis. Thus, the technique of Polak-Ribiere, together with HS and HZ are favourable. PR is now implemented as default in the NLCG inversion in EMILIA. In a next test, different trial steps were taken and convergence was analysed. It turned out that two techniques are to be preferred. First, a low-cost method is to define the trial step in the first iteration as the linear CG step (involving one pseudo-forward problem) and to adopt the best step size from the previous iteration for subsequent
iterations. This led to good convergence for all data sets, except for the case when inversion was done with preconditioner $M_2$ and convergence failed. In this case, backtracking was performed and the step sizes got very small. Thus, the next iteration was started with a too low trial step. A better solution is then to compute the linear CG step for all iterations. The additional costs caused by these evaluations can be compensated by skipping the parabolic fit that was usually performed after the trial step. Then, this technique was the fastest of all the approaches that were tested here. However, the parabolic fit can reduce the objective function significantly when the linear CG step is a bad choice. The third option, namely to define the first trial step by delimiting the maximum change of the model parameters, did not provide better convergence. A possible improvement of the trial step procedure would be to calculate the linear CG step in the first iteration and when backtracking was performed without any success. For all other iterations, the best step size from the previous iteration could be taken as the new trial step. The third test was done to evaluate whether it is sufficient to constrain the line search by the first Wolfe condition or if it is necessary to search for the minimum more exactly by applying the second Wolfe condition, as well. Tests showed that in most cases, the first Wolfe condition is adequate to obtain good convergence. A more precise line search leads to a lower objective function after the same number of iterations, but the gradient has to evaluated more often. This involves two pseudo-forward problems per gradient. It could be shown that the higher convergence rate per iteration cannot compensate for the additional costs in computation. Therefore, an inexact line search as introduced by Newman and Alumbaugh (2000) is sufficient.
6 Conclusions

In this thesis, I implemented the method of NLCG in EMILIA for MT and RMT data in two dimensions. NLCG is able to invert the data much faster than the so far available Occam and damped Occam inversion techniques, because it circumvents the computation of the Jacobian matrix and avoids expensive matrix products, which are classical bottlenecks in inversion. The matrix-vector products of the Jacobian that are needed for the NLCG inversion were implemented for the quasi-static case and for displacement currents in the TE, TM and DET mode. For small data sets ($N < 1000$ and $M < 5000$), NLCG was two to four times faster than OC and DO. This gained time can for instance be used to test different Lagrange multipliers to obtain the optimal model. For large data sets ($N > 1000$ and $M > 10000$) NLCG is more than one order of magnitude faster than OC and DO. The actual strength of NLCG is to invert huge data sets in relatively short computation times without loosing quality in the final output models.

Preconditioning has, as expected, improved convergence for the NLCG inversion significantly. However, no perfect preconditioner could be selected. In addition to the already existing approaches of Newman and Alumbaugh (2000) and Rodi and Mackie (2001), a combination of these two preconditioners was introduced, which showed more stable convergence than the preconditioner from Newman and Alumbaugh (2000) and faster convergence than the one used by Rodi and Mackie (2001)$^1$. The problems that occurred for preconditioner $M_2$ could be related to the TM mode. For now, the reasons remain however unclear. The combination of a Jacobian-based start preconditioner and the low-cost BFGS update seems to be definitely a reasonable approach. However, further tests are necessary to find a preconditioner that has fast and stable convergence for arbitrary data sets. One suggestion would be to use a preconditioner that includes more than just the diagonal of the Jacobian, as for instance suggested by Hu et al. (2011) for full-waveform seismic inversion. At any rate, adopting approaches from seismology into EM inverse problems is a good advice, because the method of NLCG has been tested extensively in these areas. This thesis has also shown that it is most often sufficient to do an inexact line search by using a weak constraint such as the first Wolfe condition. Further tests are certainly necessary to support this conclusion.

The method of NLCG can now be used to invert data from future field studies. As mentioned before, NLCG has its strength in large inverse problems. Therefore, a prospective task will be to implement the matrix-vector product operations in a future 3D MT and RMT part of EMILIA, where the number of model parameters will be much higher. Furthermore, the code can be extended to the modes of vertical magnetic transfer functions (VMT) (e.g. Kalscheuer et al., 2008) and inter-station magnetic transfer functions (e.g. Hübert et al., 2009) and finally also to the method of direct-current resistivity, which are already implemented in EMILIA for the other inversion techniques.

---

$^1$To be correct, the newly introduced preconditioner $M_3$ was compared to a simplified version of the preconditioner applied by Rodi and Mackie (2001), because only the diagonal was used here.
7 Acknowledgements

I would like to thank my supervisor, Dr. Thomas Kalscheuer, for his assistance during my master thesis. He explained to me the structure of the inversion program EMILIA and helped me to solve several problems that occurred in this context. I am also grateful to my co-supervisor, Prof. Dr. Hansruedi Maurer, for his comments and distance support from Switzerland. Special thanks go to Jochen Kamm, who took the time to explain to me the mathematical details of the NLCG algorithm. I also want to express my gratitude to Fei Huang for his useful comments, which improved the thesis. Last but not least, my thanks go to Frank Guldstrand who helped me to write the popular scientific summary in Swedish.
8 Bibliography


Appendices
A Finite-difference approach

The finite-difference approximation (FDA) is the simplest form of EM modelling. The elements of the system matrix $\mathbf{K}$ in Equation (3.24) can be assembled node-wise. A rectangle with area $A$ is spread over four resistivity cells, including the center node according to Figure A.1. The corners of the rectangle are at the center points of the adjacent cells. All the cells have a given admittivity. The impedivity $\hat{z}$ is assumed to be constant over the mesh for a given frequency. The following derivations are done for a specific node $(i, j)$, but can be extended to all inner nodes. Some special treatment is required for the boundary nodes, i.e., those nodes which have less than four neighbour nodes. This section is strictly following Kalscheuer et al. (2008), who based their derivations on Aprea et al. (1997). The differential equations that have to be solved are Equation (3.18) for the TE mode and Equation (3.19) for the TM mode. For the TE mode, we can integrate the corresponding equation over $A$ and express it as a contour integral with the use of Gauss’ flux theorem to get

$$\int_A (\hat{z}\hat{y}E_x)\,dA = \int_A \nabla \cdot \nabla E_x\,dA = \int_{\partial A} (\mathbf{n} \cdot \nabla E_x)\,dl,$$  \hspace{1cm} (A.1)$$

where $\mathbf{n}$ is a unit normal vector directed perpendicularly away from the edges of $A$. The surface integral for the part located to the upper right-hand side of the center node is for instance

$$\int_{\text{ur}}^A (\hat{z}\hat{y}E_x)\,dA \approx \hat{z}\hat{y}_{i+1/2,j-1/2}E_x^{i,j} \frac{\Delta y_{i+1/2} \Delta z_{j-1/2}}{4}.$$  \hspace{1cm} (A.2)$$

The total surface integral can then be expressed as

$$\int_A (\hat{z}\hat{y}E_x)\,dA \approx \frac{1}{4} \hat{z}\hat{y}_{i,j}^{\text{int}} E_x^{i,j} + O(\Delta^3),$$  \hspace{1cm} (A.3)$$

with

$$\hat{y}_{i,j}^{\text{int}} = \hat{y}_{i-1/2,j-1/2} \Delta y_{i-1/2} \Delta z_{j-1/2} + \hat{y}_{i+1/2,j-1/2} \Delta y_{i+1/2} \Delta z_{j-1/2} + \hat{y}_{i-1/2,j+1/2} \Delta y_{i-1/2} \Delta z_{j+1/2} + \hat{y}_{i+1/2,j+1/2} \Delta y_{i+1/2} \Delta z_{j+1/2}.$$  \hspace{1cm} (A.4)$$
The contour integral for the lower edge of \( A \) can for example be written as

\[
\int_{\partial A} (\mathbf{n} \cdot \nabla E_x) \, dl \approx \left( \frac{E_x^{i,j+1} - E_x^{i,j}}{\Delta z_{j+1/2}} \right) \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{2}.
\] (A.5)

The total contour integral is

\[
\int_{\partial A} (\mathbf{n} \cdot \nabla E_x) \, dl \approx \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{2} \left( \frac{E_x^{i,j+1} - E_x^{i,j}}{\Delta z_{j+1/2}} - \frac{E_x^{i+1,j} - E_x^{i,j}}{\Delta z_{j-1/2}} \right) + \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{2} \left( \frac{E_x^{i+1,j} - E_x^{i,j}}{\Delta y_{i+1/2}} - \frac{E_x^{i,j} - E_x^{i,j-1}}{\Delta y_{i-1/2}} \right) + O(\Delta^3).
\] (A.6)

\( O(\Delta^2) \) and \( O(\Delta^3) \) are higher order terms that are neglected in the FDA. The expressions in Equations (A.3) and (A.6) can be inserted in Equation (A.1) and rearranged to obtain

\[
0 = 2 \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{\Delta z_{j+1/2}} E_x^{i,j+1} + 2 \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{\Delta z_{j-1/2}} E_x^{i,j-1} + 2 \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{\Delta y_{i+1/2}} E_x^{i+1,j} + 2 \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{\Delta y_{i-1/2}} E_x^{i,j} - 2 \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{\Delta z_{j+1/2}} E_x^{i+1,j} - 2 \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{\Delta z_{j-1/2}} E_x^{i,j} - 2 \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{\Delta y_{i+1/2}} E_x^{i+1,j} - 2 \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{\Delta y_{i-1/2}} E_x^{i,j} + \Delta z_{i+1/2} E_x^{i,j},
\] (A.7)

for all inner mesh nodes. The index for \( K \) is global and defined as \( k = (i-2)(N_{yb} - 1) + j - 1 \). For the inner mesh nodes adjacent to the boundaries, only three (for the edges) or two (for the corners) inner neighbour nodes are existent. There, the corresponding boundary values are multiplied with the appropriate FDA coefficients of Equation (A.7) and transferred to the right-hand side vector \( s \) in Equation (3.24).

The derivations for the TM mode can be performed similarly. Applying Gauss’ flux theorem to Equation (3.19), we get

\[
\int_A \varepsilon \mathbf{H}_z \, dA = \int_A (\nabla \cdot \frac{1}{\varepsilon} \nabla \mathbf{H}_z) \, dA = \int_{\partial A} (\mathbf{n} \cdot \frac{1}{\varepsilon} \nabla \mathbf{H}_z) \, dl.
\] (A.8)

The total surface integral over the rectangle with area \( A \) gives

\[
\int_A \varepsilon \mathbf{H}_z \, dA \approx \frac{1}{4} \varepsilon \mathbf{H}_z^{i,j} \left[ (\Delta y_{i+1/2} + \Delta y_{i-1/2}) + (\Delta z_{j+1/2} + \Delta z_{j-1/2}) \right] + O(\Delta^3)
\] (A.9)
and the total contour integral can be written as

$$
\int_{\partial A} (\mathbf{n} \cdot \frac{1}{\sqrt{2}} \nabla H_x) \, dl \approx \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{2} \left( \frac{1}{\bar{\hat{\nu}''_{i,j}}} \Delta z_{j+1/2} - \frac{1}{\bar{\hat{\nu}''_{i,j}}} \Delta z_{j-1/2} \right) \\
+ \Delta z_{j+1/2} + \Delta z_{j-1/2} \left( \frac{1}{\bar{\hat{\nu}''_{i,j}}} \frac{H_x^{i+1,j} - H_x^{i,j}}{\Delta y_{i+1/2}} - \frac{1}{\bar{\hat{\nu}''_{i,j}}} \frac{H_x^{i-1,j} - H_x^{i,j}}{\Delta y_{i-1/2}} \right) \\
+ O(\Delta^2),
$$

(A.10)

where the averaged admittivities over the corresponding edge are

$$
\frac{1}{\bar{\hat{\nu}''_{i,j}}} = \frac{\frac{1}{y_{i+1/2,j+1/2}} \Delta y_{i+1/2} + \frac{1}{y_{i-1/2,j+1/2}} \Delta y_{i-1/2}}{\Delta y_{i+1/2} + \Delta y_{i-1/2}},
\quad (A.11)
$$

$$
\frac{1}{\bar{\hat{\nu}''_{i,j}}} = \frac{\frac{1}{y_{i+1/2,j-1/2}} \Delta y_{i+1/2} + \frac{1}{y_{i-1/2,j-1/2}} \Delta y_{i-1/2}}{\Delta y_{i+1/2} + \Delta y_{i-1/2}},
\quad (A.12)
$$

$$
\frac{1}{\bar{\hat{\nu}''_{i,j}}} = \frac{\frac{1}{y_{i+1/2,j+1/2}} \Delta z_{j+1/2} + \frac{1}{y_{i+1/2,j-1/2}} \Delta z_{j-1/2}}{\Delta z_{j+1/2} + \Delta z_{j-1/2}},
\quad (A.13)
$$

$$
\frac{1}{\bar{\hat{\nu}''_{i,j}}} = \frac{\frac{1}{y_{i+1/2,j+1/2}} \Delta z_{j+1/2} + \frac{1}{y_{i-1/2,j-1/2}} \Delta z_{j-1/2}}{\Delta z_{j+1/2} + \Delta z_{j-1/2}},
\quad (A.14)
$$

Inserting the derived expressions in Equation (A.8) and rearranging gives

$$
0 = 2 \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{\Delta z_{j+1/2}} \frac{1}{\bar{\hat{\nu}''_{i,j}}} H_x^{i,j+1} + 2 \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{\Delta z_{j-1/2}} \frac{1}{\bar{\hat{\nu}''_{i,j}}} H_x^{i,j-1} + 2 \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{\Delta y_{i+1/2}} \frac{1}{\bar{\hat{\nu}''_{i,j}}} H_x^{i+1,j} + 2 \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{\Delta y_{i-1/2}} \frac{1}{\bar{\hat{\nu}''_{i,j}}} H_x^{i-1,j} \\
- \left[ 2 \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{\Delta z_{j+1/2}} \frac{1}{\bar{\hat{\nu}''_{i,j}}} + 2 \frac{\Delta y_{i+1/2} + \Delta y_{i-1/2}}{\Delta z_{j-1/2}} \frac{1}{\bar{\hat{\nu}''_{i,j}}} \\
+ 2 \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{\Delta y_{i+1/2}} \frac{1}{\bar{\hat{\nu}''_{i,j}}} + 2 \frac{\Delta z_{j+1/2} + \Delta z_{j-1/2}}{\Delta y_{i-1/2}} \frac{1}{\bar{\hat{\nu}''_{i,j}}} \right] H_x^{i,j}.
\quad (A.15)
$$

The computation of the auxiliary field components at the station locations, i.e. \( H_y \) for the TE mode and \( E_y \) for the TM mode, can be done by applying a second order Taylor series expansion to Equations (3.11b) and (3.12b). Using a central difference scheme, \( H_x \) and \( E_y \) can be expressed as functions of \( E_x \) and \( H_x \) at the same node and the two vertically adjacent nodes.
Starting with the TE mode, this leads to (Kalscheuer et al., 2008)

\[-z H_{y}^{i,j} = \left( \frac{\partial E_x}{\partial z} \right)_{i,j} \]

\[= \frac{\Delta z_{j+1/2} \Delta z_{j-1/2}}{\Delta z_{j+1/2} + \Delta z_{j-1/2}} \left\{ E_x^{i,j+1} - E_x^{i,j-1} \right\} \]

\[-\frac{1}{\Delta z_{j+1/2}^2} - \frac{1}{\Delta z_{j-1/2}^2} + \frac{1}{2} \hat{\xi} (\hat{y}_{i,j}^d - \hat{y}_{i,j}^u) E_x^{i,j}, \]  \hspace{1cm} (A.16)

with

\[\hat{y}_{i,j}^d = \frac{\hat{y}_{i+1/2,j+1/2} \Delta y_{i+1/2} + \hat{y}_{i-1/2,j+1/2} \Delta y_{i-1/2}}{\Delta y_{i+1/2} + \Delta y_{i-1/2}}, \]  \hspace{1cm} (A.17)

\[\hat{y}_{i,j}^u = \frac{\hat{y}_{i+1/2,j-1/2} \Delta y_{i+1/2} + \hat{y}_{i-1/2,j-1/2} \Delta y_{i-1/2}}{\Delta y_{i+1/2} + \Delta y_{i-1/2}}, \]  \hspace{1cm} (A.18)

For the TM mode, the change of the admittivity has to be taken into account. Hence, the current density at the corresponding node is computed first and can be written as (Kalscheuer et al., 2008)

\[j_{y}^{i,j} = \left( \frac{\partial H_x}{\partial z} \right)_{i,j} = \frac{N_{i,j}}{\Delta z_{j+1/2} - \Delta z_{j-1/2}}, \]  \hspace{1cm} (A.19)

where

\[N_{i,j} = \frac{\Delta z_{j-1/2}}{\Delta z_{j+1/2}} H_{x}^{i,j+1} - \frac{\Delta z_{j+1/2}}{\Delta z_{j-1/2}} H_{x}^{i,j-1} - \frac{O_{i,j}}{2} \left( \frac{H_{x}^{i+1,j}}{\Delta y_{i+1/2}} - \frac{H_{x}^{i,j-1}}{\Delta y_{i-1/2}} \right) \]

\[+ \left[ \frac{\Delta z_{j+1/2}}{\Delta z_{j-1/2}} - \frac{\Delta z_{j-1/2}}{\Delta z_{j+1/2}} + \frac{1}{2} \frac{\Delta y_{i-1/2} - \Delta y_{i+1/2}}{\Delta y_{i-1/2} \Delta y_{i+1/2}} O_{i,j} \right] \]

\[+ \frac{\hat{\xi}}{2} \frac{\Delta z_{j-1/2} \Delta z_{j+1/2}}{\Delta y_{i-1/2} \Delta y_{i+1/2}} \left(\hat{y}_{i,j}^d - \hat{y}_{i,j}^u\right) H_{x}^{i,j}, \]  \hspace{1cm} (A.20)

with

\[O_{i,j} = \frac{\Delta z_{j-1/2} \Delta z_{j+1/2}}{\Delta y_{i-1/2} + \Delta y_{i+1/2}} \left[ \hat{y}_{i,j}^d \left( \frac{1}{\hat{y}_{i+1/2,j+1/2}} - \frac{1}{\hat{y}_{i-1/2,j+1/2}} \right) \right. \]

\[- \left. \hat{y}_{i,j}^u \left( \frac{1}{\hat{y}_{i+1/2,j-1/2}} - \frac{1}{\hat{y}_{i-1/2,j-1/2}} \right) \right]. \]  \hspace{1cm} (A.21)

To obtain a representative value of $E_{y}^{i,j}$, the current density $j_{y}^{i,j}$ is usually divided by an average admittivity value at node $(i, j)$, i.e.

\[E_{y}^{i,j} = \frac{j_{y}^{i,j}}{\bar{y}_{i,j}}, \]  \hspace{1cm} (A.22)

where

\[\bar{y}_{i,j}^{avg} = \frac{\hat{y}_{i,j}^{int}}{\Delta y_{i-1/2} \Delta z_{j-1/2} + \Delta y_{i+1/2} \Delta z_{j-1/2} + \Delta y_{i-1/2} \Delta z_{j+1/2} + \Delta y_{i+1/2} \Delta z_{j+1/2}}. \]  \hspace{1cm} (A.23)
B Sensitivity matrix-vector products

B.1 Sensitivity matrix computation

In EMILIA, the sensitivity matrices are computed with the sensitivity equation method according to Rodi (1976) and Rodi and Mackie (2001). However, the data in EMILIA are in logarithmic apparent resistivities and phases. Furthermore, the model parameters are in logarithmic resistivities. Hence, starting with $\rho_{app}$ from Equation (3.21), the derivative of the $i$-th datum with the $j$-th model parameter can be expressed as

$$\frac{\partial \log_{10}(\rho_{app}^i)}{\partial \log_{10}(m^j)} = \frac{m^j}{\rho_{app}^i} \frac{\partial \rho_{app}^i}{\partial m^j} = \frac{m^j}{\rho_{app}^i} \frac{1}{\rho_{app}^i} \frac{\partial |Z_i|^2}{\partial m^j} = m^j \frac{1}{\rho_{app}^i} \frac{\partial |Z_i|^2}{\partial m^j} = m^j \frac{1}{\rho_{app}^i} \frac{\partial |Z_i|^2}{\partial m^j} \left( Z_i \frac{\partial Z_i^*}{\partial m^j} + Z_i \frac{\partial Z_i}{\partial m^j} \right) = m^j \frac{2}{\rho_{app}^i} \frac{\partial |Z_i|^2}{\partial m^j} \left( Z_i^* \frac{\partial Z_i^*}{\partial m^j} \right),$$

(B.1)

where $Z_i^*$ is the complex conjugate of $Z_i$. For the phase from Equation (3.22), the derivation w.r.t. the model parameters is

$$\frac{\partial \phi^i}{\partial \log_{10}(m^j)} = \frac{\partial m^j}{\partial \log_{10}(m^j)} \frac{\partial \phi^i}{\partial m^j} = \ln(10)m^j \frac{\partial \phi^i}{\partial m^j} = \ln(10)m^j \frac{1}{1 + \left( \frac{\text{Im}(Z_i)}{\text{Re}(Z_i)} \right)^2} \frac{\partial}{\partial m^j} \left( \frac{\text{Im}(Z_i)}{\text{Re}(Z_i)} \right) = \ln(10)m^j \cos^2(\phi^i) \left[ \frac{1}{\text{Re}(Z_i)} \frac{\partial}{\partial m^j} \text{Im}(Z_i) - \frac{\text{Im}(Z_i)}{\text{Re}(Z_i)} \frac{\partial}{\partial m^j} \text{Re}(Z_i) \right] = \ln(10)m^j \cos^2(\phi^i) \left[ \text{Im} \left( \frac{\partial Z_i}{\partial m^j} \right) - \tan(\phi^i) \text{Re} \left( \frac{\partial Z_i}{\partial m^j} \right) \right].$$

(B.2)

It can be seen in Equations (B.1) and (B.2) that the derivative of the impedances w.r.t. the model parameters $\frac{\partial Z_i}{\partial m^j}$ is needed to get the sensitivities. Following Kalscheuer et al. (2008) for the subsequent derivations, the impedance can be written as

$$Z_i(m) = \frac{a_i(m)^T x(m)}{b_i(m)^T x(m)},$$

(B.3)

where $x(m)$ is a vector with the $E_x$ components for the TE mode or with the $H_x$ components for the TM mode at the inner mesh nodes and $a_i(m)$ and $b_i(m)$ are coefficient vectors from the computation of the auxiliary fields in Equations (A.16) to (A.23). The derivative of the $i$-th datum w.r.t. the $j$-th model
parameter can then be expressed as

\[
\frac{\partial Z_i(m)}{\partial m_j} = \left( \frac{1}{b_i^T x} \frac{\partial a_i}{\partial m_j} - \frac{a_i^T x}{(b_i^T x)^2} \frac{\partial b_i}{\partial m_j} \right)^T x + \left( \frac{1}{b_i^T x} a_i - \frac{a_i^T x}{(b_i^T x)^2} b_i \right)^T \frac{\partial x}{\partial m_j}. \tag{B.4}
\]

This can be split into two parts

\[
J_{ij}^1 = \left( \frac{1}{b_i^T x} \frac{\partial a_i}{\partial m_j} - \frac{a_i^T x}{(b_i^T x)^2} \frac{\partial b_i}{\partial m_j} \right)^T x \tag{B.5}
\]

and

\[
J_{ij}^2 = c_i^T \frac{\partial x}{\partial m_j}, \tag{B.6}
\]

where

\[
c_i = \frac{1}{b_i^T x} a_i - \frac{a_i^T x}{(b_i^T x)^2} b_i. \tag{B.7}
\]

From the forward problem, we can derive

\[
K \frac{\partial x}{\partial m_j} = - \frac{\partial K}{\partial m_j} x + \frac{\partial s}{\partial m_j}, \tag{B.8}
\]

which yields

\[
J_{ij}(m) = J_{ij}^1 + c_i^T K^{-1} \left( - \frac{\partial K}{\partial m_j} x + \frac{\partial s}{\partial m_j} \right)
= J_{ij}^1 + u_i^T \left( - \frac{\partial K}{\partial m_j} x + \frac{\partial s}{\partial m_j} \right) \underbrace{J_{ij}^2}_j
\tag{B.9}
\]

for the sensitivities, where \(u_i\) is the solution to the pseudo-forward problem

\[
K^T u_i = c_i. \tag{B.10}
\]

Note that the first part of Equation (B.9) is reformulated to obtain a system that has to be solved \(N\) times (i.e. for each datum). On the other hand, \(M\) steps would be required to solve Equation (B.8). Because \(M > N\) for multi-dimensional EM problems, solving Equation (B.10) instead of (B.8) is therefore reasonable.

For the TE mode, the numerator and denominator of Equation (B.3) are given by (Kalscheuer et al., 2008)

\[
a_i^T x = E_i^i, \tag{B.11}
\]

\[
b_i^T x = H_y^i, \tag{B.12}
\]

where the superscript \(i\) for the field components is now the station number (not the node). The entries of
vector \( \mathbf{a} \) are zero except for the \( i \)-th entry, thus (Kalscheuer et al., 2008)

\[
\mathbf{c}_i = \frac{1}{H_y} \left( 0, \ldots, 0, \frac{1}{i\text{-th entry}}, 0, \ldots, 0 \right) - \frac{Z_{xy}^i}{H_y} \mathbf{b}_i \tag{B.13}
\]

For the TM mode, the entries of vector \( \mathbf{b} \) are zero except for the \( i \)-th entry, thus (Kalscheuer et al., 2008)

\[
\mathbf{a}_i^T \mathbf{x} = E_y^i, \tag{B.14}
\]
\[
\mathbf{b}_i^T \mathbf{x} = H_x^i \tag{B.15}
\]

and

\[
\mathbf{c}_i = \frac{1}{H_x} \mathbf{a}_i - \frac{Z_{yx}^i}{H_x} \left( 0, \ldots, 0, \frac{1}{i\text{-th entry}}, 0, \ldots, 0 \right). \tag{B.16}
\]

The derivatives of the boundary values w.r.t. the model parameters are \( \frac{\partial s}{\partial m_j} \approx 0 \). I refer to Kalscheuer et al. (2008) for the derivation of the remaining elements, that is \( \frac{\partial K}{\partial m_j} \) and \( J_{ij}^1 \). For the NLCG algorithm, the sensitivity matrix does never have to be computed completely. The next section will explain how the full evaluation of \( \mathbf{J} \) can be avoided.

**B.2 Sensitivity matrix-vector product computation**

For the NLCG code in EMILIA, one matrix-vector product of the Jacobian has to be computed in each iteration. An additional matrix-vector product is computed when the linear CG step is chosen as a trial step (cf. Chapter 3.2). This section describes how matrix-vector products of the Jacobian, needed for the gradient of the objective function and the linear CG step, are computed. I follow Rodi and Mackie (2001) for most of the derivations in this section. The following technique is strongly related to the adjoint-state method which is used to compute the gradient in seismic inversion (see e.g. Pratt et al. (1998) and Plessix (2006) for details).

Starting with the linear CG step, it can be seen in Equation (3.46b) that the denominator contains the matrix-vector product \( \mathbf{Jp} \). This can be split into two parts, namely

\[
\mathbf{Jp} = \mathbf{J}_1 \mathbf{p} + \mathbf{J}_2 \mathbf{p} \tag{B.17}
\]

from Equation (B.9). The first part \( \mathbf{J}_1 \mathbf{p} \) can be calculated explicitly, because the complete evaluation of \( \mathbf{J}_1 \) is not very time-consuming. The expensive part is \( \mathbf{J}_2 \). An expression can be derived where \( \mathbf{J}_2 \mathbf{p} \) is computed directly without evaluating \( \mathbf{J}_2 \). By summing the product of the elements of \( \mathbf{J}_2 \) with the search direction over all model parameters, from Equation (B.6) we get

\[
\sum_{j=1}^{M} j_{2j}^i p^j = \mathbf{c}_i^T \mathbf{t}, \tag{B.18}
\]

where \( \mathbf{t} \) is given by

\[
\mathbf{t} = \sum_{j=1}^{M} p^j \frac{\partial \mathbf{x}}{\partial m^j}. \tag{B.19}
\]
From Equation (B.8), it can be seen that

\[ Kt = \sum_{j=1}^{M} p^j \left[ \frac{\partial s}{\partial m^j} - \left( \frac{\partial K}{\partial m^j} \right) x \right] \]  

(B.20)

and we assume that \( \frac{\partial s}{\partial m} = 0 \). Hence, to obtain \( J_2 p \) from Equation (B.18), the pseudo-forward problem in (B.20) has to be solved.

The matrix-vector product \( J^T q \) is needed to compute the gradient in Equation (3.38), where \( q = C_d^{-1}(d_{obs} - F(m)) \). The first part \( J_1^T q \) is again computed explicitly. For the second part, summing the product of the elements of \( J_2^T q \) with \( q \) over all stations, from Equation (B.9) we obtain

\[ \sum_{i=1}^{N} q^j_j^j = r^T \left[ \frac{\partial s}{\partial m^j} - \left( \frac{\partial K}{\partial m^j} \right) x \right] , \]  

(B.21)

where vector \( r \) is defined by

\[ r = \sum_{i=1}^{N} q^j u^i . \]  

(B.22)

From Equation (B.10) the pseudo-forward problem

\[ Kr = \sum_{i=1}^{N} q^i c^i \]  

(B.23)

is obtained. Therefore, one can solve Equation (B.23) and then evaluate \( J_2^T q \) by Equation (B.21).

The sequence presented here assumes that the data are complex impedances. In EMILIA, the data are transformed to apparent resistivities and phases. As it can be seen from Equations (B.1) and (B.2), the derivatives of the impedances w.r.t. the model parameters have to be multiplied by either data or model specific parameters. For instance, for the apparent resistivity the parameters \( m^j \), \( \rho_{app} \) as well as \( Z^j \) are multiplied by \( \frac{\partial Z}{\partial m^j} \) to obtain the derivative. Nested loops are used in EMILIA to compute the Jacobian. The main loop is performed over all the modes (i.e. TE, TM or DET), the second loop is over the different frequencies (i.e. \( \omega \)) and the third loop is over the stations. The shortcut of computing \( J_2^T q \) directly is only possible when the pseudo-forward problem (Equation (B.23)) is solved outside of the station loop. Thus, all the station specific parameters, i.e. \( \frac{1}{\rho_{app}} \) and \( Z^j \), have to be included in the coefficient \( c_i \) to allow an evaluation of Equation (B.23) after the station loop. From Equation (B.2), for the phase the coefficient \( c_i \) has to be multiplied by the station specific parameters \( \cos^2(\phi^i) \Re(Z^i) \) for the imaginary part and additionally by \( \tan(\phi^i) \) for the real part. The summation in Equation (B.21) is then performed over the entire vector \( q \), thus including both phases and apparent resistivities. It is therefore not possible to take the real part or imaginary part out of different elements of \( J_2^T q \). However, this is needed to compute the derivative of the phase (cf. Equation (B.2)). Therefore, the elements of \( q \) for the phase with the pre-factors of the imaginary part of \( \frac{\partial Z}{\partial m^j} \) are summed separately. Hence, a second pseudo-forward problem has to be solved. Note that the part of \( J_2^T q \) with all the apparent resistivities and the phases with the pre-factors of the real part of \( \frac{\partial Z}{\partial m^j} \) can be solved in one pseudo-forward problem, because only the real part is needed. Bringing it all together, to evaluate \( J_2^T q \), the solution of two pseudo-forward problems is required. For \( J_2 p \) however, the station specific parameters can be included after the summation in the corresponding elements of \( J_2 p \). The model specific parameters \( m^j \) are present in both the derivatives of the phases and the apparent resistivities and thus, the solution of one pseudo-forward problem is sufficient.
C Input parameters for NLCG algorithm

My NLCG algorithm in EMILIA is very flexible. The default set-up allows for stable and robust inversion of EM data. However, the user can choose out of different set-up parameters to improve convergence rates and/or inversion results for specific data sets. Table C.1 shows and explains the parameters that can be selected by the user before starting the inversion.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>max_iter</td>
<td>Maximum number of iterations for the entire NLCG inversion. Default = 100</td>
</tr>
<tr>
<td>conv_crit</td>
<td>Parameter that defines the stopping criterion of the NLCG algorithm: abs for absolute RMS, diff for RMS difference between two subsequent iterations, rel_diff for RMS relative difference between two subsequent iterations. Default = abs</td>
</tr>
<tr>
<td>rms_desired</td>
<td>Desired RMS when NLCG algorithm should stop (when conv_crit = abs). Default = 1.0</td>
</tr>
<tr>
<td>rms_diff</td>
<td>Desired RMS difference when NLCG algorithm should stop (when conv_crit = diff). Default = 0.001</td>
</tr>
<tr>
<td>rms_rel_diff</td>
<td>Desired RMS relative difference when NLCG algorithm should stop (when conv_crit = rel_diff). Default = 0.05</td>
</tr>
<tr>
<td>preconditioner</td>
<td>Preconditioner that is used for the NLCG inversion: non = no preconditioner, roddiag = preconditioner according to Equation (3.50) (M1), newman = preconditioner according to Equation (3.51) (M2), combo = combination of the two other preconditioners (M3). Default = combo</td>
</tr>
<tr>
<td>rodi_update</td>
<td>If preconditioner = roddiag, this parameter defines how often the preconditioner is updated. The specified number corresponds to the number of iterations during which the preconditioner is held constant. Default = 20</td>
</tr>
<tr>
<td>lambda</td>
<td>Lagrange multiplier in logarithmic form (w.r.t. base 10) for the model regularisation. Default = 2</td>
</tr>
<tr>
<td>alpha_min</td>
<td>Minimum step size that is allowed for the NLCG inversion. Default = $10^{-7}$</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>alpha_max</td>
<td>Maximum step size that is allowed for the NLCG inversion. Default = 10</td>
</tr>
<tr>
<td>iter_loop_step_max</td>
<td>Maximum number of iterations in line search algorithm (only necessary when second Wolfe condition is used). Default = 5</td>
</tr>
<tr>
<td>iter_loop_zoom_max</td>
<td>Maximum number of iterations in Zooming algorithm. Default = 5</td>
</tr>
<tr>
<td>wolfe_condition</td>
<td>Defines which constraints for sufficient decrease are used: 1 = only first Wolfe condition is used, 2 = both first and second Wolfe condition are used. Default = 1</td>
</tr>
<tr>
<td>c1</td>
<td>Coefficient for first Wolfe condition. Default = 0.001</td>
</tr>
<tr>
<td>c2</td>
<td>Coefficient for second Wolfe condition. Default = 0.1</td>
</tr>
<tr>
<td>alpha_factor</td>
<td>Factor to increase $\alpha$ when the first Wolfe condition is satisfied, but not the second and the gradient is still negative. Default = 2.0</td>
</tr>
<tr>
<td>iter_stpdes_max</td>
<td>Defines the number of iterations after which the search direction is reset to steepest descent. Default = 101</td>
</tr>
<tr>
<td>first_step_method</td>
<td>Defines the technique of determining the trial step: 1 = maximum allowed model change in first iteration (option 1 in Equation (3.46a)) and taking along previous $\alpha$ for subsequent iterations (option 1 in Equation (3.46b)), 2 = linear CG step in first iteration (option 2 in Equation (3.46a)) and linear CG step in subsequent iterations (option 2 in Equation (3.46b)), 3 = linear CG step in first iteration (option 2 in Equation (3.46a)) and taking along previous $\alpha$ for subsequent iterations (option 1 in Equation (3.46b)). Default = 3</td>
</tr>
<tr>
<td>dm_start</td>
<td>Defines $\Delta m$ for option 1 in Equation (3.46a) (only if first_step_method = 1). Default = 1.0</td>
</tr>
<tr>
<td>parabolic_fit</td>
<td>Yes = parabolic fit for first Wolfe condition is performed, no = parabolic fit for first Wolfe condition is not performed. Default = yes (setting parabolic_fit = no does only make sense when first_step_method = 2, otherwise convergence is slow!)</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cont_high_norm</td>
<td>Yes = inversion is continued even if higher objective function is found compared to previous iteration, no = inversion is stopped if higher objective function is found compared to previous iteration. Default = no</td>
</tr>
<tr>
<td>cont_high_rms</td>
<td>Yes = inversion is continued even if higher RMS is found compared to previous iteration, no = inversion is stopped if higher RMS is found compared to previous iteration. Default = yes</td>
</tr>
<tr>
<td>mode</td>
<td>Defines the technique to update $\beta$: polak = $\beta^{PR}$, fletcher = $\beta^{FR}$, hestenes = $\beta^{HS}$, dai = $\beta^{DY}$, hager = $\beta^{HZ}$. Default = polak</td>
</tr>
</tbody>
</table>
D Additional results

In this part of the appendix, additional plots are shown that were not presented in Chapter 4. Additionally, resistivity models obtained by different values of $\lambda$ are illustrated. For the best $\lambda$, resistivity models at different stages of the inversion process are presented. This exemplifies which parts of the model were changed in early iterations and where the model was improved for late iterations. For each data set, the performance of the preconditioners pertaining to the reduction of RMS is shown. Further results of the step size and search direction analysis are presented.

D.1 TE data from Smørgrav, Norway

![Comparison of convergence between Occam and NLCG inversion with preconditioner $M_2$ for the TE data from Smørgrav. $\lambda$ was set to $\sim 3.16$. For both inversion techniques, the desired RMS = 1.0 was not reached. The RMS of the NLCG inversion is decreasing continuously, whereas the RMS in Occam inversion is oscillating after reaching the minimum in iteration 6.](image-url)
Figure D.2: Solution convergence of the objective function (a) and RMS (b) for the NLCG inversion without preconditioner and with preconditioners $M_1$, $M_2$ and $M_3$ for the TE data from Smørgrav. $\lambda$ was set to $\sim 3.16$. The enhanced symbols mark the last iteration, i.e. $k = 100$.

Figure D.3: Resistivity models obtained by NLCG inversion with preconditioner $M_2$ after 100 iterations with $\lambda = 10^{0.5} \approx 3.16$ (RMS = 1.167) (a), $\lambda = 10$ (RMS = 1.501) (b) and $\lambda = 10^{1.5} \approx 31.6$ (RMS = 2.177) (c) for the TE data from Smørgrav.
Figure D.4: Resistivity models obtained by NLCG inversion with preconditioner $M_2$ and $\lambda \approx 3.16$ after 25 iterations (RMS = 1.207) (a), 50 iterations (RMS = 1.176) (b), 75 iterations (RMS = 1.168) (c) and 100 iterations (RMS = 1.167) (d) for the TE data from Smørgrav.
D.2 TM data from Smørgrav, Norway

Figure D.5: Comparison of convergence between Occam (10 iterations) and NLCG inversion (200 iterations) for the TM data from Smørgrav. Preconditioner $M_1$ was used and updated every 20-th iteration. For NLCG, $\lambda$ was set to 100. Occam’s inversion reaches the desired RMS (dashed line) after four iterations, whereas the method of NLCG stagnates at an RMS of 2.64.

Figure D.6: Solution convergence of the objective function (a) and RMS (b) for the NLCG inversion without preconditioner and with preconditioners $M_1$, $M_2$ and $M_3$ for the TM data from Smørgrav. $\lambda$ was set to 100. The enhanced symbols mark the last iteration, i.e. $k = 100$. 
Figure D.7: Resistivity models obtained by NLCG inversion with preconditioner $M_1$ after 43 iterations with $\lambda = 10^{0.5} \approx 3.16$ (RMS = 0.997) (a) and after 200 iterations with $\lambda = 10$ (RMS = 1.149) (b), $\lambda = 10^{1.5} \approx 31.6$ (RMS = 1.775) (c) and $\lambda = 100$ (RMS = 2.643) (d) for the TM data from Smørgrav.
Figure D.8: Resistivity models obtained by NLCG inversion with preconditioner $M_1$ and $\lambda \approx 31.6$ after 50 iterations (RMS = 1.905) (a), 100 iterations (RMS = 1.825) (b), 150 iterations (RMS = 1.775) (c) and 200 iterations (RMS = 1.775) (d) for the TM data from Smørørvik.
D.3 Joint inversion of TE and TM data from Smørgrav, Norway

Figure D.9: Solution convergence for the joint inversion of TE and TM data from Smørgrav of the objective function (a) and RMS (b) for four different Lagrange multipliers: $\lambda \approx 3.16$, $\lambda = 10$, $\lambda \approx 31.6$ and $\lambda = 100$. The NLCG inversion was performed with preconditioner $M_3$. The dashed line in (b) designates the desired RMS of 1.0.

Figure D.10: Solution convergence of the objective function (a) and RMS (b) for the NLCG inversion without preconditioner and with preconditioners $M_1$, $M_2$ and $M_3$ for the joint inversion of TE and TM data from Smørgrav. $\lambda$ was set to 10. The enhanced symbols mark the last iteration, i.e. $k = 100$. 
Figure D.11: Resistivity models obtained by NLCG inversion with preconditioner $M_3$ after 100 iterations with $\lambda = 10^{0.5} \approx 3.16$ (RMS = 1.110) (a), $\lambda = 10$ (RMS = 1.367) (b), $\lambda = 10^{1.5} \approx 31.6$ (RMS = 1.924) (c) and $\lambda = 100$ (RMS = 2.838) (d) for the joint inversion of TE and TM data from Smørgrav.
Figure D.12: Resistivity models obtained by NLCG inversion with preconditioner $M_3$ and $\lambda = 10$ after 25 iterations (RMS = 1.547) (a), 50 iterations (RMS = 1.410) (b), 75 iterations (RMS = 1.383) (c) and 100 iterations (RMS = 1.367) (d) for the joint inversion of TE and TM data from Smørgrav.
D.4 DET data from Skediga, Sweden

Figure D.13: Solution convergence for the DET data from Skediga of the objective function (a) and RMS (b) for four different Lagrange multipliers: $\lambda \approx 3.16$, $\lambda = 10$, $\lambda \approx 31.6$ and $\lambda = 100$. The inversions with $\lambda \approx 3.16$ and $\lambda = 10$ were stopped after 62 and 58 iterations, respectively, because a higher objective function was obtained. The NLCG inversion was performed with preconditioner $M_3$. The dashed line in (b) designates the desired RMS of 1.0.

Figure D.14: Solution convergence of RMS for NLCG inversion without preconditioner and with preconditioners $M_2$ and $M_3$ for the DET data from Skediga. Preconditioner $M_2$ was applied together with a different line search procedure. Instead of taking the best step from the previous iteration as the trial step, the linear CG step was taken for every iteration as the trial step. $\lambda$ was set to $\approx 31.6$. The enhanced symbols mark iteration 100.
Figure D.15: Resistivity models obtained by NLCG inversion with preconditioner $\mathbf{M}_3$ after 62 iterations with $\lambda = 10^{0.5} \approx 3.16$ (RMS = 1.707) (a), after 58 iterations with $\lambda = 10$ (RMS = 1.421) (b), after 100 iterations with $\lambda = 10^{1.5} \approx 31.6$ (RMS = 1.335) (c) and after 100 iterations with $\lambda = 100$ (RMS = 1.780) (d) for the DET data from Skediga. The inversions with $\lambda \approx 3.16$ and $\lambda = 10$ were stopped after 62 and 58 iterations, respectively, because a higher objective function was obtained.
Figure D.16: Resistivity models obtained by NLCG inversion with preconditioner $M_3$ and $\lambda \approx 31.6$ after 25 iterations (RMS = 2.213) (a), 50 iterations (RMS = 1.461) (b), 75 iterations (RMS = 1.372) (c) and 100 iterations (RMS = 1.335) (d) for the DET data from Skediga.
D.5 DET data from Åvrö, Sweden

Figure D.17: Solution convergence for the DET data from Åvrö with displacement currents of the objective function (a) and RMS (b) for three different Lagrange multipliers: $\lambda = 10$, $\lambda \approx 31.6$ and $\lambda = 100$. The inversions with $\lambda \approx 31.6$ and $\lambda = 100$ were stopped after 26 and 20 iterations, respectively, because difference convergence in RMS was reached. The NLCG inversion was performed with preconditioner $M_2$. The dashed line in (b) designates the desired RMS of 1.0.

Figure D.18: Solution convergence of RMS for the NLCG inversion without preconditioner and with preconditioners $M_2$ and $M_3$ for the DET data from Åvrö with displacement currents. $\lambda$ was set to 100. The enhanced symbols mark iteration 47 (red circles), 20 (green crosses) and 50 (black plus signs).
Figure D.19: Resistivity models obtained by NLCG inversion with preconditioner $M_2$ after 50 iterations with $\lambda = 10$ (RMS = 1.766) (a), after 26 iterations with $\lambda = 10^{1.5} \approx 31.6$ (RMS = 1.846) (b) and after 20 iterations with $\lambda = 100$ (RMS = 2.208) (c) for the DET data from Åvrö with displacement currents.
Figure D.20: Resistivity models obtained by NLCG inversion with preconditioner $M_2$ and $\lambda = 100$ after 5 iterations (RMS = 6.359) (a), 10 iterations (RMS = 2.532) (b) and 20 iterations (RMS = 2.208) (c) for the DET data from Avrø with displacement currents.
D.6 Convergence analysis of NLCG input parameters

Figure D.21: Solution convergence for the TM data from Smørgrav of the objective function (a) and RMS (b) for five different update techniques for $\beta$: Polak-Ribiere, Fletcher-Reeves, Hestenes-Stiefel, Dai-Yuan and Hager-Zhang. The NLCG inversion was performed with preconditioner $M_1$, updated every 20-th iteration. $\lambda$ was set to $\sim 3.16$. The desired RMS was reached after 76 (PR), 96 (FR), 75 (HS), 99 (DY) and 78 (HZ) iterations.

Figure D.22: Solution convergence for the DET data from Skediga of the objective function (a) and RMS (b) for five different update techniques for $\beta$: Polak-Ribiere, Fletcher-Reeves, Hestenes-Stiefel, Dai-Yuan and Hager-Zhang. The NLCG inversion was performed without preconditioner. $\lambda$ was set to $\sim 31.6$. The enhanced symbols mark iteration 100.
Figure D.23: Solution convergence for the TM data from Smørgrav of the objective function (a) and RMS (b) for five different trial step techniques: Linear CG step for the first iteration and adopting best $\alpha$ for subsequent iterations, linear CG step for all iterations, linear CG step for all iterations but without parabolic fit and defining trial step by setting maximum allowed change of model parameters to 1.0 and 2.0. The NLCG inversion was performed with preconditioner $M_1$, updated every 20-th iteration. $\lambda$ was set to $\sim 3.16$.

Figure D.24: Solution convergence for the DET data from Skediga of the objective function (a) and RMS (b) for five different trial step techniques: Linear CG step for the first iteration and adopting best $\alpha$ for subsequent iterations, linear CG step for all iterations, linear CG step for all iterations but without parabolic fit and defining trial step by setting maximum allowed change of model parameters to 1.0 and 2.0. The NLCG inversion was performed without preconditioner. $\lambda$ was set to $\sim 31.6$. 
Figure D.25: Solution convergence for the TE data from Smørgrav of the objective function (a) and RMS (b) for the line search using the first Wolfe condition only and using both the first and the second Wolfe condition. The NLCG inversion was performed with preconditioner $M_2$. $\lambda$ was set to $\sim 3.16$.

Figure D.26: Solution convergence for the TM data from Smørgrav of the objective function (a) and RMS (b) for the line search using the first Wolfe condition only and using both the first and the second Wolfe condition. The NLCG inversion was performed with preconditioner $M_1$, updated every 20-th iteration. $\lambda$ was set to $\sim 3.16$. The desired RMS of 1.0 was reached after 75 iterations using the first Wolfe condition and after 50 iterations using the second Wolfe condition.