Observatório Nacional

Ministério da
Ciência, Tecnologia e Inovação

MODELAGEM DIRETA E INVERSÃO DE CAMPOS GRAVITACIONAIS EM COORDENADAS ESFÉRICAS

Leonardo Uieda

Tese de Doutorado apresentada ao Programa de Pós-graduação em Geofísica do Observatório Nacional/MCTI, como parte dos requisitos necessários à obtenção do Título de Doutor em Ciências.

Orientador: Valéria Cristina Ferreira Barbosa

Rio de Janeiro
Abril de 2016

MODELAGEM DIRETA E INVERSÃO DE CAMPOS GRAVITACIONAIS EM COORDENADAS ESFÉRICAS

Leonardo Uieda

TESE SUBMETIDA AO PROGRAMA DE PÓS-GRADUAÇÃO EM GEOFÍSICA DO OBSERVATÓRIO NACIONAL/MCTI COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO TÍTULO DE DOUTOR EM CIÊNCIAS EM GEOFÍSICA.

Examinada por:

Dra. Valéria Cristina Ferreira Barbosa, ON/MCTI

Prof. Nome da Segunda Examinadora Sobrenome, Ph.D.

Dr. Nome da Terceira Examinadora Sobrenome, D.Sc.

Prof. Nome do Quarto Examinador Sobrenome, Ph.D.

Prof. Nome do Quinto Examinador Sobrenome, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
ABRIL DE 2016

Uieda, Leonardo

Modelagem direta e inversão de campos gravitacionais em coordenadas esféricas /Leonardo Uieda. – Rio de Janeiro: ON/MCTI, 2016.

XVII, 81 p.: il.; 29, 7cm.

Orientador: Valéria Cristina Ferreira Barbosa

Tese (doutorado) – ON/MCTI/Programa de Pós-graduação em Geofísica, 2016.

Bibliography: p. 72 – 81.

1. Gravimetria. 2. Inversão. 3. Software. I. , . II. Observatório Nacional/MCTI, Programa de Pós-graduação em Geofísica. III. Título.

Agradecimentos

Agradeço à CAPES pelo apoio financeiro na forma de uma bolsa de estudos.

Meus agradecimentos às pessoas que fizeram desse trabalho uma diversão:

Minha orientadora Valéria Cristina Ferreira Barbosa pela paciência, pelo incentivo, pela falta de paciência nas horas certas, pelo uso do chicote não-metafórico quando necessário, pelos conselhos pessoais e profissionais, pelas viagens e pelos 6 anos de amizade (que estão quase no limite de qualquer relacionamento, 7 anos, mas eu acho que essa teoria é furada).

Minha companheira Ana Caroline Colombo (Paper) que me apoiou sempre, mesmo nos dias de mau humor intenso, enxaqueca e desânimo. Obrigado por me incentivar a querer ser melhor a cada dia.

Meus pais Virgínia Sanches Uieda e Wilson Uieda que me inicializaram na vida acadêmica desde pequeno, sempre me estimularam a desenvolver meus interesses e nunca me reprimiram por corrigir os professores que achavam que baleias são peixes.

Os colegas do Observatório Nacional que potencializaram a aprendizagem e tornaram a vida acadêmica mais interessante: Birocolês, Saulinho, Rod, Bonilla, Bragança, Flora, Fillipe Claudio, Dionisio, Leo Miquelutti, *et al.*

Resumo da Tese apresentada ao Programa de Pós-graduação em Geofísica do Observatório Nacional/MCTI como parte dos requisitos necessários para a obtenção do título de Doutor em Ciências (D.Sc.)

MODELAGEM DIRETA E INVERSÃO DE CAMPOS GRAVITACIONAIS EM COORDENADAS ESFÉRICAS

Leonardo Uieda

Abril/2016

Orientador: Valéria Cristina Ferreira Barbosa

Programa: Geofísica

Apresentamos avanços metodológicos na área de modelagem direta e inversão regional de dados de gravimetria por satélite. Com esse fim, desenvolvemos dois projetos computacionais de código livre. O primeiro é um conjunto de programas de linha de comando feitos na linguagem C chamado *Tesseroids*. Os programas calculam o potencial, aceleração e tensor gradiente gravitacional de um prisma esférico, ou tesseroide. *Tesseroids* implementa e aprimora um algoritmo de discretização adaptativa para automaticamente garantir a acurácia das computações. Os resultados com testes numéricos mostram que, para obter o mesmo nível de acurácia, a aceleração gravitacional demanda uma discretização mais fina que o potencial. Por sua vez, o tensor gradiente gravitacional demanda discretização mais fina ainda que a aceleração. O segundo projeto computacional é o *Fatiando a Terra*, uma biblioteca feita na linguagem Python para inversão, modelagem direta, processamento e visualização de dados. A biblioteca permite que o usuário combine as ferramentas de modelagem direta e de inversão para implementar novos métodos de inversão. As ferramentas de modelagem direta incluem uma implementação do algoritmo utilizado no programa *Tesseroids*. Combinamos os recursos de inversão e modelagem direta com tesseroides do *Fatiando a Terra* para desenvolver um método rápido para a inversão não-linear de dados de gravidade. O método estima a profundidade da interface crosta-manto (a Moho) baseado em dados de gravidade utilizando uma aproximação esférica da Terra. Adaptamos o método de Bott, que é computacionalmente eficiente, para incluir regularização de suavidade e utilizar tesseroides ao invés

de prismas retangulares retos. A inversão é controlada por três hiper-parâmetros: o parâmetro de regularização, o contraste de densidade entre a Terra real e o modelo de referência (a Terra Normal) e a profundidade da Moho da Terra Normal. Aplicamos dois tipos de validação cruzada para estimar esses parâmetros de maneira automática. Testes com dados sintéticos confirmam a capacidade do método proposto de estimar os três hiper-parâmetros e o relevo suave da Moho. Finalmente, aplicamos o método de inversão desenvolvido para gerar um modelo de profundidade da Moho para a América do Sul. O modelo de profundidade da Moho estimado ajusta os dados de gravidade observados e as estimativas da profundidade da Moho provenientes da sismologia nas regiões oceânicas e nas partes central e leste do continente. Observamos desajustes aos dados na região dos Andes, onde a profundidade da Moho é a maior do continente. Nas bacias do Amazonas, Solimões e Paraná, o modelo ajusta os dados de gravidade mas não as estimativas da sismologia. Essas discrepâncias indicam a presença de anomalias de densidade na crosta ou manto superior, como sugerido anteriormente na literatura.

Abstract of Thesis presented to Observatório Nacional/MCTI as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

FORWARD MODELING AND INVERSION OF GRAVITATIONAL FIELDS IN SPHERICAL COORDINATES

Leonardo Uieda

April/2016

Advisor: Valéria Cristina Ferreira Barbosa

Department: Geophysics

We present methodological improvements to forward modeling and regional inversion of satellite gravity data. For this purpose, we developed two open-source software projects. The first is a C language suite of command-line programs called *Tesseroids*. The programs calculate the gravitational potential, acceleration, and gradient tensor of a spherical prism, or tesseroid. *Tesseroids* implements and extends an adaptive discretization algorithm to automatically ensure the accuracy of the computations. Our numerical experiments show that, to achieve the same level of accuracy, the gravitational acceleration components require finner discretization than the potential. In turn, the gradient tensor requires finner discretization still than the acceleration. The second open-source project is *Fatiando a Terra*, a Python language library for inversion, forward modeling, data processing, and visualization. The library allows the user to combine the forward modeling and inversion tools to implement new inversion methods. The gravity forward modeling tools include an implementation of the algorithm used in the *Tesseroids* software. We combined the inversion and tesseroid forward modeling utilities of *Fatiando a Terra* to develop a new method for fast non-linear gravity inversion. The method estimates the depth of the crust-mantle interface (the Moho) based on observed gravity data using a spherical Earth approximation. We extended the computationally efficient Bott's method to include smoothness regularization and use tesseroids instead right rectangular prisms. The inversion is controlled by three hyper-parameters: the regularization parameter, the density-contrast between the real Earth and the reference model (the Normal Earth), and the depth of the Moho of the Normal Earth. We

employ two cross-validation procedures to automatically estimate these parameters. Tests on synthetic data confirm the capability of the proposed method to estimate smoothly varying Moho depths and the three hyper-parameters. Finally, we applied the inversion method developed to produce a Moho depth model for South America. The estimated Moho depth model fits the gravity data and seismological Moho depth estimates in the oceanic areas and the central and eastern portions of the continent. We observe large misfits in the Andes region, where Moho depth is largest. In Amazon, Solimões, and Paraná Basins, the model fits the observed gravity but disagrees with seismological estimates. These discrepancies suggest the existence of density-anomalies in the crust or upper mantle, as has been suggested in the literature.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Gravity measurements are one of the instruments that geophysicists use to investigate the subsurface of the Earth. Gravity data can be acquired on the ground, airborne, shipborne or through artificial satellites. Ground, airborne, and shipborne data are routinely used in local or regional studies, with applications ranging from archaeological studies (e.g., PANISOVA *et al.*, 2013) to mapping the depth of sedimentary basins (e.g., GORDON *et al.*, 2013). Satellite gravity data make continental and global scale studies possible (e.g., BOUMAN *et al.*, 2013b; BRAITENBERG, 2015; REGUZZONI *et al.*, 2013; VAN DER MEIJDE *et al.*, 2013, 2015). This is particularly important in regions where data acquisition by other means is lacking or difficult to perform, such as South America, Africa, and Antarctica. Another advantage of satellite gravity measurements is the almost homogeneous spacial coverage. Satellite data also enable investigation of temporal variations of the Earth's gravity field through the GRACE mission. Applications using the time series data from GRACE include mapping ice-mass variation in the Arctic (CHEN *et al.*, 2011) and Antarctic regions (RAMILLIEN *et al.*, 2006), quantifying deformation following large earthquakes (MIKHAILOV *et al.*, 2014), and groundwater monitoring (HUMPHREY *et al.*, 2016).

Deriving geophysical Earth models from observed gravity data is an ill-posed inverse problem (BACKUS e GILBERT, 1967, 1968). Designing a method for solving these inverse problems presents many challenges. The first challenge is to establish a functional mapping between the model parameters and the data. This is known as the forward problem and it must be solved in a stable and accurate way for the inversion to succeed. The second challenge is to choose and implement an optimization algorithm to estimate the model parameters that best fit the observed data. There are several well established optimization methods to choose from the literature, for example, gradient-descent methods like the Gauss-Newton method and Steepest Descent or stochastic methods like the Genetic Algorithm (e.g., ASTER *et al.*, 2012; KELLEY, 1987; MENKE, 1984). Finally, there is the challenge of

stabilizing the ill-posed inverse problem, usually through Tikhonov regularization (TIKHONOV e ARSENIN, 1977). Different regularizing functions favor different model attributes, such as smoothness or compactness (e.g., SILVA *et al.*, 2001). Choosing an appropriate regularizing function is an indirect way to include prior geological or geophysical information in the inverse problem.

When developing a new inversion methodology, one must implement in a software application three modules: the forward problem, an optimization algorithm, and a regularizing function. Fortunately, these modules can usually be uncoupled. For example, the implementation of the forward problem does not depend on the choice of optimization method or regularizing function. Likewise, the software implementation of an optimization algorithm requires only a knowledge of a function to be optimized (and possibly its derivatives), no matter what is the forward problem or regularizing function. Furthermore, changing the regularizing function used, in principle, does not require changes to the implementations of the forward problem and the optimization method. Thus, the ideal software design is to have independent and reusable routines for forward modeling, optimization, and regularization. These three modules can be combined in different ways to produce new inversion software.

Here, we develop two software projects and apply their reusable modules to develop a 3D gravity inversion method in spherical coordinates. Chapter 2 describes the open-source software *Tesseroids*. This C language program calculates the gravitational potential and its first and second derivatives of a tesseroid (or spherical prism). The software also improves upon existing algorithms for the forward modeling calculations. Chapter 3 describes the Python language library *Fatiando a Terra*. The library contains a collection of functions and classes for inverse problems, forward modeling, data and model visualization, and data processing. The inverse problems tools implement optimization and regularization classes that are uncoupled from specific forward problems. These tools can be reused and combined in different ways to implement new inversion methods. *Fatiando a Terra* also implements the tesseroid forward modeling algorithm described in Chapter 2. Finally, in Chapter 4 we build upon the foundation of Chapters 2 and 3 to develop a novel gravity inversion method. The method estimates the depth of the crust-mantle interface (the Moho) from observed gravity data using a spherical Earth approximation. The software implementation of the inversion combines and extends the optimization, regularization, and forward modeling available in *Fatiando a Terra*. We apply our method to estimate the depth of the Moho for the South American continent.

Chapter 2

Tesseroids: forward modeling gravitational fields in spherical coordinates

This chapter has been submitted for publication in the "Geophysical Software and Algorithms" section of the journal GEOPHYSICS.

2.1 Abstract

We present the open-source software *Tesseroids*, a set of command-line programs to perform the forward modeling of gravitational fields in spherical coordinates. The software is implemented in the C programming language and uses tesseroids (spherical prisms) for the discretization of the subsurface mass distribution. The gravitational fields of tesseroids are calculated numerically using the Gauss-Legendre Quadrature (GLQ). We have improved upon an adaptive discretization algorithm to guarantee the accuracy of the GLQ integration. Our implementation of adaptive discretization uses a "stack" based algorithm instead of recursion to achieve more control over execution errors and corner cases. The algorithm is controlled by a scalar value called the distance-size ratio (D) that determines the accuracy of the integration as well as the computation time. We determined optimal values of D for the gravitational potential, gravitational acceleration, and gravity gradient tensor by comparing the computed tesseroids effects with those of a homogeneous spherical shell. The values required for a maximum relative error of 0.1% of the shell effects are $D = 1$ for the gravitational potential, $D = 1.5$ for the gravitational acceleration, and $D = 8$ for the gravity gradients. Contrary to previous assumptions, our results show that the potential and its first and second derivatives require different values of D to achieve the same accuracy. These values were incorporated as defaults in

the software.

2.2 Introduction

Satellite missions dedicated to measuring the Earth's gravity field (like CHAMP, GRACE, and GOCE) have provided geophysicists with almost uniform and global data coverage. These new data have enabled interpretations on regional and global scales (e.g. BRAITENBERG, 2015; REGUZZONI *et al.*, 2013). Modeling at such scales requires taking into account the curvature of the Earth and calculating gravity gradients as well as the traditional gravitational acceleration. A common approach to achieve this is to discretize the Earth into tesseroids (Figure 2.1) instead of rectangular prisms. An analytical solution exists when the computation point is along the polar axis and the tesseroid is extended into a spherical cap (GROMBEIN *et al.*, 2013; LAFEHR, 1991; MIKUŠKA *et al.*, 2006). For more general cases, the integral formula for the gravitational effects of a tesseroid must be solved numerically. Approaches to this numerical integration include Taylor series expansion (GROMBEIN *et al.*, 2013; HECK e SEITZ, 2007) and the Gauss-Legendre Quadrature (ASGHARZADEH *et al.*, 2007). Taylor series expansion produces accurate results at low latitudes but presents a decrease in accuracy towards the polar regions. This is attributed to tesseroids degenerating into an approximately triangular shape at the poles. The Gauss-Legendre Quadrature (GLQ) integration consists in approximating the volume integral by a weighted sum of the effect of point masses. An advantage of the GLQ approach is that it can be controlled by the number of point masses used. The larger the number of point masses, the better the accuracy of GLQ integration. A disadvantage is the increased computation time as the number of point masses increases. Thus, there is a trade-off between accuracy and computation time. This is a common theme in numerical methods. WILD-PFEIFFER (2008) investigated the use of different mass elements, including tesseroids, to compute the gravitational effects of topographic masses. The author concludes that using tesseroids with GLQ integration gives the best results for near-zone computations. However, the question of how to determine the optimal parameters for GLQ integration remained open.

Previous work by KU (1977) investigated the use of the GLQ in gravity forward modeling. KU (1977) numerically integrated the vertical component of the gravitational acceleration of right rectangular prisms. The author suggested that the accuracy of the GLQ integration depends on the ratio between distance to the computation point and the distance between adjacent point masses. Based on this, KU (1977) proposed an empirical criterion that the distance between point masses should be greater than the distance to the computation point. ASGHARZADEH

et al. (2007) used this criterion for the GLQ integration of the gravity gradient tensor of tesseroids. To our knowledge, an analysis of how well this ad hoc criteria of KU (1977) works for gravity gradient components or for tesseroids has never been done before. There has also been no attempt to quantify the error committed in the GLQ integration when applying the criteria of KU (1977).

LI *et al.* (2011) devised an algorithm to automatically enforce the criteria of KU (1977). Their algorithm divides the tesseroid into smaller ones instead of increasing the number of point masses per tesseroid. A tesseroid is divided if the minimum distance to the computation point is smaller than the largest dimension of the tesseroid. This division is repeated recursively until all tesseroids obey the criterion. Then, GLQ integration is performed for each of the smaller tesseroids using the specified number of point masses. The advantage of this adaptive discretization over increasing the number of points masses is that the total distribution of point masses will be greater only close to the computation point. This makes the adaptive discretization more computationally efficient.

GROMBEIN *et al.* (2013) developed optimized formula for the gravitational fields of tesseroids using Cartesian integral kernels. These formulas are faster to compute and do not have singularities at the poles like their spherical counterparts. The Cartesian formulae are numerically integrated using a Taylor series expansion as per HECK e SEITZ (2007). GROMBEIN *et al.* (2013) use a near-zone separation to mitigate the increased error at high latitudes. In the so called "near-zone" of the computation point they use a finer discretization composed by smaller tesseroids. This is accomplished by dividing the tesseroids along their horizontal dimensions. However, the determination of an optimal size of the near-zone remains an open question (GROMBEIN *et al.*, 2013).

We have implemented a modified version of the adaptive discretion of LI *et al.* (2011) into the open-source software package *Tesseroids*. The software uses the Cartesian formula of GROMBEIN *et al.* (2013) for improved performance and robustness. Previous versions of the software have been used by, e.g., ÁLVAREZ *et al.* (2012); BOUMAN *et al.* (2013a,b); BRAITENBERG (2015); BRAITENBERG *et al.* (2011); FULLEA *et al.* (2015); MARIANI *et al.* (2013).

This article describes the software design and the implementation of our modified adaptive discretization algorithm. We also present a numerical investigation of the error committed in the computations. These results allow us to calibrate the adaptive discretization algorithm separately for the gravitational potential, gravitational acceleration, as well as the gravity gradient tensor components.

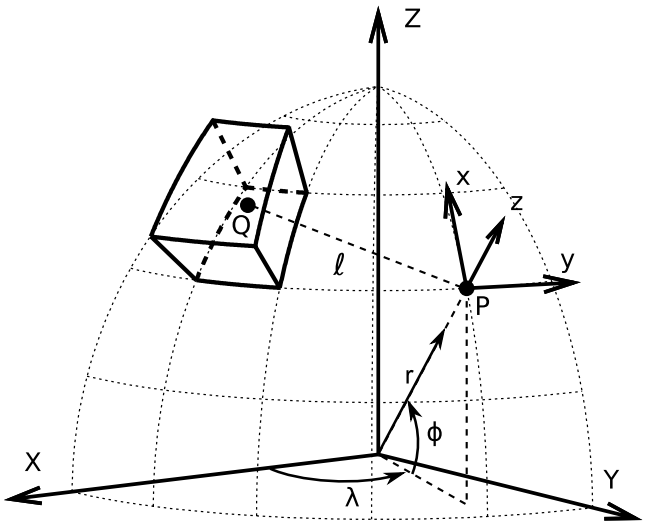

Figure 2.1: View of a tesseroid, the integration point Q inside the tesseroid, a geocentric coordinate system (X, Y, Z), the computation P and it's local coordinate system (x, y, z). r, ϕ, λ are the radius, latitude, and longitude, respectively, of point P, and ℓ is the Cartesian distance between P and Q.

2.3 Theory

A tesseroid is a mass element defined in geocentric spherical coordinates (Figure 2.1). It is bounded by two meridians, two parallels, and two concentric circles. The gravitational fields of a tesseroid at a point $P = (r, \phi, \lambda)$ are determined with respect to the local North-oriented coordinate system at P (x, y, z in Figure 2.1). GROMBEIN *et al.* (2013) formulated Cartesian kernels for the volume integrals that define the tesseroid gravitational potential, gravitational acceleration, and Marussi tensor, respectively,

$$V(r, \phi, \lambda) = G\rho \int_{\lambda_1}^{\lambda_2} \int_{\phi_1}^{\phi_2} \int_{r_1}^{r_2} \frac{1}{\ell} \kappa \; dr' d\phi' d\lambda', \tag{2.1}$$

$$g_\alpha(r, \phi, \lambda) = G\rho \int_{\lambda_1}^{\lambda_2} \int_{\phi_1}^{\phi_2} \int_{r_1}^{r_2} \frac{\Delta_\alpha}{\ell^3} \kappa \; dr' d\phi' d\lambda', \tag{2.2}$$

and

$$g_{\alpha\beta}(r, \phi, \lambda) = G\rho \int_{\lambda_1}^{\lambda_2} \int_{\phi_1}^{\phi_2} \int_{r_1}^{r_2} I_{\alpha\beta} \; \kappa \; dr' d\phi' d\lambda', \tag{2.3}$$

$$I_{\alpha\beta} = \left(\frac{3\Delta_\alpha \Delta_\beta}{\ell^5} - \frac{\delta_{\alpha\beta}}{\ell^3} \right), \tag{2.4}$$

where $\alpha, \beta \in \{x, y, z\}$, ρ is the density, $G = 6.674 \times 10^{-11} \; m^3 kg^{-1} s^{-1}$ is the gravitational constant, $\delta_{\alpha\beta}$ is Kronecker's delta ($\delta_{\alpha\beta} = 1$ if $\alpha = \beta$ and $\delta_{\alpha\beta} = 0$ if $\alpha \neq \beta$),

and

$$\Delta_x = r'(\cos\phi \sin\phi' - \sin\phi \cos\phi' \cos(\lambda' - \lambda)), \tag{2.5}$$

$$\Delta_y = r' \cos\phi' \sin(\lambda' - \lambda), \tag{2.6}$$

$$\Delta_z = r' \cos\psi - r, \tag{2.7}$$

$$\kappa = r'^2 \cos\phi', \tag{2.8}$$

$$\ell = \sqrt{r'^2 + r^2 - 2r'r\cos\psi}, \tag{2.9}$$

$$\cos\psi = \sin\phi \sin\phi' + \cos\phi \cos\phi' \cos(\lambda' - \lambda). \tag{2.10}$$

We will follow ASGHARZADEH *et al.* (2007) and perform the numerical integration using the Gauss-Legendre Quadrature (GLQ). The GLQ consists in approximating the integral by a weighted sum of the integration kernel (HILDEBRAND, 1987),

$$\int_a^b f(x)dx \approx \frac{b-a}{2} \sum_{i=1}^{N} W_i f(x_i), \tag{2.11}$$

in which N is the order of the quadrature, i.e. the number of points used in the GLQ. The points x_i are called the quadrature nodes. They are the roots of the N^{th} order Legendre polynomial $P_N(x)$. For a second order polynomial ($P_2(x)$), the roots are $x = \pm 0.577350269$. Roots for larger order polynomials can be determined by a root finder algorithm. Roots of Legendre polynomials will be within the range $[-1, 1]$. Before being used for GLQ integration, the roots must be scaled to the integration limits $[a, b]$ using

$$x_i^{scaled} = \frac{b-a}{2} x_i + \frac{b+a}{2}. \tag{2.12}$$

The weights of the GLQ are given by (HILDEBRAND, 1987),

$$W_i = \frac{2}{(1 - x_i^2)(P'_N(x_i))^2}. \tag{2.13}$$

The values of $P_N(x)$ and its first derivative $P'_N(x)$ can be calculated with recursive relations.

The Gauss-Legendre Quadrature for three-dimensional volume integrals, like equations 2.1-2.3, becomes (ASGHARZADEH *et al.*, 2007)

$$\iiint_\Omega f(r', \lambda', \phi')d\Omega \approx A \sum_{i=1}^{N^r} \sum_{j=1}^{N^\phi} \sum_{k=1}^{N^\lambda} W_i^r W_j^\phi W_k^\lambda f(r_i, \phi_j, \lambda_k), \tag{2.14}$$

where

$$A = \frac{(\lambda_2 - \lambda_1)(\phi_2 - \phi_1)(r_2 - r_1)}{8}. \tag{2.15}$$

Comparing equation 2.14 with equations 2.1-2.3, we see that $f(r_i, \phi_j, \lambda_k)$ is the effect of a point mass located on the quadrature nodes. Thus, it can be said that the GLQ integration approximates the volume integrals by a weighted sum of point mass effects.

The accuracy of the integration depends on the number of point masses used in the summation. KU (1977) showed that it also depends on the ratio between the distance to the computation point and the distance between adjacent nodes. Figure 2.2 illustrates this effect on the g_{xy} gravity gradient component. The g_{xy} component was produced by a $7^\circ \times 7^\circ \times 20\ km$ tesseroid with 2.67 $g.cm^{-3}$ density and top at $z = 0\ km$. The maps were calculated on a regular grid with 100×100 points. Figure 2.2a shows the g_{xy} component calculated at 400 km height using GLQ with order two ($2 \times 2 \times 2 = 8$ point masses). Figure 2.2b shows g_{xy} computed with order two GLQ as well but at 150 km height. Notice that the computed effect is concentrated around each point mass of the GLQ (black dots) and does not resemble the effect of a tesseroid. KU (1977) determined an ad hoc criterion that the distance between point masses (quadrature nodes) should be smaller than the minimum distance to the computation point. Thus, if a computation point is too close to the tesseroid one would have to decrease the distance between the point masses in order to obtain an accurate result. One way to accomplish this would be increase the order of the quadrature N in all three directions. Figure 2.2c shows the g_{xy} component calculated at 150km height but with a GLQ order of 30 ($30 \times 30 \times 30 = 27,000$ point masses). The computed g_{xy} component more closely resembles the expect results for a single tesseroid (ASGHARZADEH *et al.*, 2007).

2.3.1 Adaptive discretization

LI *et al.* (2011) proposed an alternative method for decreasing the distance between point masses on the quadrature nodes aiming at achieving an accurate integration. Instead of increasing the GLQ order, they keep it fixed to a given number and divide the tesseroid into smaller volumes. The sum of the effects of the smaller tesseroids is equal to the gravitational effect of the larger tesseroid. This division effectively decreases the distance between nodes because of the smaller size of the tesseroids. The criterion for dividing a tesseroid is that the distance to the computation point should be smaller than a constant times the size of the tesseroid. This is analogous to the criterion proposed by KU (1977) because the size of the tesseroid serves as a proxy for the distance between point masses. This procedure is repeated recursively

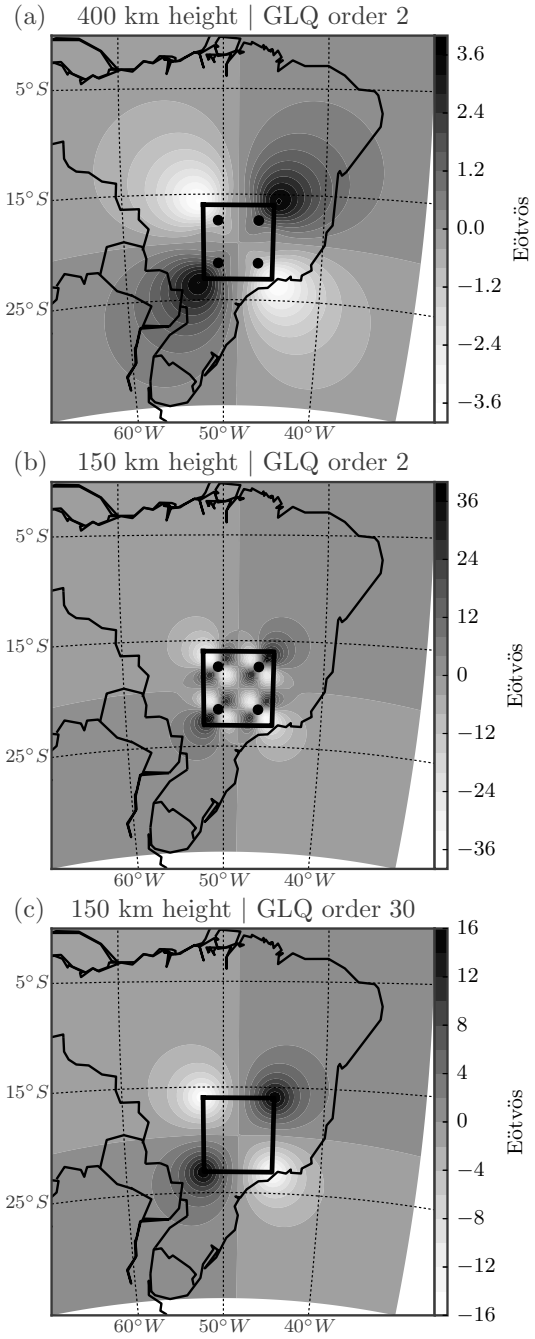

Figure 2.2: Example of the effect of varying the computation height and the number of point masses in the Gauss-Legendre Quadrature. Black circles represent the horizontal location of the point masses. a) g_{xy} calculated at 400 km height using GLQ order 2 ($2 \times 2 \times 2 = 8$ point masses). b) At 150 km height and GLQ order 2, the result resembles that of four point masses instead of a single tesseroid. This effect was shown by KU (1977). c) At 150 km but with a higher GLQ order of 30. In (c) the horizontal locations of the point masses were not shown. Notice that the results shown in (c) are similar to that expected for a single mass source.

until all tesseroids are within the acceptable ratio of distance and size or a minimum size is achieved.

The advantage of this adaptive discretization is that the number of point masses is only increased in parts of the tesseroid that are closer to the computation point. Notice that the alternative approach of simply increasing the order of the GLQ would increase the number of point masses evenly throughout the whole tesseroid.

2.4 Implementation

We have implemented the calculation of the tesseroid gravitational fields with adaptive discretization in version 1.2 of the open-source package *Tesseroids*. It is freely available online (`http://tesseroids.leouieda.com` or `http://dx.doi.org/10.5281/zenodo.16033`) under the BSD 3-clause open-source license. An archived version of the source code is also available as part of this article.

Tesseroids consists of command-line programs written in the C programming language. The package includes programs to calculate the gravitational fields of tesseroids and rectangular prisms (in both Cartesian and spherical coordinates). All programs receive input through command-line arguments and the standard input channel ("STDIN") and output the results through the standard output channel ("STDOUT"). For example, the command to generate a regular grid with $NLON \times NLAT$ points, calculate g_z and g_{zz} caused by the tesseroids in a file "MODELFILE", and save the results to a file called "OUTPUT" is:

```
tessgrd -rW/E/S/N -bNLON/NLAT -zHEIGHT | \
    tessgz MODELFILE | \
    tessgzz MODELFILE > OUTPUT
```

The *src* folder of the source code archive contains the C files that build the command-line programs (e.g., *tessgz.c*). The *src/lib* folder contains the source files that implement the numerical computations. We will not describe here the implementation of the input/output parsing and other miscellanea. Instead, we will focus on the details of the Gauss-Legendre Quadrature integration of equations 2.1-2.3 and the adaptive discretization of tesseroids.

2.4.1 Numerical integration

The source file *src/lib/glq.c* contains the code necessary to perform a Gauss-Legendre Quadrature integration. The first step in the GLQ is to compute the locations of the discretization points (i.e., the point masses). These points are roots of Legendre polynomials. Precomputed values are available for low order polynomials,

typically up to order five. For flexibility and to compute higher order roots, we use the multiple root-finder algorithm of BARRERA-FIGUEROA *et al.* (2006). The additional computational load is minimal because the root-finder algorithm must be run only once per program execution. The root-finder is implemented in functions *glq_nodes* and *glq_next_root*. The computed roots will be in the range $[-1, 1]$ and must be scaled to the integration limits (the physical boundaries of the tesseroid) using function *glq_set_limits* (see equation 2.12).

The GLQ weights (equation 2.13) are computed by function *glq_weights*. Both the computed roots and weights are stored in a data structure (a C *struct*) called *GLQ*. Function *glq_new* handles memory allocation, calculates the roots and weights, and returns the complete *GLQ* structure.

The numerical integration of the tesseroid gravitational fields is performed by the functions in module *src/lib/grav_tess.c*. Functions *tess_pot*, *tess_gx*, *tess_gy*, and so on, compute the gravitational fields of a single tesseroid on a single computation point. These functions require three *GLQ* structures, each containing the roots and weights for GLQ integration in the three dimensions. The roots must be scaled to the integration limits $[\lambda_1, \lambda_2], [\phi_1, \phi_2], [r_1, r_2]$ (see equations 2.1-2.3). The integration consists of three loops that sum the weighted kernel functions evaluated at each GLQ point mass (the scaled roots).

The biggest bottlenecks for the numerical integration are the number of point masses used and the evaluation of the trigonometric functions in equations 2.1-2.3 inside the inner loops. Better performance is achieved by pre-computing the sine and cosine of latitudes and moving some trigonometric function evaluations to the outer loops.

2.4.2 Implementation of adaptive discretization

Our implementation of the adaptive discretization algorithm differs in a few ways from the one proposed by LI *et al.* (2011). In LI *et al.* (2011), a tesseroid will be divided when the smallest distance between it and the computation point is smaller than a constant times the largest dimension of the tesseroid. Instead of the smallest distance, we use the easier to calculate distance between the computation point (r, λ, ϕ) and the geometric center of the tesseroid (r_t, λ_t, ϕ_t)

$$d = \left[r^2 + r_t^2 - 2rr_t \cos\psi_t\right]^{\frac{1}{2}}, \tag{2.16}$$

$$\cos\psi_t = \sin\phi \sin\phi_t + \cos\phi \cos\phi_t \cos(\lambda - \lambda_t). \tag{2.17}$$

Our definition of the dimensions of the tesseroid (the "side lengths" of LI *et al.* (2011)) along longitude, latitude, and radius, respectively, are (Figure 2.3a)

$$L_\lambda = r_2 \arccos(\sin^2 \phi_t + \cos^2 \phi_t \cos(\lambda_2 - \lambda_1)), \tag{2.18}$$

$$L_\phi = r_2 \arccos(\sin \phi_2 \sin \phi_1 + \cos \phi_2 \cos \phi_1), \tag{2.19}$$

$$L_r = r_2 - r_1. \tag{2.20}$$

L_λ and L_ϕ are arc-distances measured along the top surface of the tesseroid (Figure 2.3a). Specifically, L_λ is measured long the middle latitude of the tesseroid (ϕ_t).

To determine if a tesseroid must be divided, we check if

$$\frac{d}{L_i} \geq D, \tag{2.21}$$

for each $i \in (\lambda, \phi, r)$. D is a positive scalar hereafter referred to as the "distance-size ratio". If the inequality holds for all three dimensions, the tesseroid is not divided. Thus, the distance-size ratio determines how close the computation point can be before we must divide the tesseroid. The value of D is indirectly responsible for the accuracy of the solution and the computation time. We will explore the relationship with the accuracy in the following section.

Figure 2.3 shows examples of the resulting tesseroid models after adaptive discretization. Figure 2.3a shows the initial tesseroid and computation point P. Figures 2.3b-d are the result of adaptive discretization using different values of the distance-size ratio D, respectively, $D = 1$, $D = 2$, and $D = 6$. The number of tesseroids in the resulting discretization is, respectively, 4, 38, and 936.

Instead of using recursive function calls, as originally proposed by LI *et al.* (2011), we use a stack-based implementation of the algorithm. Stacks are array-like data structures with a particular way of inserting and removing elements from it. In a stack, one can only insert elements to the top of the stack (the last empty position). Likewise, one can only remove the last element of the stack (commonly referred to as "popping" the stack). Because of these restrictions, stacks are also known as "Last-In-First-Out" (LIFO) data structures.

The discretization algorithm is implemented in function *calc_tess_model_adapt* of the file *src/lib/grav_tess.c*. This function calculates the effect of a single tesseroid on a single computation point. The stack of tesseroids is represented by the *stack* variable, an array of *TESSEROID* structures. We must define a maximum size for the stack to allocate memory for it. Defining a maximum size allows us to avoid an infinite loop in case the computation point is on (or sufficiently close to) the surface of the tesseroid. We use the integer *stktop* to keep track of the index of the last element in the stack (the top of the stack).

Below, we describe the algorithm to calculate the effect of a single tesseroid from

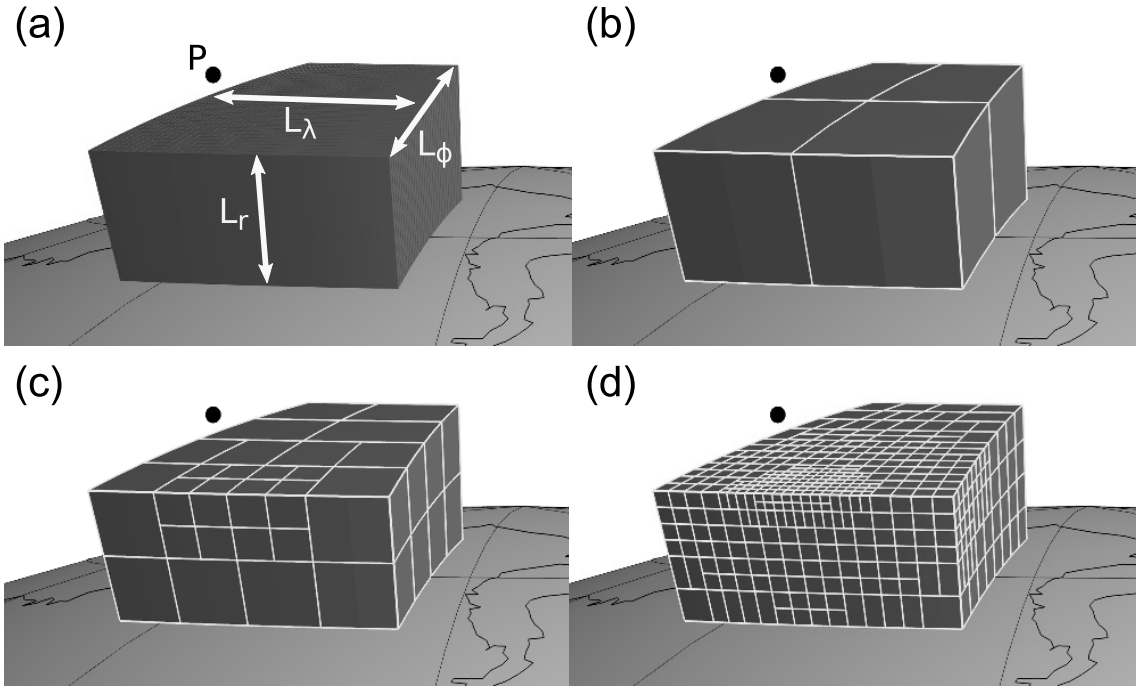

Figure 2.3: Adaptive discretization of the tesseroid shown in (a) for a computation point P using the distance-size ratio D equal to (b) 1, (c) 2, and (d) 6. L_r, L_ϕ, and L_λ are the dimensions of the tesseroid. Note that increasing D results in a fine division of the tesseroid close the computation point and a coarser division further away.

the input model on a single computation point. The algorithm starts by creating an empty stack of tesseroids. Then, the stack is initialized with the single input tesseroid. The initialization is done by copying the tesseroid into the stack and setting *stktop* to zero (the first element). It is important to note that the stack is not the input tesseroid model. Instead, it is a buffer used to temporarily store each stage of the discretization algorithm.

Once the stack is initialized, the steps of the algorithm are:

1. "Pop" the stack (i.e., take the last tesseroid from it). This will cause *stktop* to be reduced by one. This tesseroid is the one that will be evaluated in the following steps.

2. Compute the distance d (equation 2.16) between the geometric center of the tesseroid and the computation point.

3. Compute the dimensions of the tesseroid L_λ, L_ϕ, and L_r using equations 2.18-2.20.

4. Check the condition in equation 2.21 for each dimension of the tesseroid.

5. If all dimensions hold the inequality 2.21, the tesseroid is not divided and its gravitational effect is computed using the Gauss-Legendre Quadrature (equa-

tions 2.1-2.3 and 2.14). We use a GLQ order of two for all three dimensions ($2 \times 2 \times 2 = 8$ point masses) by default. This value can be changed using a command-line argument of the modeling programs.

6. If any of the dimensions fail the condition:

   (a) Divide the tesseroid in half along each dimension that failed the condition.

   (b) Check if there is room in the stack for the new tesseroids (i.e.,the number of new elements plus *stktop* is smaller than the maximum stack size). If there isn't, warn the user of a "stack overflow" and compute the effect of the tesseroid, as in step 5. If there is room in the stack, place the smaller tesseroids into the stack.

7. Repeat the above steps until the stack is empty (*stktop* is equal to -1).

The algorithm above is repeated for every tesseroid of the input model and the results are summed. This will yield the gravitational effect of the input tesseroid model on a single point. Thus, the computations must be repeated for every computation point. The whole algorithm can be summarized in the following pseudo-code.

```
Initialize the output array with zeros.
for tesseroid in model:
  for point in grid:
    Initialize the stack with tesseroid.
    stktop = 0
    while stktop >= 0:
      Perform steps 1-6 of the algorithm.
    Sum the calculated value to the output.
```

This stack-based implementation has some advantages over the original recursive implementation, namely: (1) It gives the developer more control over the recursion step. (2) In general, it is faster because it bypasses the overhead of function calls. In recursive implementations, the developer has no control over the maximum number of consecutive recursive calls (i.e., the "recursion depth"). This limit may vary with programming language, compiler, and operating system. Overflowing the maximum recursion depth may result in program crashes, typically with cryptic or inexistent error messages. In the stack-based implementation, the developer has complete control. Overflowing of the stack can be handled gracefully with an error message or even performing a suitable approximation of the result.

2.4.3 Code for figures and error analysis

The error analysis and all figures in this article were produced in IPython notebooks (PÉREZ e GRANGER, 2007). The notebook files combine source code in various programming languages, program execution, text, equations, and the figures generated by the code into a single document. We used the following Python language libraries to perform the error analysis and generate figures: *pandas* by MCKINNEY (2010), *matplotlib* by HUNTER (2007) for 2D figures and maps, and *Mayavi* by RAMACHANDRAN e VAROQUAUX (2011) for 3D figures.

The IPython notebooks and the data generated for the error analysis, as well as instructions for installing the software and running the programs, are also included in the source code archive that accompanies this article. Alternatively, all accompanying material is available in the online repository `https://github.com/pinga-lab/paper-tesseroids`.

2.5 Evaluation of the accuracy

The key controlling point of the adaptive discretization algorithm is the distance-size ratio D (equation 2.21). The specific value chosen for D determines how many divisions will be made (Figure 2.3). Thus, D indirectly controls both the accuracy of the integration and the computation time. In this section, we investigate the relationship between the distance-size ratio and the integration error. We perform the analysis for the gravitational potential, acceleration, and gradient tensor components to evaluate if the same value of D yields compatible error levels for different fields.

The reference against which we compare the computed tesseroid fields is a homogeneous spherical shell. The shell has analytical solutions along the polar axis (GROMBEIN *et al.*, 2013; LAFEHR, 1991; MIKUŠKA *et al.*, 2006) and can be perfectly discretized into tesseroids. We chose a spherical shell with a thickness of 1 km, density of 2670 $kg.m^{-3}$, bottom at height 0 km above the reference sphere, and top at 1 km height. We produced tesseroid models of the shell by discretizing it along the horizontal dimensions into a regular mesh.

Figure 2.2 shows that the largest errors are spread over on top of the tesseroid. Thus, calculating the tesseroid fields at a single point might not capture the point of largest error. Instead, we calculate the effect of the tesseroid model on a regular grid of 10×10 points at different geographic locations (see Table 2.1). Fortunately, the symmetry of the shell allows us to consider the computation point at any geocentric coordinate. Therefore, the effect of the shell will be same along the entire grid. We compute the differences between the effects of the shell and the tesseroid model on

the grid. However, we will consider only the largest error in our analysis.

We placed the grid on top of a particular tesseroid to increase the chances of capturing the true largest integration error. We calculate the errors for values of the distance-size ratio D varying from 0 (i.e., no divisions) to 10 in 0.5 intervals. Furthermore, we repeated the error analysis in four different numerical experiments, each with computation grids at different locations and different tesseroid model sizes. Table 2.1 describes the different numerical experiments and the corresponding parameters of the computation grid and tesseroid model.

Figure 2.4 shows the maximum difference between the shell and tesseroid fields as a function of D for the four experiments. The differences are given as a percentage of the shell value. We established a maximum tolerated error of 0.1%, represented by the horizontal solid lines in Figure 2.4. Only results for the gravitational potential, g_z, and g_{zz} are shown. The results for the other diagonal components of the gravity gradient tensor are similar to g_{zz}. Figures for these components can be found in the supplementary material (see section "Reproducing the analysis and results").

For the potential V, a distance-size ratio $D = 1$ guarantees that the curves for all experiments are below the 0.1% error threshold. For g_z, the same is achieved with $D = 1.5$. Conversely, g_{zz} requires a value of $D = 8$ to achieve an error level of 0.1%. For a computation height of 260 km, the error curve for g_{zz} intercepts the error threshold line at $D = 2.5$. This behavior suggests that the error curves for g_{zz} might depend on the computation height. To test this hypothesis, we computed the error curves for g_{zz} at heights 2, 10, 50, 150, and 260 km. Figure 2.5 shows the results for g_{zz} at varying computation heights. Notice that the distance-size ratio required to achieve 0.1% accuracy decreases as the computation height increases. For example, computation at 260 km height requires $D = 2.5$ whereas at 10 km height a value of $D = 5.5$ is required to achieve the same accuracy. One can take advantage of this behavior to reduce the distance-size ratio for computations of the gravity gradient tensor at high altitudes, saving computation time.

We have implemented the values of the distance-size ratio producing 0.1% accuracy determined above as defaults for the software *Tesseroids*. We chose the conservative value of $D = 8$ for the gravity gradient components as a fail-safe alternative. Users can control the value of D used in the computations through command-line arguments to achieve greater performance at the cost of accuracy.

2.6 Conclusions

We have presented the open-source software *Tesseroids*. It consists of command-line programs, written in the C programming language, to perform the forward modeling of gravitational fields in spherical coordinates. The fields are calculated from a

	Grid location	Grid height	Tesseroid size
Experiment 1 (pole)	89N–90N/0E–1E	2 km	$1° \times 1°$
Experiment 2 (equator)	0N–1N/0E–1E	2 km	$1° \times 1°$
Experiment 3 (260 km)	89N–90N/0E–1E	260 km	$1° \times 1°$
Experiment 4 (30° size)	60N–90N/0E–30E	2 km	$30° \times 30°$

Table 2.1: Parameters of the numerical experiments to quantify the accuracy of the numerical integration. All grids had 10×10 regularly spaced computation points at a constant height. Tesseroids used to discretize the spherical shell had 1 km thickness and the horizontal dimensions shown in the table.

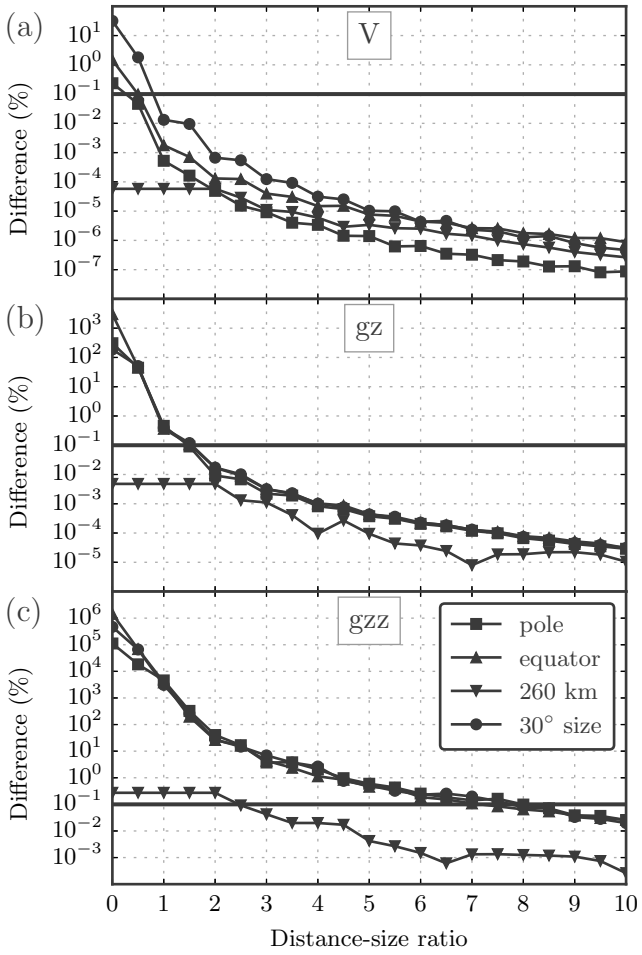

Figure 2.4: The maximum difference between the computed tesseroid and shell effects as a function of the distance-size ratio D for (a) the gravitational potential, (b) g_z, and (c) g_{zz}. The difference is given as a percentage of the shell effect. Curves correspond to the different tesseroid models and computation grids shown in Table 2.1. The horizontal solid black line marks the established error threshold of 0.1%. A value of $D = 0$ means that no divisions are made.

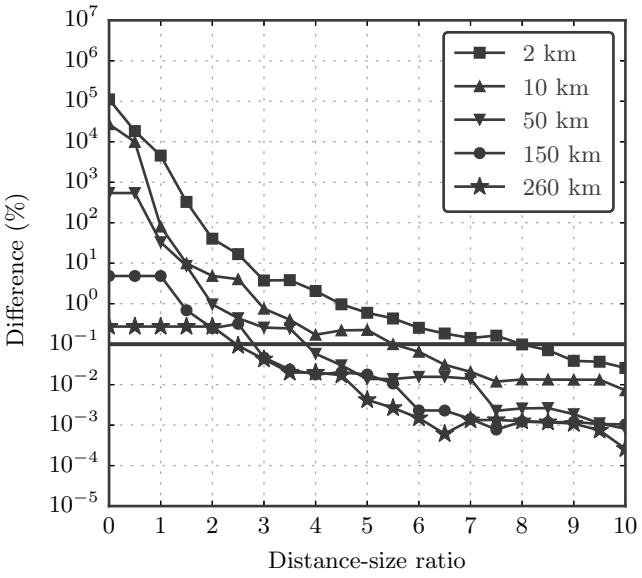

Figure 2.5: Difference between the computed g_{zz} for the spherical shell and the tesseroid model at different heights. Curves show the maximum difference as a percentage of the shell value. The horizontal solid black line marks the established error threshold of 0.1%. A value of $D = 0$ means that no divisions are made.

mass model composed of spherical prisms, the so-called tesseroids. The volume integrals of the gravitational fields of a tesseroid are solved numerically using the Gauss-Legendre Quadrature (GLQ). The GLQ approximates the volume integrals by weighted sums of point mass effects. The error of the GLQ integration increases as the computation point gets closer to the tesseroid. To counter this effect, the accuracy of the GLQ integration can be increased by using more point masses or by dividing each tesseroid into smaller ones.

We have implemented and improved upon an adaptive discretization algorithm to achieve an optimal division of tesseroids. Tesseroids are divided into more parts closer to the computation point, where more point masses are needed. Our implementation of the adaptive discretization uses a "stack" data structure in place of the originally proposed recursive implementation. As a rule of thumb in procedural languages (like C), stack-base implementations are computationally faster than the equivalent code using function recursion. Furthermore, the stack-based algorithm allows more control over errors when too many divisions are necessary. The adaptive discretization is controlled by a scalar called the distance-size ratio (D). The algorithm ensures that all tesseroids will have dimensions smaller than D times the distance to the computation point. The value of D indirectly controls the accuracy of the integration as well as the computation time.

We performed an error analysis to determine the optimal value of D required to achieve a target accuracy. We used a spherical shell as a reference to calculate the computation error of our algorithm for different values of D. Our results show that

the values of D required to achieve a maximum error of 0.1% of the shell values are 1 for the gravitational potential, 1.5 for the gravitational acceleration, and 8 for the gravity gradients. Previous assumptions in the literature were that accurate results are guaranteed if the distance to the tesseroid is larger than the distance between point masses. This condition was previously applied indiscriminately to both the gravitational acceleration and the gravity gradients. That assumption is equivalent to using $D = 1.5$ for all fields. Our results show that this is valid for the gravitational acceleration and results in a 0.1% computation error. This is expected because the original study that determined the above condition was performed on the vertical component of gravitational acceleration. However, applying the same condition to the gravity gradients produces an error of the order of 10^2%.

For the gravity gradients in particular, the distance-size ratio required for 0.1% error decreases with height. We believe this is because the decay factor for the gravity gradient components is d^{-3}, whereas the discretization algorithm uses d/L_i. As the computation point becomes closer to the tesseroid, the field increases more rapidly than the algorithm increases the amount of discretization. Hence, a higher value of D (i.e., more discretization) is required.

The values of the distance-size ratio determined above were incorporated as defaults in the software *Tesseroids*. We chose the value $D = 8$ for the gravity gradients as a conservative default. If the user desires, the value of D used can be controlled by a command-line argument.

In situations that require many tesseroid divisions, the stack used in the algorithm will overflow and further divisions become impossible. The current implementation warns the user that the overflow occurred and proceeds with the GLQ integration without division. Future improvements to the algorithm include a better way to handle such situations as they arise. An alternative would be to replace the tesseroid by an equivalent right rectangular prism and compute its effects instead. This would allow accurate computations at smaller distances. Furthermore, the computation time increases drastically as the computation point gets closer to the tesseroid. This effect can be prohibitive for computing the gravity gradients at relatively low heights (e.g., for terrain corrections of ground or airborne surveys). Further investigation of different criteria for dividing the tesseroids could yield better performance through a reduced number of divisions.

2.7 Online repository

The source code, compiled binary files, and user documentation for *Tesseroids* can be found at the project website `http://tesseroids.leouieda.com`. The source code for version 1.2 of *Tesseroids* is permanently archived at `http://dx.doi.org/`

10.5281/zenodo.16033. The data and source code that produced the results and figures presented here can be downloaded from https://github.com/pinga-lab/paper-tesseroids.

Chapter 3

Modeling the Earth with Fatiando a Terra

This chapter was published in the Proceedings of the 12th Python in Science Conference in 2013 (`http://conference.scipy.org/proceedings/scipy2013/uieda.html`). It describes the open-source software library *Fatiando a Terra* and the state of the project at the time of the conference. *Fatiando a Terra* was developed by me as part of my PhD thesis work. The library has grown since 2013 and now contains more features than described here. For example, the algorithm for gravitational forward modeling with tesseroids presented in Chapter 2 is also implemented in *Fatiando a Terra*. Furthermore, the gravity inversion method presented in Chapter 4 is implemented using functions and classes from *Fatiando a Terra* and will be included in a future release of the library. The official website (`http://www.fatiando.org`) contains more up-to-date information about the project.

3.1 Abstract

Geophysics is the science of using physical observations of the Earth to infer its inner structure. Generally, this is done with a variety of numerical modeling techniques and inverse problems. The development of new algorithms usually involves copy and pasting of code, which leads to errors and poor code reuse. *Fatiando a Terra* is a Python library that aims to automate common tasks and unify the modeling pipeline inside of the Python language. This allows users to replace the traditional shell scripting with more versatile and powerful Python scripting. The library can also be used as an API (Application Programming Interface) for developing stand-alone programs. Algorithms implemented in *Fatiando a Terra* can be combined to build upon existing functionality. This flexibility facilitates prototyping of new algorithms and quickly building interactive teaching exercises. In the future, we

plan to continuously implement sample problems to help teach geophysics as well as classic and state-of-the-art algorithms.

3.2 Introduction

Geophysics studies the physical processes of the Earth. Geophysicists make observations of physical phenomena and use them to infer the inner structure of the planet. This task requires the numerical modeling of physical processes. These numerical models can then be used in inverse problems to infer inner Earth structure from observations. Different geophysical methods use different kinds of observations. Geothermal methods use the temperature and heat flux of the Earth's crust. Potential field methods use gravitational and magnetic field measurements. Seismics and seismology use the ground motion caused by elastic waves from active (man-made) and passive (earthquakes) sources, respectively.

The seismic method is among the most widely studied due to the high industry demand. Thus, a range of well established open-source software have been developed for seismic processing. These include Seismic Un*x (STOCKWELL JR., 1999, `http://www.cwp.mines.edu/cwpcodes/`), Madagascar (MADAGASCAR DEVELOPMENT TEAM, 2013, `http://www.ahay.org/`), OpendTect (`http://opendtect.org`), and GêBR (`http://www.gebrproject.com`). A noteworthy open-source project that is not seismic related is the Generic Mapping Tools (GMT) project (WESSEL e SMITH, 1991, `http://gmt.soest.hawaii.edu/`). The GMT are a well established collection of command-line programs for plotting maps with a variety of different map projections. For geodynamic modeling there is the Computational Infrastructure for Geodynamics (`http://www.geodynamics.org`), which has grouped various well documented software packages. However, even with this wide range of well maintained software projects, many geophysical modeling software that are provided online still have no open-source license statement, have cryptic I/O files, are hard to integrate into a pipeline, and make code reuse and remixing challenging. Some of these problems are being worked on by the Solid Earth Teaching and Research Environment (SEATREE) (MILNER *et al.*, 2009, `http://geosys.usc.edu/projects/seatree/`) by providing a common graphical interface for previously existing software. The numerical computations are performed by the pre-existing underlying C/Fortran programs. Conversely, the SEATREE code (written in Python) handles the I/O and user interface. This makes the use of these tools easier and more approachable to students. However, the lack of a common Application Programming Interface (API) means that the code for these programs cannot be easily combined to create new modeling tools.

Fatiando a Terra (`http://www.fatiando.org`, see Figure 3.1) aims at provid-

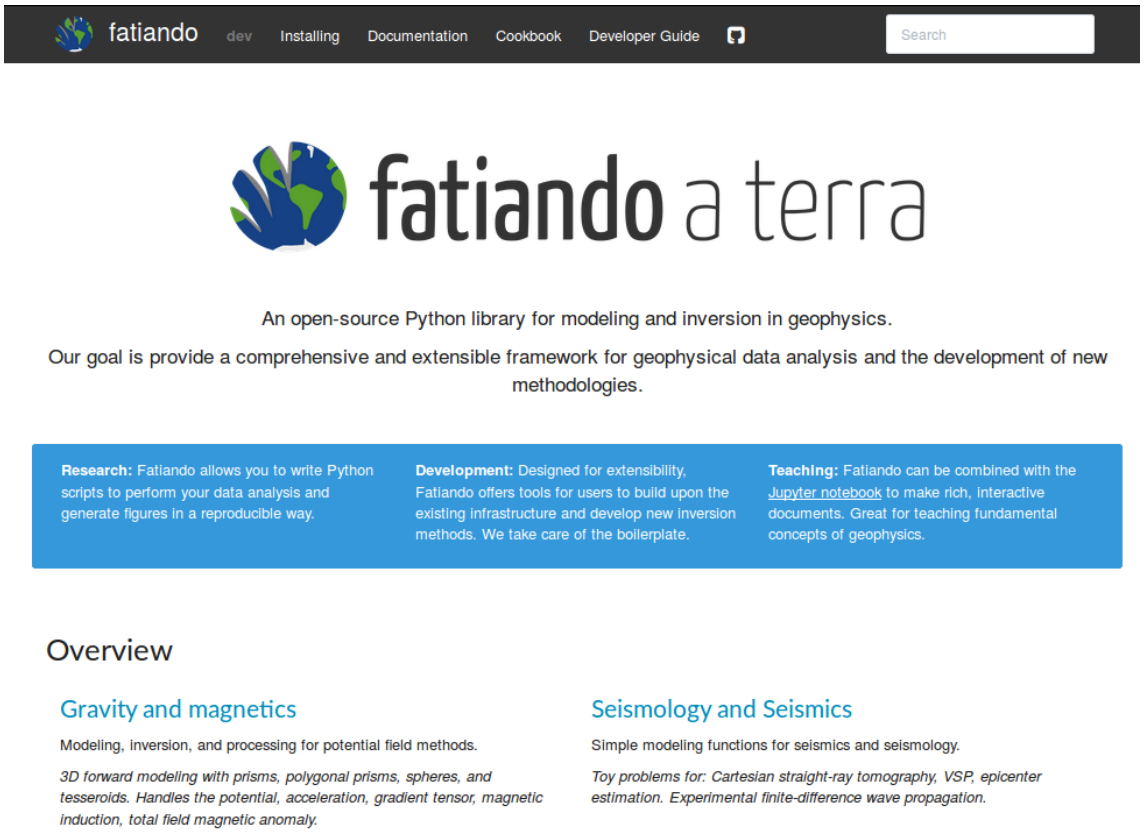

Figure 3.1: Screen capture of the `http://www.fatiando.org` website (accessed 30 of March 2016).

ing such an API for geophysical modeling. Functions in the `fatiando` package use compatible data and mesh formats so that the output of one modeling function can be used as input for another. Furthermore, routines can be combined and reused to create new modeling algorithms. *Fatiando a Terra* also automates common tasks such as griding, map plotting with Matplotlib (HUNTER, 2007, `http://matplotlib.org`), and 3D plotting with Mayavi (RAMACHANDRAN e VAROQUAUX, 2011, `http://code.enthought.com/projects/mayavi`). Version 0.1[1] of *Fatiando a Terra* is focused on gravity and magnetic methods because this is the main focus of the developers. However, simple "toy" problems for seismology and geothermics are available and can be useful for teaching geophysics.

The following sections illustrate the functionality and design of Fatiando a Terra using various code samples. An IPython (PÉREZ e GRANGER, 2007, `http://ipython.org/`) notebook file with these code samples is provided by UIEDA *et al.* (2013a) at `http://dx.doi.org/10.6084/m9.figshare.708390`.

[1] This was the current version in 2013 when this chapter was published in the conference proceedings. As of April 2016, the latest version is 0.3.

3.3 Package structure

The modules and packages of *Fatiando a Terra* are bundled into the `fatiando` package. Each type of geophysical method has its own package. As of version 0.1, the available modules and packages are:

- `fatiando.gravmag`: gravity and magnetic methods;
- `fatiando.seismic`: seismic methods and seismology;
- `fatiando.geothermal`: geothermal modeling;
- `fatiando.mesher`: geometric elements and meshes;
- `fatiando.gridder`: grid generation, slicing, interpolation, etc;
- `fatiando.io`: I/O of models and data sets from web repositories;
- `fatiando.utils`: miscellaneous utilities;
- `fatiando.constants`: physical constants;
- `fatiando.gui`: simple graphical user interfaces;
- `fatiando.vis`: 2D and 3D plotting;
- `fatiando.inversion`: inverse problem solvers and regularization;

3.4 Griding and map plotting

Fatiando a Terra handles map data as 1D Numpy arrays, typically x-, y-, z-coordinates and an extra array with the corresponding data. However, Matplotlib functions, like `contourf` and `pcolor`, require data to be passed as 2D arrays. Moreover, geophysical data sets are often irregularly sampled and require griding before they can be plotted. Thus, griding and array reshaping are ideal targets for automation.

The `fatiando.vis.mpl` module imports all the functions in `matplotlib.pyplot`, adds new functions, and overwrites others to automate repetitive tasks (such as griding). Thus, the basic functionality of the `pyplot` interface is maintained while customizations facilitate common tasks. The following example illustrates the use of the custom `fatiando.vis.mpl.contourf` function to automatically grid and plot some irregularly sampled data (Figure 3.2):

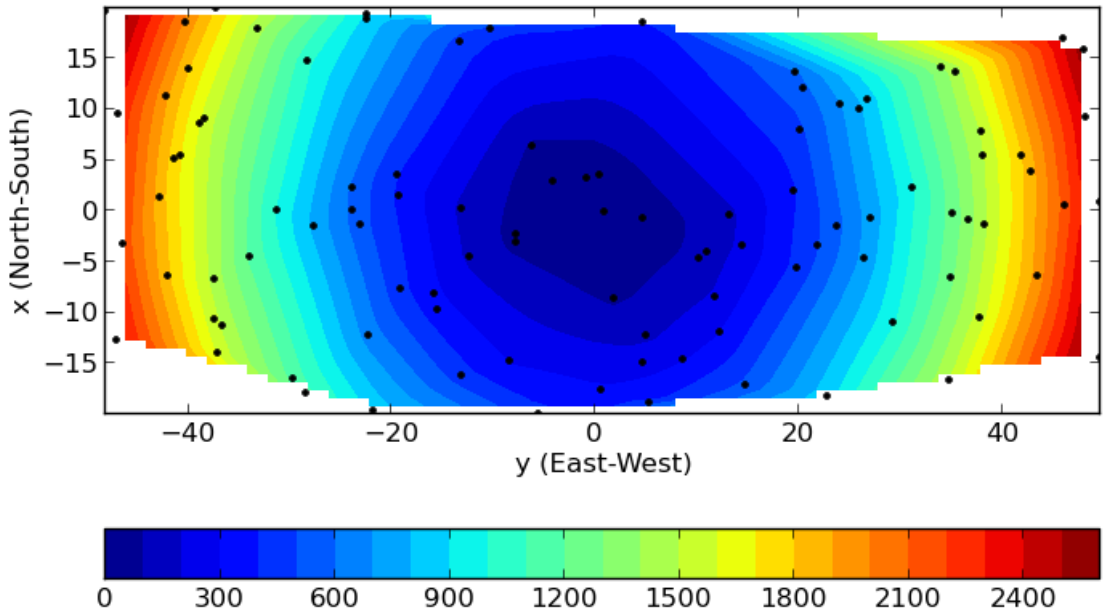

Figure 3.2: Example of 1) generating a random scatter of points (black dots), 2) using that to make synthetic data, and 3) automatically gridding and plotting the data using a *Fatiando a Terra* wrapper for the Matplotlib "contourf" function.

```
from fatiando import gridder
from fatiando.vis import mpl
area = [-20, 20, -50, 50]
x, y = gridder.scatter(area, n=100)
data = x**2 + y**2
mpl.figure()
mpl.axis('scaled')
mpl.contourf(y, x, data, shape=(50, 50),
    levels=30, interp=True)
mpl.colorbar(orientation='horizontal')
mpl.plot(y, x, '.k')
mpl.xlabel('y (East-West)')
mpl.ylabel('x (North-South)')
mpl.show()
```

Notice that, in the calls to `mpl.contourf` and `mpl.plot`, the x- and y-axis are switched. That is because it is common practice in geophysics for x to point North and y to point East.

Map projections in Matplotlib are handled by the Basemap toolkit (`http://matplotlib.org/basemap`). The `fatiando.vis.mpl` module also provides helper functions to automate the use of this toolkit. The `fatiando.vis.mpl.basemap` function automates the creation of the `Basemap` objects with common parameters.

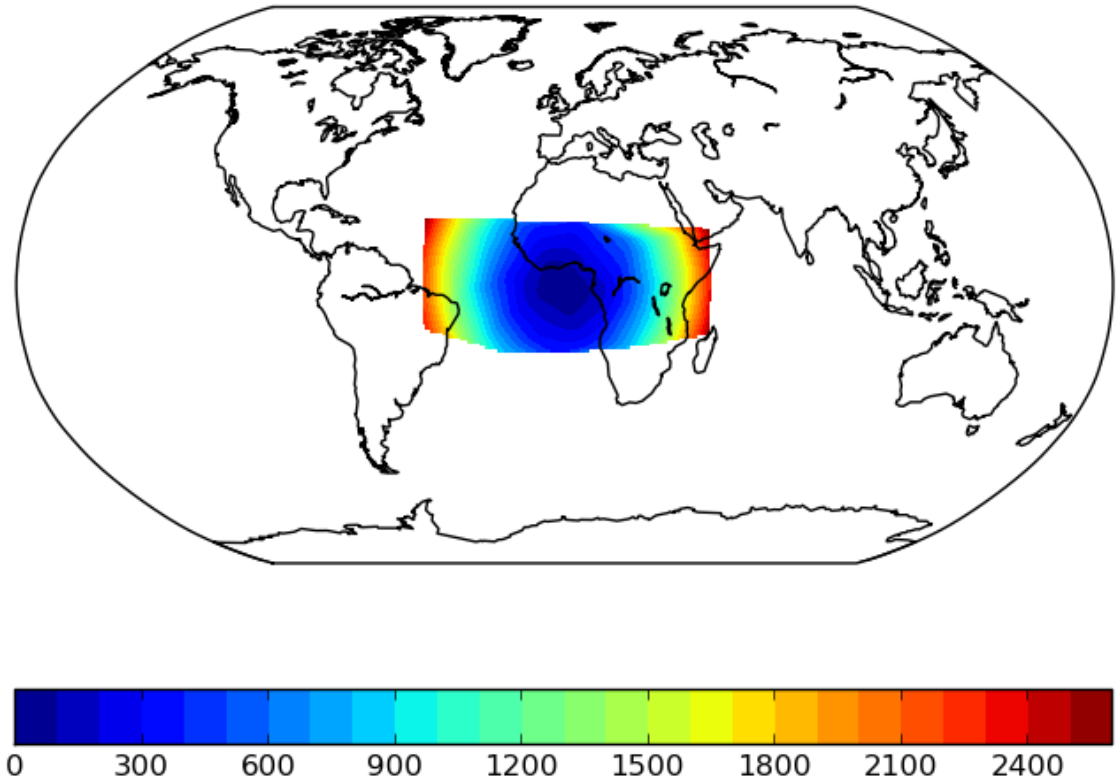

Figure 3.3: Example of map plotting with the Robinson projection using the Matplotlib Basemap toolkit.

This object can then be passed to the `contourf`, `contour` and `pcolor` functions in `fatiando.vis.mpl` and they will automatically plot using the given projection (Figure 3.3):

```
mpl.figure()
bm = mpl.basemap(area, projection='robin')
bm.drawmapboundary()
bm.drawcoastlines()
mpl.contourf(x, y, data, shape=(50, 50), levels=30,
    interp=True, basemap=bm)
mpl.colorbar(orientation='horizontal')
mpl.show()
```

3.5 Meshes and 3D plotting

The representation of 2D and 3D geometric elements is handled by the classes in the `fatiando.mesher` module. Geometric elements in *Fatiando a Terra* can be assigned physical property values, like density, magnetization, seismic wave velocity,

impedance, etc. This is done through a `props` dictionary whose keys are the name of the physical property and values are the corresponding values in SI units:

```
from fatiando import mesher
model = [
    mesher.Prism(5, 8, 3, 7, 1, 7,
        props={'density':200}),
    mesher.Prism(1, 2, 4, 5, 1, 2,
        props={'density':1000})]
```

The `fatiando.vis.myv` module contains functions to automate 3D plotting using Mayavi (RAMACHANDRAN e VAROQUAUX, 2011). The `mayavi.mlab` interface requires geometric elements to be formatted as TVTK objects. Thus, plotting functions in `fatiando.vis.myv` automatically create TVTK representations of `fatiando.mesher` objects and plot them using a suitable function of `mayavi.mlab`. Also included are utility functions for drawing axes, walls on the figure bounding box, etc. For example, the `fatiando.vis.myv.figure` function creates a figure and rotates it so that the z-axis points down, as is standard in geophysics. The following example shows how to plot the 3D right rectangular prism model that we created previously (Figure 3.4):

```
from fatiando.vis import myv
bounds = [0, 10, 0, 10, 0, 10]
myv.figure()
myv.prisms(model, 'density')
myv.axes(myv.outline(bounds))
myv.wall_bottom(bounds)
myv.wall_north(bounds)
myv.show()
```

The `fatiando.mesher` module also contains classes for collections of elements (e.g., meshes). A good example is the `PrismMesh` class that represents a structured mesh of right rectangular prisms. This class behaves as a list of `fatiando.mesher.Prism` objects and can be passed to functions that ask for a list of prisms, like `fatiando.vis.myv.prisms`. Physical properties can be assigned to the mesh using the `addprop` method (Figure 3.5):

```
mesh = mesher.PrismMesh(bounds, shape=(3, 3, 3))
mesh.addprop('density', range(mesh.size))
myv.figure()
myv.prisms(mesh, 'density')
```

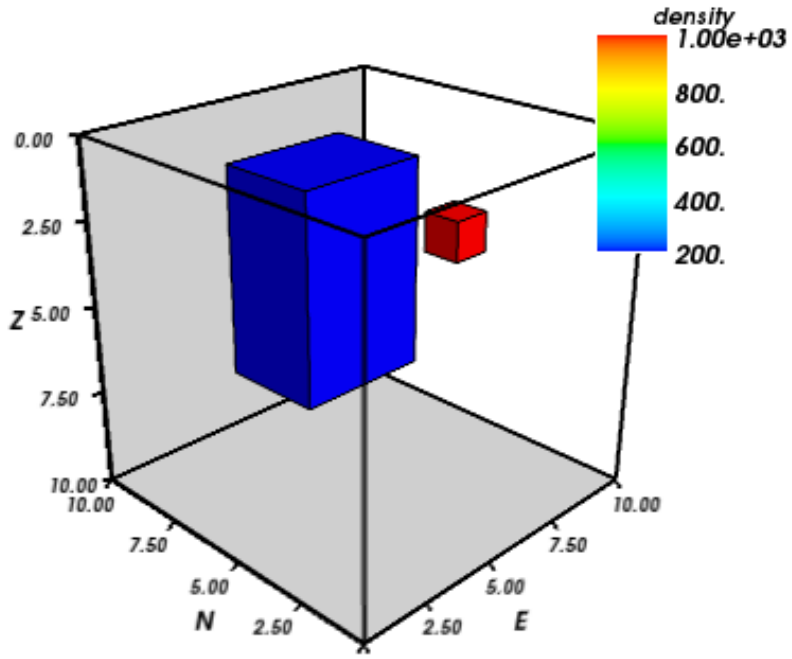

Figure 3.4: Example of plotting a list of right rectangular prisms in Mayavi.

```
myv.axes(myv.outline(bounds))
myv.show()
```

Often times the mesh is used to make a detailed model of an irregular region of the Earth's surface. In such cases, it is necessary to consider the topography of the region. The `PrismMesh` class has a `carvetopo` method that masks the prisms that fall above the topography. The example below illustrates this functionality using synthetic topography (Figure 3.6):

```
from fatiando import utils
x, y = gridder.regular(bounds[:4], (50, 50))
heights = -5 + 5*utils.gaussian2d(x, y, 10, 5,
    x0=10, y0=10)
mesh = mesher.PrismMesh(bounds, (20, 20, 20))
mesh.addprop('density', range(mesh.size))
mesh.carvetopo(x, y, heights)
myv.figure()
myv.prisms(mesh, 'density')
myv.axes(myv.outline(bounds))
myv.wall_north(bounds)
myv.show()
```

When modeling involves the whole Earth, or a large area of it, the geophysicist needs to take into account the Earth's curvature. In such cases, rectangular

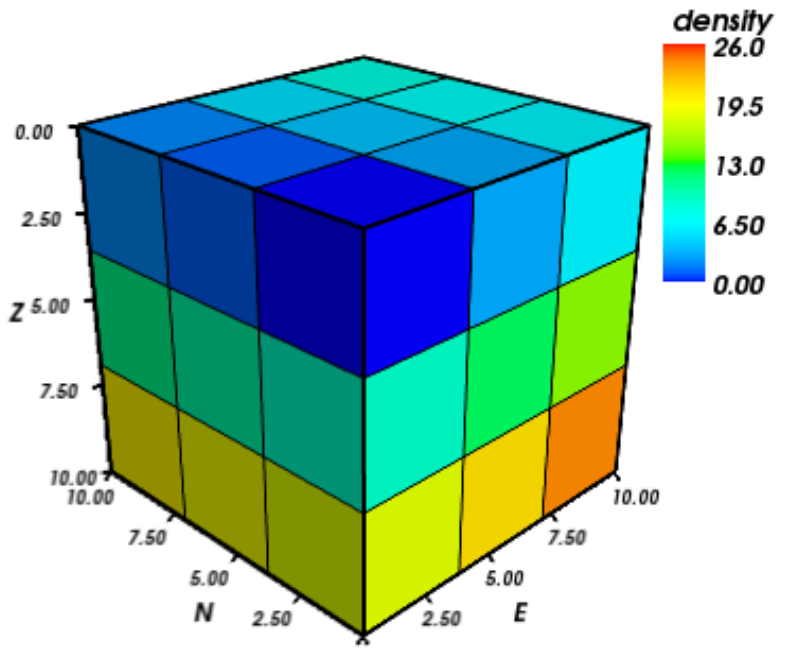

Figure 3.5: Example of generating and visualizing a structured prism mesh.

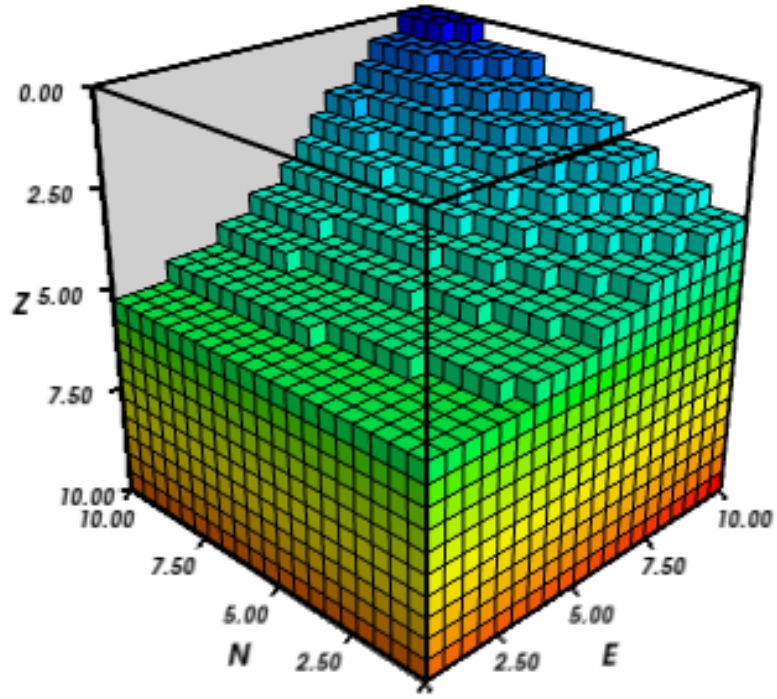

Figure 3.6: Example of generating and visualizing a prism mesh with masked topography.

prisms are inadequate for modeling and tesseroids (e.g., spherical prisms) are better suited. The `fatiando.vis.myv` module contains auxiliary functions to plot along with tesseroids: an Earth-sized sphere, meridians and parallels, as well as continental borders (Figure 3.7):

```
model = [
    mesher.Tesseroid(-60, -55, -30, -27, 500000, 0,
        props={'density':200}),
    mesher.Tesseroid(-66, -55, -20, -10, 300000, 0,
        props={'density':-100})]
fig = myv.figure(zdown=False)
myv.tesseroids(model, 'density')
myv.continents(linewidth=2)
myv.earth(opacity=1)
myv.meridians(range(0, 360, 45), opacity=0.2)
myv.parallels(range(-90, 90, 45), opacity=0.2)
# Rotate the camera to get a good view
scene = fig.scene
scene.camera.position = [21199620.406122234,
    -12390254.839673528, -14693312.866768979]
scene.camera.focal_point = [-535799.97230670298,
    -774902.33205294283, 826712.82283183688]
scene.camera.view_angle = 19.199999999999996
scene.camera.view_up = [0.33256519487680014,
    -0.47008782429014295, 0.81756824095039038]
scene.camera.clipping_range = [7009580.0037488714,
    55829873.658824757]
scene.camera.compute_view_plane_normal()
scene.render()
myv.show()
```

3.6 Forward modeling

In geophysics, the term "forward modeling" is used to describe the process of generating synthetic data from a given Earth model. Conversely, geophysical inversion is the process of estimating Earth model parameters from observed data.

The *Fatiando a Terra* packages have separate modules for forward modeling and inversion algorithms. The forward modeling functions usually take as arguments geometric elements from `fatiando.mesher` with assigned physical properties and

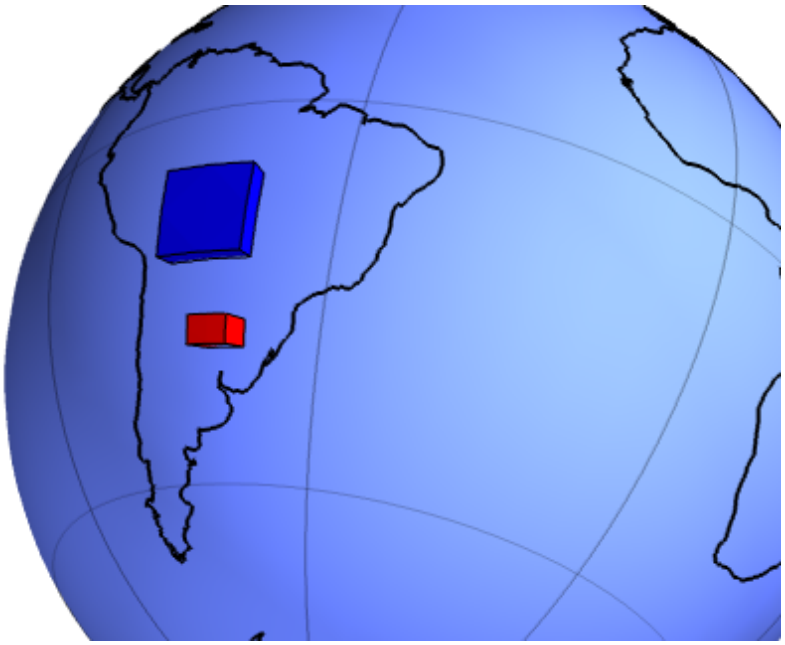

Figure 3.7: Example of creating a tesseroid (spherical prism) model and visualizing it in Mayavi.

return the synthetic data. For example, the module `fatiando.gravmag.tesseroid` is a Python implementation of the program Tesseroids (`http://leouieda.github.io/tesseroids`) and calculates the gravitational fields of tesseroids (e.g., spherical prisms). The following example shows how to calculate the gravity anomaly of the tesseroid model generated in the previous section (Figure 3.8):

```
from fatiando import gravmag
area = [-80, -30, -40, 10]
shape = (50, 50)
lons, lats, heights = gridder.regular(area, shape,
    z=2500000)
gz = gravmag.tesseroid.gz(lons, lats, heights, model)
mpl.figure()
bm = mpl.basemap(area, 'ortho')
bm.drawcoastlines()
bm.drawmapboundary()
bm.bluemarble()
mpl.title('Gravity anomaly (mGal)')
mpl.contourf(lons, lats, gz, shape, 30, basemap=bm)
mpl.colorbar()
mpl.show()
```

The module `fatiando.gravmag.polyprism` implements the method of PLOUFF (1976) to forward model the gravity fields of a 3D right polygonal prism. The

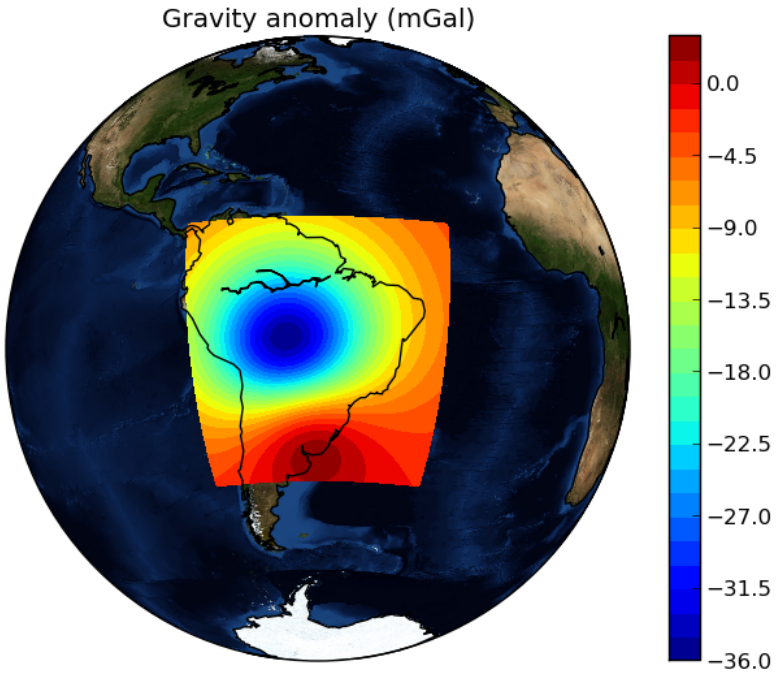

Figure 3.8: Example of forward modeling the gravity anomaly using the tesseroid model shown in Figure 3.7.

following code sample shows how to interactively generate a polygonal prism model and calculate its gravity anomaly (Figures 3.9 and 3.10):

```
# Draw a polygon and make a polygonal prism
bounds = [-1000, 1000, -1000, 1000, 0, 1000]
area = bounds[:4]
mpl.figure()
mpl.axis('scaled')
vertices = mpl.draw_polygon(area, mpl.gca(),
    xy2ne=True)
model = [mesher.PolygonalPrism(vertices, z1=0,
    z2=500, props={'density':500})]
# Calculate the gravity anomaly
shape = (100, 100)
x, y, z = gridder.scatter(area, 300, z=-1)
gz = gravmag.polyprism.gz(x, y, z, model)
mpl.figure()
mpl.axis('scaled')
mpl.title("Gravity anomaly (mGal)")
mpl.contourf(y, x, gz, shape=(50, 50),
    levels=30, interp=True)
mpl.colorbar()
```

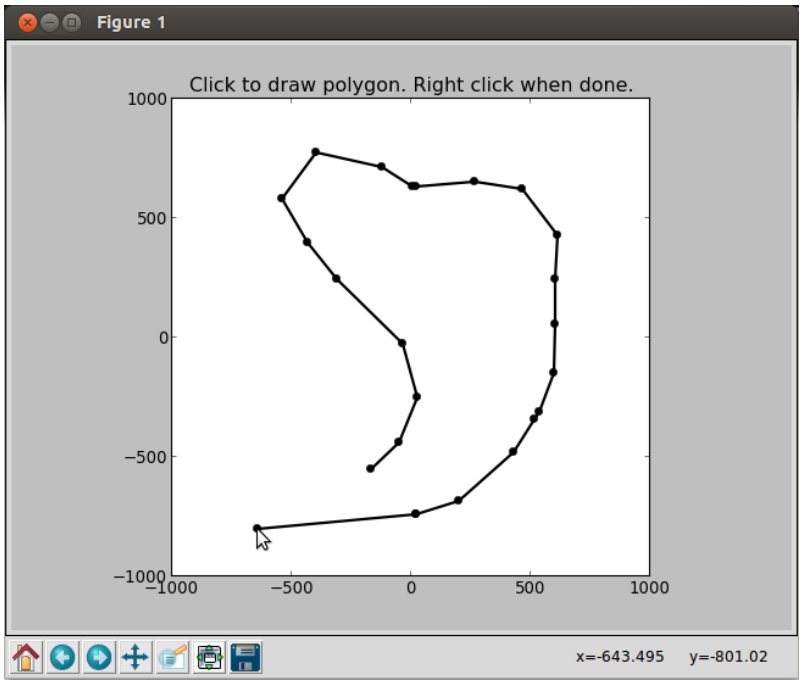

Figure 3.9: Screen-shot of interactively drawing the contour of a 3D polygonal prism, as viewed from above.

```
mpl.polygon(model[0], '.-k', xy2ne=True)
mpl.set_area(area)
mpl.m2km()
mpl.show()
myv.figure()
myv.polyprisms(model, 'density')
myv.axes(myv.outline(bounds),
        ranges=[i*0.001 for i in bounds])
myv.wall_north(bounds)
myv.wall_bottom(bounds)
myv.show()
```

3.7 Gravity and magnetic methods

Geophysics uses anomalies in the gravitational and magnetic fields generated by density and magnetization contrasts within the Earth to investigate the inner Earth structure. The *Fatiando a Terra* 0.1 release has been focused on gravity and magnetic methods. Therefore, the `fatiando.gravmag` package contains more advanced and state-of-the-art algorithms than the other packages.

The module `fatiando.gravmag.imaging` implements the imaging methods described in FEDI e PILKINGTON (2012). These methods aim to produce an image

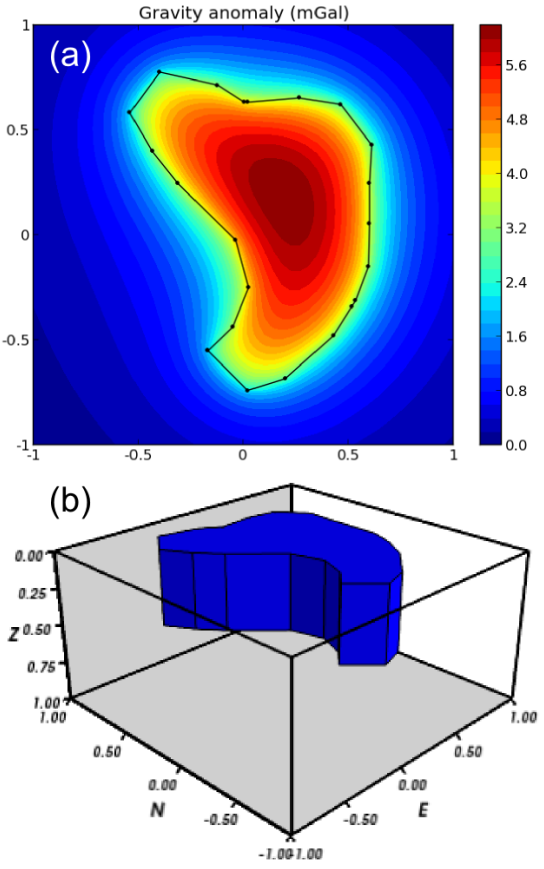

Figure 3.10: Example of forward modeling the gravity anomaly of a 3D polygonal prism. a) forward modeled gravity anomaly. b) 3D plot of the polygonal prism.

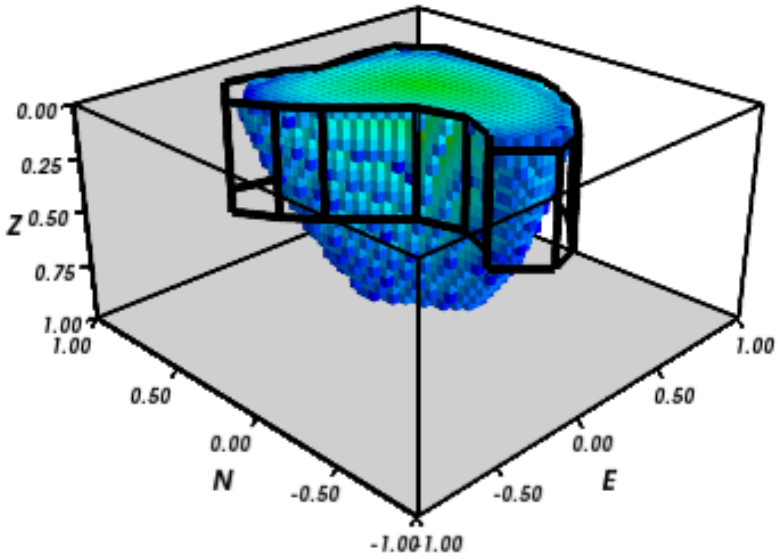

Figure 3.11: Example of using the "sandwich model" imaging method to recover a 3D image of a geologic body based on its gravity anomaly. The colored blocks are a cutoff of the imaged body. The black contours are the true source of the gravity anomaly.

of the geologic source from the observed gravity or magnetic data. The following code sample uses the "sandwich model" method (PEDERSEN, 1991) to image the polygonal prism, produced in the previous section, based on its gravity anomaly (Figure 3.11):

```
estimate = gravmag.imaging.sandwich(x, y, z, gz,
    shape, zmin=0, zmax=1000, nlayers=20, power=0.2)
body = mesher.vfilter(1.3*10**8, 1.7*10**8,
    'density', estimate)
myv.figure()
myv.prisms(body, 'density', edges=False)
p = myv.polyprisms(model, 'density',
    style='wireframe', linewidth=4)
p.actor.mapper.scalar_visibility = False
p.actor.property.color = (0, 0, 0)
myv.axes(myv.outline(bounds),
    ranges=[i*0.001 for i in bounds])
myv.wall_north(bounds)
myv.wall_bottom(bounds)
myv.show()
```

Also implemented in *Fatiando a Terra* are some recent developments in gravity and magnetic inversion methods. The method of "planting anomalous densities" by

UIEDA e BARBOSA (2012) is implemented in the `fatiando.gravmag.harvester` module. In contrast to imaging methods, this is an inversion method, i.e., it estimates a physical property distribution (density in the case of gravity data) that fits the observed data. This particular method requires the user to specify a "seed" (Figure 3.12) around which the estimated density distribution grows (Figure 3.13):

```
# Make a mesh and a seed
mesh = mesher.PrismMesh(bounds, (15, 30, 30))
seeds = gravmag.harvester.sow(
    [[200, 300, 100, {'density':500}]],
    mesh)
myv.figure()
myv.prisms([mesh[s.i] for s in seeds])
p = myv.polyprisms(model, 'density',
    style='wireframe', linewidth=4)
p.actor.mapper.scalar_visibility = False
p.actor.property.color = (0, 0, 0)
myv.axes(myv.outline(bounds),
    ranges=[i*0.001 for i in bounds])
myv.wall_north(bounds)
myv.wall_bottom(bounds)
myv.show()
# Now perform the inversion
data = [gravmag.harvester.Gz(x, y, z, gz)]
estimate = gravmag.harvester.harvest(data, seeds,
    mesh, compactness=0.1, threshold=0.0001)[0]
mesh.addprop('density', estimate['density'])
body = mesher.vremove(0, 'density', mesh)
myv.figure()
myv.prisms(body, 'density')
p = myv.polyprisms(model, 'density',
    style='wireframe', linewidth=4)
p.actor.mapper.scalar_visibility = False
p.actor.property.color = (0, 0, 0)
myv.axes(myv.outline(bounds),
    ranges=[i*0.001 for i in bounds])
myv.wall_north(bounds)
myv.wall_bottom(bounds)
myv.show()
```

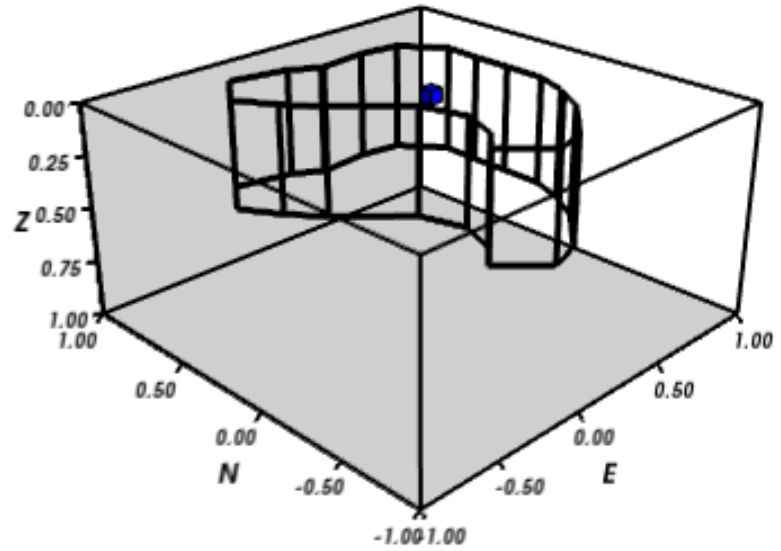

Figure 3.12: The small blue prism is the seed used by `fatiando.gravmag.harvester` to perform the inversion of a gravity anomaly. The black contours are the true source of the gravity anomaly.

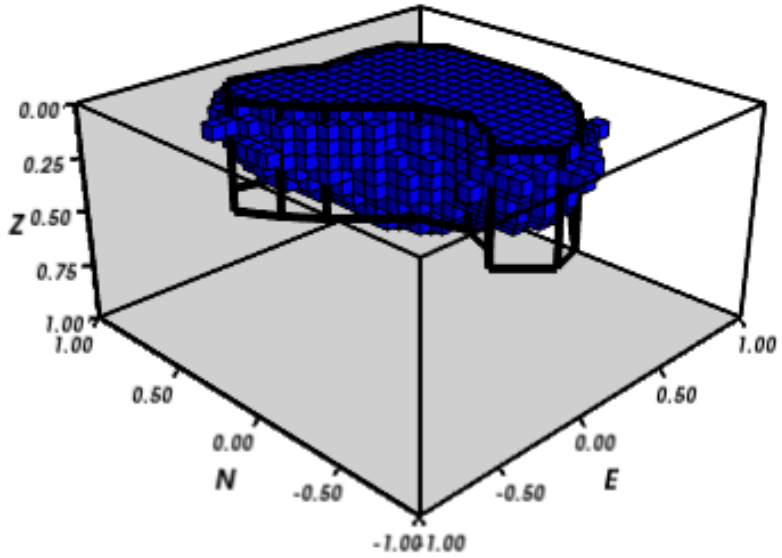

Figure 3.13: The blue prisms are the result of a gravity inversion using module `fatiando.gravmag.harvester`. The black contours are the true source of the gravity anomaly. Notice how the inversion was able to recover the approximate geometry of the true source.

3.8 A toy seismic tomography

The following example uses module `fatiando.seismic.srtomo` to perform a simplified 2D tomography on synthetic seismic wave travel-time data. To generate the travel-times we used a seismic wave velocity model constructed from an image file. The colors of the image are converted to gray-scale and the intensity is mapped to seismic wave velocity by the `img2prop` method of the `fatiando.mesher.SquareMesh` class. This model (Figure 3.14) is then used to calculate the travel-times between a random set of earthquake locations and seismic receivers (seismometers):

```
import urllib
from fatiando import mesher, utils, seismic
from fatiando.vis import mpl
area = (0, 500000, 0, 500000)
shape = (30, 30)
model = mesher.SquareMesh(area, shape)
link = '/'.join(["http://fatiando.readthedocs.org",
    "en/Version0.1/_static/logo.png"])
urllib.urlretrieve(link, 'model.png')
model.img2prop('model.png', 4000, 10000, 'vp')
quake_locations = utils.random_points(area, 40)
receiver_locations = utils.circular_points(area, 20,
    random=True)
quakes, receivers = utils.connect_points(
    quake_locations, receiver_locations)
traveltimes = seismic.ttime2d.straight(model, 'vp',
    quakes, receivers)
noisy = utils.contaminate(traveltimes, 0.001,
    percent=True)
```

Now the noise-corrupted synthetic travel-times can be used in our simplified tomography:

```
mesh = mesher.SquareMesh(area, shape)
slowness, residuals = seismic.srtomo.run(noisy,
    quakes, receivers, mesh, smooth=10**6)
velocity = seismic.srtomo.slowness2vel(slowness)
mesh.addprop('vp', velocity)
# Make the plots
mpl.figure(figsize=(9, 7))
```

```
mpl.subplots_adjust(top=0.95, bottom=0.05,
    left=0.05, right=0.95)
mpl.subplot(2, 2, 1)
mpl.title('Velocity model (m/s)')
mpl.axis('scaled')
mpl.squaremesh(model, prop='vp', cmap=mpl.cm.seismic)
mpl.colorbar(pad=0.01)
mpl.points(quakes, '*y', label="Sources")
mpl.points(receivers, '^g', label="Receivers")
mpl.m2km()
mpl.subplot(2, 2, 2)
mpl.title('Ray paths')
mpl.axis('scaled')
mpl.squaremesh(model, prop='vp', cmap=mpl.cm.seismic)
mpl.colorbar(pad=0.01)
mpl.paths(quakes, receivers)
mpl.points(quakes, '*y', label="Sources")
mpl.points(receivers, '^g', label="Receivers")
mpl.m2km()
mpl.subplot(2, 2, 3)
mpl.title('Estimated velocity (m/s)')
mpl.axis('scaled')
mpl.squaremesh(mesh, prop='vp', cmap=mpl.cm.seismic,
    vmin=4000, vmax=10000)
mpl.colorbar(pad=0.01)
mpl.m2km()
mpl.subplot(2, 2, 4)
mpl.title('Residuals (s)')
mpl.hist(residuals, bins=10)
mpl.show()
```

Even though the implementation in `fatiando.seismic.srtomo` is greatly simplified and not usable in real tomography problems, the result in Figure 3.14 illustrates interesting inverse problem concepts. Notice how the estimated velocity is blurred in the corners where no rays pass through. This is because the data (travel-times) provide no information about the velocity in those areas. Areas like those constitute the null space of the inverse problem (MENKE, 1984), where any velocity value estimated will provide an equal fit to the data. Thus, the tomography problem requires the use of prior information in the form of regularization. Most commonly used in tomography problems is the Tikhonov first-order regularization, e.g., a smoothness

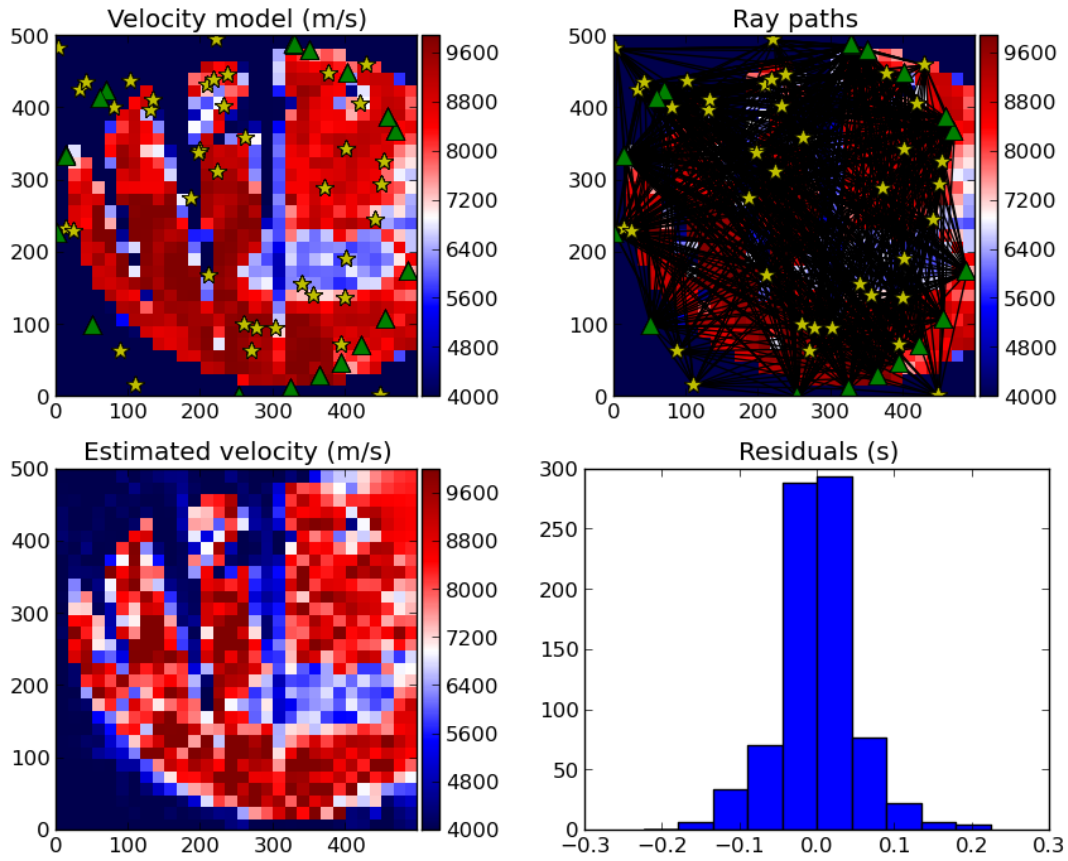

Figure 3.14: Example run of a simplified 2D tomography. The top-left panel shows the true velocity model with the locations of earthquakes (yellow stars) and receivers (green triangles). The top-right panel shows the ray-paths between earthquakes and receivers. The bottom-left panel is the velocity estimated by the tomography. The bottom-right panel is a histogram of the travel-time residuals of the tomography. Notice how the majority of residuals are close to 0 s, indicating a good fit to the data.

constraint (MENKE, 1984). The amount of smoothness imposed on the solution is controlled by the `smooth` argument of function `fatiando.seismic.srtomo.run`. That is how we are able to estimate a unique and stable solution and why the result is specially smoothed where there are no rays.

3.9 Conclusion

The *Fatiando a Terra* package provides an API to develop modeling algorithms for a variety of geophysical methods. The current version (0.1)[2] has a few state-of-the-art gravity and magnetic modeling and inversion algorithms. There are also toy problems in gravity, seismics and seismology that are useful for teaching basic concepts of geophysics, modeling, and inverse problems.

[2] As of April 2016, the latest version is 0.3.

Fatiando a Terra enables quick prototyping of new algorithms because of the collection of fast forward modeling routines and the simple syntax and high level of the Python language. After prototyping, the performance bottlenecks of these algorithms can be easily diagnosed using the advanced profiling tools available in the Python language. Optimization of only small components of code can be done without loss of flexibility using the Cython language (BEHNEL *et al.*, 2011).

The project was started as part of my PhD thesis work but grew beyond this context. The development is on-going and recently new collaborators from outside of Brazil have started contributing code to the project. The biggest challenge that *Fatiando a Terra* faces in the near future is the development of a strong user and, consequently, a developer community. This is a key part for the survival of any open-source project.

Chapter 4

Fast non-linear gravity inversion in spherical coordinates with application to the South American Moho

This chapter has been submitted for publication in the Geophysical Journal International.

4.1 Abstract

Estimating the relief of the Moho from gravity data is a computationally intensive non-linear inverse problem. What is more, the modeling must take the Earths curvature into account when the study area is of regional scale or greater. We present a regularized non-linear gravity inversion method that has a low computational footprint and employs a spherical Earth approximation. To achieve this, we combine the highly efficient Bott's method with smoothness regularization and a discretization of the anomalous Moho into tesseroids (spherical prisms). The computational efficiency of our method is attained by harnessing the fact that all matrices involved are sparse. The inversion results are controlled by three hyper-parameters: the regularization parameter, the anomalous Moho density-contrast, and the reference Moho depth. We estimate the regularization parameter using the method of hold-out cross-validation. Additionally, we estimate the density-contrast and the reference depth using knowledge of the Moho depth at certain points. We apply the proposed method to estimate the Moho depth for the South American continent using satellite gravity data and seismological data. The final Moho model is in accordance with previous gravity-derived models and seismological data. The misfit to the gravity

and seismological data is worse in the Andes and best in oceanic areas, central Brazil and Patagonia, and along the Atlantic coast. Similarly to previous results, the model suggests a thinner crust of 30-35 km under the Andean foreland basins. Discrepancies with the seismological data are greatest in the Guyana Shield, the central Solimões and Amazonas Basins, the Paraná Basin, and the Borborema province. These differences suggest the existence of crustal or mantle density anomalies that were unaccounted for during gravity data processing.

4.2 Introduction

The Mohorovičić discontinuity (or Moho) that marks the transition from the crust to the mantle, is studied almost exclusively through indirect geophysical methods. The two main geophysical methods used to estimate the depth of the Moho are seismology, with both natural and controlled sources, and gravimetry. With the advent of satellite gravimetry missions like GRACE and GOCE, gravity-derived crustal models can be produced in regional or global scales (e.g. REGUZZONI *et al.*, 2013; VAN DER MEIJDE *et al.*, 2013, 2015). New spherical harmonic gravity models that use these satellite observation, like GOCO5S (MAYER-GUERR *et al.*, 2015), provide almost homogeneous data coverage in difficult to access regions traditionally poor in terrestrial data. An example is South America, where seismologic and terrestrial gravity data are traditionally concentrated around urban centers and coastal areas, resulting in large areas (e.g., forests and mountains) devoid of data.

Estimating Moho depth from gravity data is a non-linear inverse problem. One can generalize this problem of estimating the depths of an interface separating two media, such as the sediment-basement interface of a sedimentary basin or the crust-mantle interface (Moho). Several methods have been developed over the years to solve this inverse problem, for example BARBOSA *et al.* (1999a,b); BARNES e BARRAUD (2012); BOTT (1960); LEÃO *et al.* (1996); MARTINS *et al.* (2010, 2011); OLDENBURG (1974); REGUZZONI *et al.* (2013); SANTOS *et al.* (2015); SILVA *et al.* (2006, 2014), to name a few. Solving the inverse problem is computationally demanding because it requires the construction of large dense matrices and the solution of large linear systems. As a result, some authors search for ways to increase the computational efficiency of this class of inverse problem. BOTT (1960) proposed a method based on iteratively applying corrections to a starting estimate based on the inversion residuals. The algorithm is fast because it bypasses the construction and solution of linear systems and only involves forward modeling. OLDENBURG (1974) showed that the fast FFT-based forward modeling of PARKER (1973) could be rearranged to estimate the relief. BARNES e BARRAUD (2012) use a form of adaptive discretization to compute the Jacobian, or sensitivity, matrix.

For each data point, the discretization will be progressively coarser the further way from the point. This reduces the matrix and, consequently, the linear systems to a sparse form that can be solved efficiently. Recently, SILVA *et al.* (2014) extended and generalized the original method of BOTT (1960) and SANTOS *et al.* (2015) used this extension to estimate a basement relief with sharp boundaries.

A spherical Earth approximation is preferred when estimating the Moho depth from gravity data in continental and global scale studies. WIECZOREK e PHILLIPS (1998) developed a spherical harmonic equivalent of the Parker-Oldenburg FFT algorithm and applied it to estimate the crustal structure of the Moon. REGUZZONI *et al.* (2013) use a spherical Earth approximation to estimate the global Moho relief using data from the GOCE satellite mission. Another approach is to use non-spectral (space domain) gravity inversion methods. Many such methods were developed for estimating the basement relief of a sedimentary basin (e.g., BARBOSA *et al.*, 1999a,b; MARTINS *et al.*, 2010, 2011; SUN e LI, 2014). These methods approximate the sedimentary pack by a set of juxtaposed right-rectangular prisms. The top of the prisms coincide with the Earth's surface and the prisms' thicknesses represent the depths to the basement and are the parameters to be estimated in the inversion. The use of rectangular prisms implies a planar Earth approximation and may not be adequate for depth-to-Moho estimates in a continental-scale study. A straightforward way to circumvent this hindrance is to adapt one of the methods developed for rectangular prisms to use tesseroids (spherical prisms). One of the difficulties of this approach is that the forward problem for a tesseroid must be solved numerically. Two alternatives proposed in the literature to the numerical solution are Taylor series expansion (GROMBEIN *et al.*, 2013; HECK e SEITZ, 2007) and the Gauss-Legendre Quadrature (ASGHARZADEH *et al.*, 2007). Numerical experiments by WILD-PFEIFFER (2008) suggest that the Gauss-Legendre Quadrature (GLQ) offers superior results. However, the GLQ suffers from numerical instability when the computation point is close to the tesseroid (ASGHARZADEH *et al.*, 2007). To overcome the numerical instability, LI *et al.* (2011) proposed an adaptive discretization algorithm which was later improved upon by UIEDA *et al.* (2016).

In any gravity inversion for estimating the relief of an interface, two hyper-parameters control the inversion results: the density-contrast between the two media and the reference level around which the interface undulates. The reference level is the constant depth of the Normal Earth Moho in the case of the anomalous Moho. For regularized inversions, an additional hyper-parameter is the regularization parameter that balances the relative importance between the data-misfit measure and the regularizing function. The two most commonly used methods for estimating the regularization parameter are the L-curve criterion and Generalized Cross Val-

idation (GCV). FARQUHARSON e OLDENBURG (2004) provide for a thorough comparison of both methods. Estimating the density-contrast in a sedimentary basin context has been tackled by SILVA *et al.* (2006) and MARTINS *et al.* (2010) when the basement depth is known at a few points. To the authors' knowledge no attempt has been made to estimate the reference level.

We present a non-linear gravity inversion to estimate the Moho depth in a spherical Earth approximation. Our method is based on the SILVA *et al.* (2014) Gauss-Newton formulation of the method of BOTT (1960). We use tesseroids to discretize the anomalous Moho and the adaptive discretization algorithm of UIEDA *et al.* (2016) for the forward modeling. The stability of the inversion is achieved through smoothness regularization. In order to maintain the computational efficiency of Bott's method, we exploit the sparse nature of all matrices involved in the computations. We employ a variant of GCV known as hold-out cross-validation (KIM, 2009) to estimate the regularization parameter. Additionally, we estimate the density-contrast and reference level simultaneously in a second cross-validation. Similarly to SILVA *et al.* (2006) and MARTINS *et al.* (2010), this cross-validation procedure uses knowledge of the Moho depths at certain points. Finally, we apply the proposed method to estimate the Moho depth for South America using gravity data from the GOCO5S model (MAYER-GUERR *et al.*, 2015) and the seismological data of ASSUMPÇÃO *et al.* (2013).

4.3 Methodology

In potential field methods, we must isolate the target anomalous density distribution before modeling and inversion. In our case, the target is the relief of the real Moho undulating around a reference Moho. We do this by removing all other effects from the gravity observations. The first correction is to remove the scalar gravity of an ellipsoidal reference Earth (the Normal Earth), hereafter denoted as γ. This effect is calculated on the same point P where the gravity observation was made (Fig 4.1a-b). $\gamma(P)$ is calculated using the closed-form solution presented by LI e GÖTZE (2001). The difference between the observed gravity at point P ($g(P)$) and Normal gravity at the same point is known as the gravity disturbance,

$$\delta(P) = g(P) - \gamma(P). \tag{4.1}$$

The disturbance contains only the gravitational effects of density distributions that are anomalous with respect to the Normal Earth (see Fig. 4.1c). This includes all masses above the surface of the ellipsoid (the topography), the mass deficiency of the oceans, the mass deficiency of sedimentary basins, crustal sources (e.g., igneous

intrusions, lateral density changes, etc), heterogeneities below the upper mantle, and the effect of the difference between the real Moho topography and the Moho of the Normal Earth.

To estimate the anomalous Moho relief from gravity data, we must first isolate its gravitational attraction. Thus, all other gravitational effects must be either removed or assumed negligible. Here, we will remove the gravitational effect produced by the known topography and ocean masses to obtain the full Bouguer disturbance (Fig 4.1d),

$$\delta_{bg}(P) = \delta(P) - g_{topo}(P). \tag{4.2}$$

We will also remove the gravitational effect of know sedimentary basins but assume that the effects of other crustal and mantle sources are negligible. Thus, the only effect left will be that of the anomalous Moho relief (Fig 4.1e). The gravitational attraction of the topography, oceans, and basins are calculated in a spherical Earth approximation by forward modeling using tesseroids (Fig. 4.2). The tesseroid effects are calculated numerically using Gauss-Legendre Quadrature (GLQ) integration (ASGHARZADEH *et al.*, 2007). The accuracy of the GLQ integration is improved by the adaptive discretization scheme of UIEDA *et al.* (2016).

4.3.1 Parametrization and the forward problem

We parameterize the forward problem by discretizing the anomalous Moho into a grid of $M_{lon} \times M_{lat} = M$ juxtaposed tesseroids (Fig 4.1f). The true (real Earth) Moho varies in depth with respect to the Moho of the Normal Earth. Hereafter we will refer to the depth of the Normal Earth Moho as z_{ref} (see Fig. 4.1b). If the true Moho is above z_{ref}, the top of the kth tesseroid is the Moho depth z_k, the bottom is z_{ref}, and the density-contrast ($\Delta\rho$) is positive (red tesseroids in Fig 4.1f). If the Moho is below z_{ref}, the top of the tesseroid is z_{ref}, the bottom is z_k, and $\Delta\rho$ is negative (grey tesseroids in Fig 4.1f).

Considering that the absolute value of the density-contrasts of the tesseroids is a fixed parameter, the predicted gravity anomaly of the Moho is a non-linear function of the parameters z_k, $k = 1, \ldots, M$,

$$d_i = f_i(\mathbf{p}), \tag{4.3}$$

in which d_i is the ith element of the N-dimensional predicted data vector $\mathbf{d}$, $\mathbf{p}$ is the M-dimensional parameter vector containing the M Moho depths (z_k), and f_i is the ith non-linear function that maps the parameters onto the data. The functions f_i are the radial component of the gravitational attraction of the tesseroid Moho

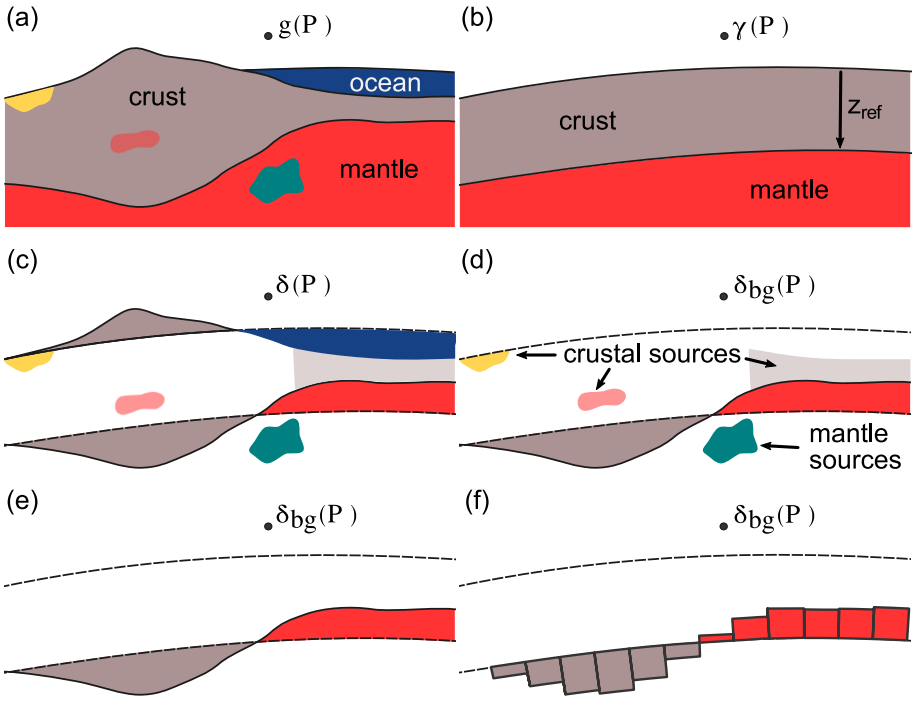

Figure 4.1: Sketch of the stages in gravity data correction and the discretization of the anomalous Moho relief using tesseroids. (a) The Earth and the measured gravity at point P ($g(P)$). (b) The Normal Earth and the calculated normal gravity at point P ($\gamma(P)$). z_{ref} is the depth of the Normal Earth Moho. (c) The gravity disturbance ($\delta(P)$) and the corresponding density anomalies after removal of the normal gravity: topography, oceans, crustal and mantle heterogeneities, and the anomalous Moho. (d) The Bouguer disturbance ($\delta_{bg}(P)$) after topographic correction and the remaining density anomalies. (e) All density anomalies save the anomalous Moho are assumed to have been removed before inversion. (f) The discretization of the anomalous Moho in tesseroids. Grey tesseroids will have a negative density contrast while red tesseroids will have a positive one.

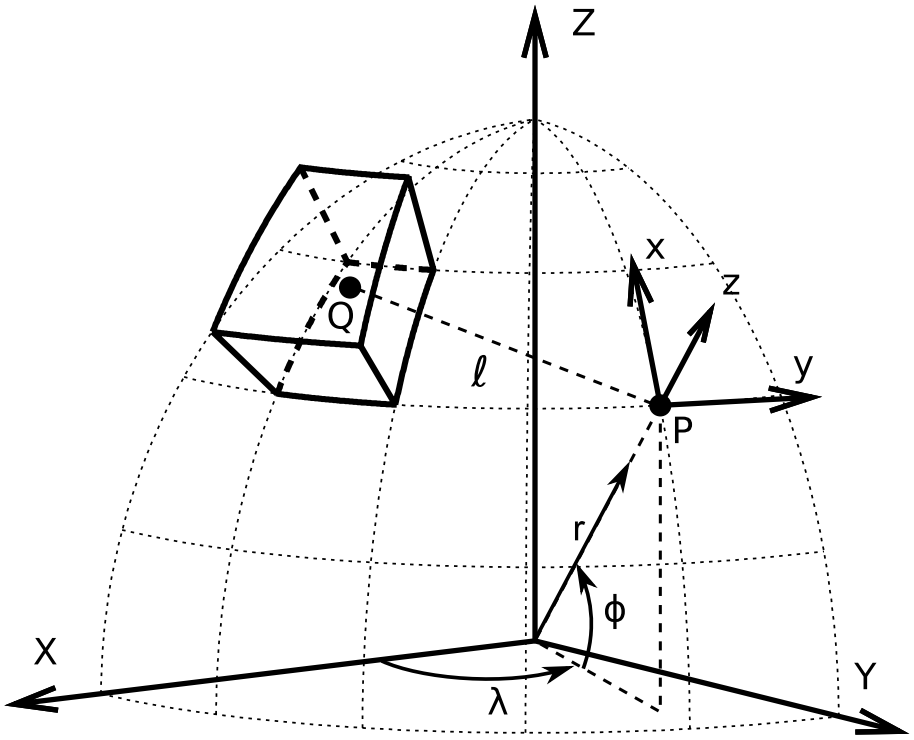

Figure 4.2: Sketch of a tesseroid (spherical prism) in a geocentric coordinate system (X, Y, Z). Observations are made at point P with respect to it's local North-oriented coordinate system (x, y, z). After UIEDA (2015).

model.

4.3.2 Inverse problem

We wish to estimate the parameter vector $\mathbf{p}$ from a set of observed gravity data $\mathbf{d}^o$. The least-squares estimate is the one that minimizes the data-misfit function

$$\phi(\mathbf{p}) = [\mathbf{d}^o - \mathbf{d}(\mathbf{p})]^T [\mathbf{d}^o - \mathbf{d}(\mathbf{p})]. \tag{4.4}$$

Function $\phi(\mathbf{p})$ is non-linear with respect to $\mathbf{p}$. Thus, we can determine its minimum using gradient-based iterative optimization methods like Gauss-Newton or Steepest Descent. Such methods start from an initial approximation to the model parameter vector $\mathbf{p}^0$ and estimate a parameter perturbation vector $\mathbf{\Delta p}^0$. The perturbation vector is used to update $\mathbf{p}^0$ to $\mathbf{p}^1 = \mathbf{p}^0 + \mathbf{\Delta p}^0$. This procedure is repeated until a minimum of function $\phi(\mathbf{p})$ (Eq. 4.4) is reached.

For the Gauss-Newton method, the parameter perturbation vector at the kth iteration $\mathbf{\Delta p}^k$ is obtained by solving the linear system

$$\mathbf{H}^k \mathbf{\Delta p}^k = -\nabla \phi^k, \tag{4.5}$$

in which $\nabla \phi^k$ and $\mathbf{H}^k$ are, respectively, the gradient vector and the Hessian matrix

of $\phi(\mathbf{p})$.

The gradient vector and the Gauss-Newton approximation of the Hessian matrix of $\phi(\mathbf{p})$ are, respectively,

$$\nabla\phi^k = -2\mathbf{A}^{k^T}[\mathbf{d}^o - \mathbf{d}(\mathbf{p}^k)], \tag{4.6}$$

and

$$\mathbf{H}^k \approx 2\mathbf{A}^{k^T}\mathbf{A}^k, \tag{4.7}$$

in which $\mathbf{A}^k$ is the $N \times M$ Jacobian or sensitivity matrix whose elements are

$$A^k_{ij} = \frac{\partial f_i}{\partial p_j}(\mathbf{p}^k). \tag{4.8}$$

4.3.3 Regularization

Non-linear gravity inversions for estimating the relief of an interface separating two media (like the Moho) are ill-posed and require additional constraints in the form of regularization (SILVA *et al.*, 2001). A common approach is to use the first-order Tikhonov regularization (TIKHONOV e ARSENIN, 1977) to impose smoothness on the solution. The cost function for smoothness regularization is given by

$$\theta(\mathbf{p}) = \mathbf{p}^T\mathbf{R}^T\mathbf{R}\mathbf{p}, \tag{4.9}$$

where $\mathbf{R}$ is an $L \times M$ finite-difference matrix representing L first-order differences between adjacent tesseroids.

To transform the ill-posed inverse problem into a well-posed one via Tikhonov regularization, we adopted the well-established procedure of formulating a constrained inverse problem that is solved by minimizing an unconstrained goal function

$$\Gamma(\mathbf{p}) = \phi(\mathbf{p}) + \mu\theta(\mathbf{p}), \tag{4.10}$$

in which μ is the regularization parameter that controls the balance between fitting the observed data and obeying the smoothness constraint imposed by the regularizing function $\theta(\mathbf{p})$ (Eq. 4.9).

The goal function $\Gamma(\mathbf{p})$ is also non-linear with respect to $\mathbf{p}$ and can be minimized using the Gauss-Newton method. The gradient vector and Hessian matrix of the goal function are, respectively,

$$\nabla\mathbf{\Gamma}^k = -2\mathbf{A}^{k^T}[\mathbf{d}^o - \mathbf{d}(\mathbf{p}^k)] + 2\mu\mathbf{R}^T\mathbf{R}\mathbf{p}^k, \tag{4.11}$$

and

$$\mathbf{H}^k = 2{\mathbf{A}^k}^T\mathbf{A}^k + 2\mu\mathbf{R}^T\mathbf{R}. \tag{4.12}$$

At the kth iteration, the parameter perturbation vector $\mathbf{\Delta p}^k$ is obtained by solving the linear equation system

$$\left[{\mathbf{A}^k}^T\mathbf{A}^k + \mu\mathbf{R}^T\mathbf{R}\right]\mathbf{\Delta p}^k = {\mathbf{A}^k}^T[\mathbf{d}^o - \mathbf{d}(\mathbf{p}^k)] - \mu\mathbf{R}^T\mathbf{R}\mathbf{p}^k. \tag{4.13}$$

Estimating the Moho depths using the above equations is computationally costly because of two main factors: (1) the evaluation and storage of the dense $N \times M$ Jacobian matrix $\mathbf{A}^k$ and (2) the solution of the resulting $M \times M$ equation system (not required for Steepest Descent). In practice, the derivatives in the Jacobian (Eq. 4.8) are often calculated through a first-order finite-difference approximation. Thus, evaluating $\mathbf{A}^k$ requires $2 \times M \times N$ forward modeling operations for each iteration of the gradient descent algorithm. These computations are performed for each iteration of the optimization of the goal function $\Gamma(\mathbf{p})$.

4.3.4 Bott's method

BOTT (1960) developed an efficient method to determined the depth of the basement of a sedimentary basin from gravity observations. The method requires data on a regular grid of $N_x \times N_y = N$ observations. The basement relief is then discretized into an equal grid of $M_x \times M_y = M$ elements with $M_x = N_x$ and $M_y = N_y$. Bott's iterative method starts with an initial approximation of the basement depths $\mathbf{p}^0$ equal to the null vector. The method updates the approximation by calculating a parameter perturbation vector $\mathbf{\Delta p}^k$ using the formula

$$\mathbf{\Delta p}^k = \frac{\mathbf{d}^o - \mathbf{d}(\mathbf{p}^k)}{2\pi G\Delta\rho}, \tag{4.14}$$

in which G is the gravitational constant and $\Delta\rho$ is the contrast between the density of the sediments and the reference density. The iterative process stops when the inversion residuals $\mathbf{r}^k = \mathbf{d}^o - \mathbf{d}(\mathbf{p}^k)$ fall below the assumed noise level of the data.

SILVA *et al.* (2014) showed that Bott's method can be formulated as a special case of the Gauss-Newton method (Eq. 4.5) by setting the Jacobian matrix (Eq. 4.8) to

$$\mathbf{A} = 2\pi G\Delta\rho\mathbf{I}, \tag{4.15}$$

where $\mathbf{I}$ is the identity matrix. In this framework, Bott's method uses a Bouguer plate approximation of the gravitational effect of the relief, $d_i = 2\pi G\Delta\rho z_i$. The derivative of d_i with respect to the parameter z_i is $2\pi G\Delta\rho$, thus linearizing the Ja-

cobian matrix. However, the non-linearity of the predicted data $\mathbf{d}(\mathbf{p}^k)$ is preserved.

One of the advantages of Bott's method over the traditional Gauss-Newton or Steepest Descent is the elimination of the computation and storage of the dense Jacobian matrix $\mathbf{A}^k$. Furthermore, Bott's method also does not require the solution of equation systems. However, a disadvantage of Bott's method is that it suffers from instability (SILVA *et al.*, 2014). A common approach to counter this issue is to apply a smoothing filter after the inversion to the unstable estimate, as in SILVA *et al.* (2014).

4.3.5 Regularized Bott's method in spherical coordinates

We propose a regularized version of Bott's method to invert gravity data for estimating the depth of the Moho in spherical coordinates. To adapt Bott's method to spherical coordinates, we replace the right-rectangular prisms in the forward modeling ($\mathbf{d}(\mathbf{p}^k)$ in Eq. 4.14) with tesseroids. The tesseroid forward modeling uses the adaptive discretization algorithm of UIEDA *et al.* (2016) to achieve accurate results. Furthermore, our formulation maintains the regularized solution for the Gauss-Newton method (Eq. 4.13) but replaces the full Jacobian matrix with the Bouguer plate approximation (Eq. 4.15). Here, the Jacobian matrix is a diagonal matrix whose elements are invariant along successive iterations. Using this approximation eliminates the cost of computing and storing the full Jacobian matrix $\mathbf{A}^k$ at each iteration (Eq. 4.8). Matrix arithmetic operations can be performed efficiently by taking advantage of the sparse nature of matrices $\mathbf{A}$ and $\mathbf{R}$ (respectively, Eq. 4.15 and 4.9). The same is true for solving the equation system in the Gauss-Newton method (Eq. 4.13). However, the computational cost of forward modeling is still present. Particularly, forward modeling using tesseroids is more computationally intensive than using right-rectangular prisms because of the numerical integration and adaptive discretization (UIEDA *et al.*, 2016). We show later in this article that sparse matrix multiplications and solving the sparse linear system in Eq. 4.13 account for less than 0.1% of the computation time required for a single inversion. Hence, by employing the use of sparse matrices, our formulation retains the efficiency of Bott's method while stabilizing the solution through the well established formalism of Tikhonov regularization.

4.3.6 Estimating the inversion hyper-parameters

Parameters that influence the inversion result but are not estimated directly in the inversion are known as hyper-parameters. In the case of our regularized Moho depth inversion, the hyper-parameters are the regularization parameter μ (Eq. 4.10), the Moho density-contrast $\Delta\rho$ (Eq. 4.15), and the depth of the Normal Earth Moho, or

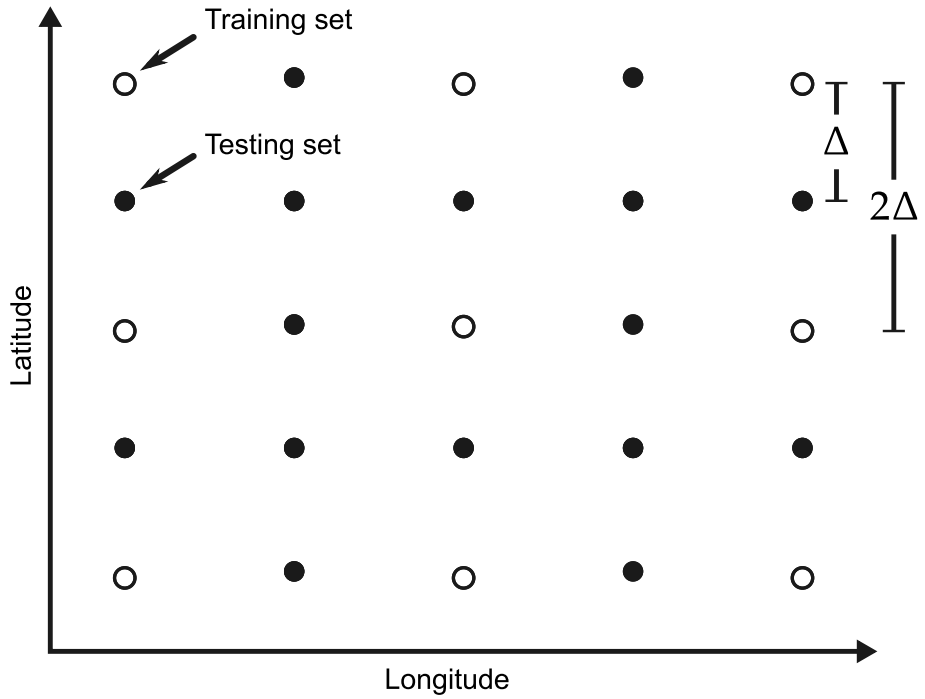

Figure 4.3: Sketch of a data grid separated into the training (open circles) and testing (black dots) data sets. The training data set is still displayed on a regular grid but with twice the grid spacing of the original data grid.

reference level, z_{ref} (Fig. 4.1b).

We estimate these hyper-parameters in two steps. First, we assume fixed values for z_{ref} and $\Delta\rho$ and perform a cross-validation procedure (HANSEN, 1992) to estimate an optimal value for μ. Our investigations suggest that the optimal value of μ does not depend on the particular values of z_{ref} and $\Delta\rho$ used. Second, we use the estimated μ to perform a second cross-validation procedure to estimate z_{ref} and $\Delta\rho$. The outcome of both cross-validations is not only the values of the three hyper-parameters but the final estimated Moho depths.

Estimating the regularization parameter

The regularization parameter μ controls how much smoothness is applied to the inversion result. An optimal value of μ will stabilize and smooth the solution while not compromising the fit to the observed data. Two widely used methods to estimate an optimal μ are the L-curve criterion and cross-validation (HANSEN, 1992). Here, we will adopt the hold-out method of cross-validation (KIM, 2009). The hold-out method consists of splitting the observed data set into two independent parts: a training set $\mathbf{d}^o_{inv}$ and a testing set $\mathbf{d}^o_{test}$. The training set is used in the inversion while the testing set is kept back and used to judge the quality of the chosen value of μ. For a value of the regularization parameter μ_n, the training set is inverted using μ_n to obtain an estimate $\hat{\mathbf{p}}^n$. This estimate is used to calculate predicted data

on the same points as the testing set via forward modeling

$$\mathbf{d}^n_{test} = \mathbf{f}(\hat{\mathbf{p}}^n). \tag{4.16}$$

The metric chosen to evaluate μ_n is the mean square error (MSE) of the misfit between the observed and predicted testing data sets,

$$MSE_n = \frac{\|\mathbf{d}^o_{test} - \mathbf{d}^n_{test}\|^2}{N_{test}}, \tag{4.17}$$

in which N_{test} is the number of data in the testing set. The optimal value of μ will be the one that minimizes the MSE, i.e. the one that best predicts the testing data. We emphasize that the inversion is performed on the training data set only.

The algorithm for the hold-out cross-validation is summarized as follows:

1. Divide the observed data into the training ($\mathbf{d}^o_{inv}$) and testing ($\mathbf{d}^o_{test}$) sets.
2. For each $\mu_n \in [\mu_1, \mu_2, \ldots, \mu_{N_\mu}]$:
   (a) Estimate $\hat{\mathbf{p}}^n$ by inverting the training set $\mathbf{d}^o_{inv}$.
   (b) Use $\hat{\mathbf{p}}^n$ to calculate the predicted testing set $\mathbf{d}^n_{test}$ using Eq. 4.16.
   (c) Calculate the mean square error MSE_n using Eq. 4.17.
3. The final solution is the $\hat{\mathbf{p}}^n$ corresponding to the smallest MSE_n.

The separation of the training and testing data sets is commonly done by taking random samples from the full data set. However, we cannot perform the separation in this way because Bott's method requires data on a regular grid as well as having model elements directly below each data point. Thus, we take as our training set the points from the observed data grid that fall on a similar grid but with twice the grid spacing (open circles in Fig. 4.3). All other points from the original data grid make up the testing data set (black dots in Fig. 4.3). This separation will lead to a testing data set with more points than the training data set. A way to balance this loss of data in the inversion is to generate a data grid with half of the desired grid spacing, either through interpolation or from a spherical harmonic model.

Estimating z_{ref} and $\Delta\rho$

The depth of the Normal Earth Moho (z_{ref}) and the density-contrast of the anomalous Moho ($\Delta\rho$) are other hyper-parameters of the inversion. That is, their value influences the final solution but they are not estimated during the inversion. Both hyper-parameters cannot be determined from the gravity data alone. Estimating z_{ref} and $\Delta\rho$ requires information that is independent of the gravity data, such as

knowledge of the parameters at certain points. This information can be used in a manner similar to the cross-validation described in the previous section. In this study, we use point estimates of the Moho depth to determine the optimal values of z_{ref} and $\Delta\rho$. These points will generally come from seismologic studies, like receiver functions, surface wave dispersion, and deep refraction experiments.

Let $\mathbf{z}_s^o$ be a vector of N_s known Moho depths. We use the mean square error (MSE) as a measure of how well a given inversion output $\hat{\mathbf{p}}^{l,m}$ fits the know depths. The optimal values of z_{ref} and $\Delta\rho$ are the ones that best fit the independent known Moho depths (i.e., produce the smallest MSE). However, the points do not necessarily coincide with the model elements of the inversion. Before computing the MSE, we interpolate $\hat{\mathbf{p}}^{l,m}$ on the known points to obtain the predicted depths $\mathbf{z}_s^{l,m}$. The MSE is defined as

$$MSE = \frac{\|\mathbf{z}_s^o - \mathbf{z}_s^{l,m}\|^2}{N_s}. \tag{4.18}$$

The algorithm for estimating z_{ref} and $\Delta\rho$ is:

1. For every combination of $z_{ref,l} \in [z_{ref,1}, z_{ref,2}, \ldots, z_{ref,N_z}]$ and $\Delta\rho_m \in [\Delta\rho_1, \Delta\rho_2, \ldots, \Delta\rho_{N_\rho}]$:
   (a) Perform the inversion on the training data set $\mathbf{d}_{inv}^o$ using $z_{ref,l}$, $\Delta\rho_m$, and the previously estimated value of μ. The inversion output is the vector $\hat{\mathbf{p}}^{l,m}$.
   (b) Interpolate $\hat{\mathbf{p}}^{l,m}$ on the known points to obtain the predicted depths $\mathbf{z}_s^{l,m}$.
   (c) Calculate the MSE between $\mathbf{z}_s^o$ and $\mathbf{z}_s^{l,m}$ using Eq. 4.18.
2. The final solution is the $\hat{\mathbf{p}}^{l,m}$ corresponding to the smallest MSE.

A similar approach was used by SILVA *et al.* (2006) and MARTINS *et al.* (2010) to estimate the parameters defining the density-contrast variation with depth of a sedimentary basin. VAN DER MEIJDE *et al.* (2013) also had a similar methodology for dealing with the hyper-parameters, though in a less formalized way.

4.3.7 Software implementation

The inversion method proposed here is implemented in the Python programming language. The software is freely available under the terms of the BSD 3-clause open-source software license. Our implementation relies on the open-source libraries scipy and numpy (JONES *et al.*, 2001, `http://scipy.org`) for array-based computations, matplotlib (HUNTER, 2007, `http://matplotlib.org`) and seaborn (WASKOM *et al.*, 2015, `http://stanford.edu/~mwaskom/software/seaborn`) for plots and

maps, and *Fatiando a Terra* (UIEDA *et al.*, 2013b, `http://www.fatiando.org`) for geophysics specific tasks, particularly for forward modeling using tesseroids. We use the scipy.sparse package for sparse matrix arithmetic and linear algebra. The sparse linear system in Eq. 4.13 is solved using the conjugate gradient method implemented in scipy.sparse.

The computational experiments (e.g., data processing, synthetic tests, real data application) were performed in Jupyter (formerly IPython) notebooks (PÉREZ e GRANGER, 2007, `http://jupyter.org/`). The notebook files combine the source code used to run the experiments, the results and figures generated by the code, and rich text to explain and document the analysis.

All source code, Jupyter notebooks, data, and results can be found at the online repository `https://github.com/pinga-lab/paper-moho-inversion-tesseroids`. The repository also contains instructions for replicating all results presented here. An archived version of this repository is also available at `http://dx.doi.org/...` (**Note to reviewers: the archived version will be uploaded upon publication**).

4.4 Application to synthetic data

We test and illustrate the proposed inversion method by applying it to two noise-corrupted synthetic data sets. The first one is generated by a simple Moho model simulating the transition from a thicker continental crust to a thinner oceanic crust. This application uses cross-validation to estimate the regularizing parameter (μ) while assuming that the anomalous Moho density-contrast ($\Delta\rho$) and the Normal Earth Moho depth (z_{ref}) are known quantities. This first test is simplified in order to investigate solely the efficiency of the inversion and the cross-validation procedure to estimate μ. The second data set is generated by a more complex model derived from the South American portion of the global CRUST1.0 model (LASKE *et al.*, 2013). This application uses cross-validation to estimate all three hyper-parameters: μ, $\Delta\rho$, and z_{ref}. The model and corresponding synthetic data are meant to simulate with more fidelity the real data application.

4.4.1 Simple model

We simulate the transition from a continental-type Moho to an oceanic-type Moho using a model composed of $M_{lat} \times M_{lon} = 40 \times 50$ grid of juxtaposed tesseroids (a total of $M = 2000$ model elements). The anomalous Moho density-contrast is $\Delta\rho = 400\ kg/m^3$ and the Normal Earth Moho depth is $z_{ref} = 30\ km$. Fig. 4.4a shows the model Moho depths where we can clearly see an eastward crustal thinning.

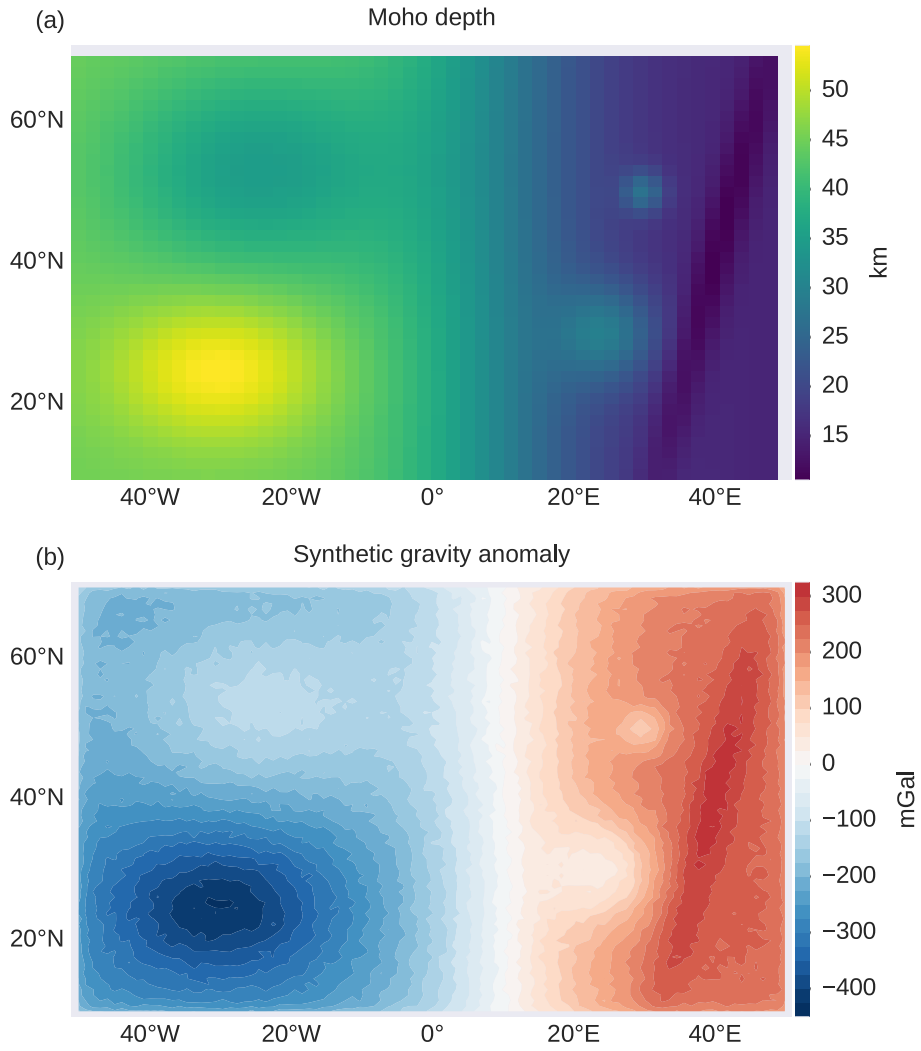

Figure 4.4: A simple Moho model made of tesseroids for synthetic data application. (a) The Moho depth of the model in kilometers. The model transitions from a deep Moho in the right to a shallow Moho in left, simulating the transition between a continental and an oceanic Moho. Each pixel in the pseudo-color image corresponds to a tesseroid of the model. (b) Noise-corrupted synthetic gravity data generated from the model shown in (a).

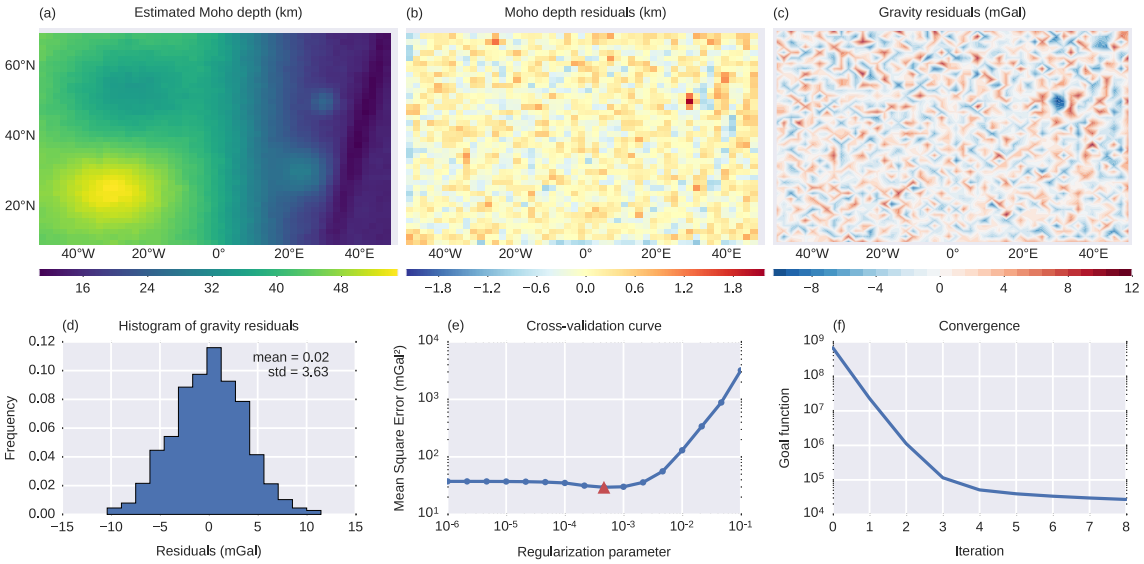

Figure 4.5: Results from the inversion of the simple synthetic data. (a) The estimated Moho depth. (b) The Moho depth residuals (difference between the true and estimated Moho depths). (c) The gravity residuals (difference between the observed and predicted gravity data). (d) Histogram of the gravity residuals shown in c, with the calculated mean and standard deviation (std) of the residuals in mGal. (e) Cross-validation curve used to determine the optimal regularization parameter (Eq. 4.10). Both axis are in logarithmic scale. The minimum Mean Square Error (Eq. 4.17) is found at $\mu = 0.00046$ (red triangle). (f) Goal function value (Eq. 4.10) per Gauss-Newton iteration showing the convergence of the gradient descent. The y-axis is in logarithmic scale.

In Fig. 4.4a, each pixel in the pseudo-color image corresponds to a tesseroid of the model.

The synthetic data were forward modeled on a regular grid of $N_{lat} \times N_{lon} = 79 \times 99$ points (a total of $N = 7821$ observations) at a constant height of 50 km. The data were contaminated with pseudo-random noise sampled from a normal distribution with zero mean and 5 mGal standard deviation. Fig. 4.4b shows the noise-corrupted full synthetic data set exhibiting an eastward increase due to the simulated eastward crustal thinning shown in Fig. 4.4a. The data grid spacing is half the grid spacing of the tesseroid model so that, when separating the training and testing data sets (Fig. 4.3), the training data set points will fall directly above each model element.

We separated the synthetic data into training and testing data sets following Fig. 4.3. The training data set is a regular grid of $N_{lat} \times N_{lon} = 40 \times 50$ points (a total of $N_{train} = 2000$). The testing data set is composed of $N_{test} = 5821$ observations. We used cross-validation to estimate an optimal regularization parameter (μ) from a set of $N_{\mu} = 16$ values equally spaced on a logarithmic scale between 10^{-6} and 10^{-1}. We ran our regularized inversion on the training data set for each value of μ, obtaining 16 Moho depth estimates. For all inversions, the initial Moho depth estimate used to start the Gauss-Newton optimization was set to 60 km depth for all inversion parameters. Furthermore, z_{ref} and $\Delta\rho$ are set to their respective true values. Finally, we computed the mean square error (MSE, Eq. 4.17) for each estimate and chose as the final estimated Moho model the one that minimizes the MSE.

Fig. 4.5a shows the final estimated Moho depth after cross-validation. The recovered model is smooth, indicating that the cross-validation procedure was effective in estimating an optimal regularization parameter. Fig. 4.5b shows difference between the true Moho depth (Fig. 4.4a) and the estimated Moho depth. The differences appear to be semi-randomly distributed with a maximum coinciding with a short-wavelength feature in the true model. The maximum and minimum differences are approximately 2.19 and -2.13 km, respectively. Fig. 4.5c shows inversion residuals, defined as the difference between the observed and predicted data (in mGal). The largest residual (in absolute value) coincides with the largest difference between the true model and the estimate. The inversion residuals are normally distributed, as shown in Fig. 4.5d, with 0.02 mGal mean and a standard deviation of 3.63 mGal. The cross-validation curve in Fig. 4.5e shows a clear minimum MSE at $\mu = 0.00046$ (indicated by the red triangle). Fig. 4.5f shows the convergence of the Gauss-Newton optimization in eight iterations.

We also investigated the computation time spent in each section of the inversion process using a source code profiler. The profiler measures how much time is spent inside each function during the execution of a program. We ran the profiler on a

Table 4.1: Time spent on each function during a single inversion of simple synthetic data. The inversion was performed on a laptop computer with a Intel(R) Core(TM) i7-3612QM CPU @ 2.10GHz processor. The total time for the inversion was 42.133 seconds.

Function description	Time (s)	Percentage of total time (%)
Sparse conjugate gradient	0.021	0.050
Sparse dot product	0.007	0.017
Tesseroid forward modeling	42.059	99.824

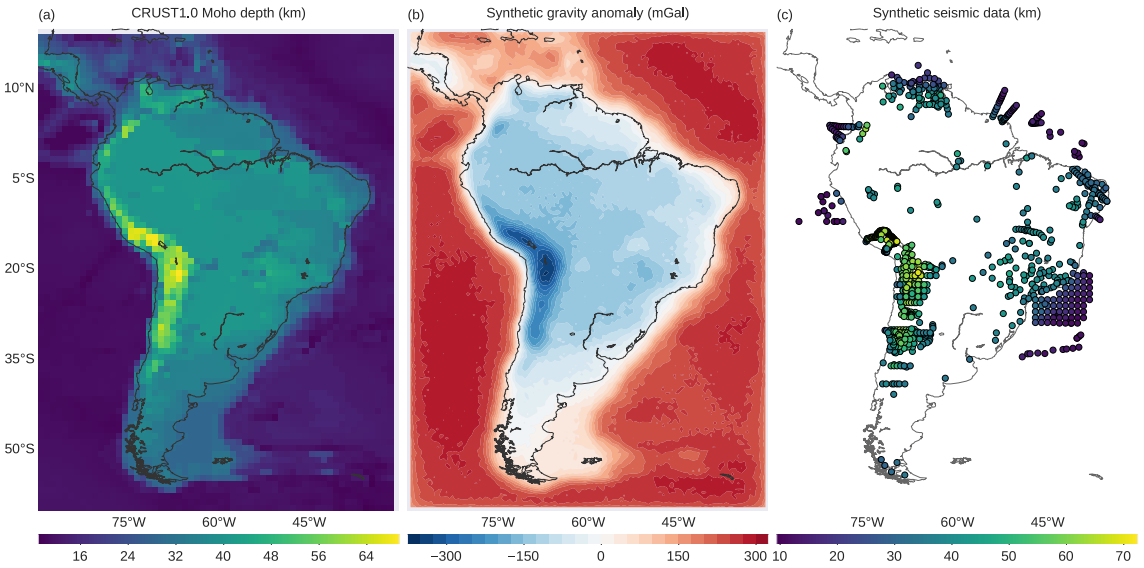

Figure 4.6: Synthetic data of a model derived from CRUST1.0. The model is made of tesseroids with an constant density-contrast of $\Delta\rho = 350\ kg/m^3$ and assuming a reference level of $z_{ref} = 30\ km$. (a) The Moho depth of the model in kilometers. Each pixel in the pseudo-color image corresponds to a tesseroid of the model. (b) Noise-corrupted synthetic gravity data generated from the model. (c) Simulated points where the Moho depths are known from seismological estimates (color dots). Here, these point were obtained by interpolating the Moho depth in (a).

single inversion of the training data set using the estimated regularization parameter. We tracked the total time spent inside each of the three functions that represent the largest computational bottlenecks of the inversion: solving the linear system in Eq. 4.13 using the conjugate gradient method, performing the dot products required to compute the Hessian matrix (Eq. 4.12) and the gradient vector (Eq. 4.11), and forward modeling to calculate the predicted data (Eq. 4.3). The profiling results presented in Table 4.1 show that the time spent on forward modeling accounts for approximately 99.8% of the total computation time.

4.4.2 Model based on CRUST1.0

In this test, we simulate the anomalous Moho of South America using Moho depth information extracted from the CRUST1.0 model (LASKE *et al.*, 2013). We con-

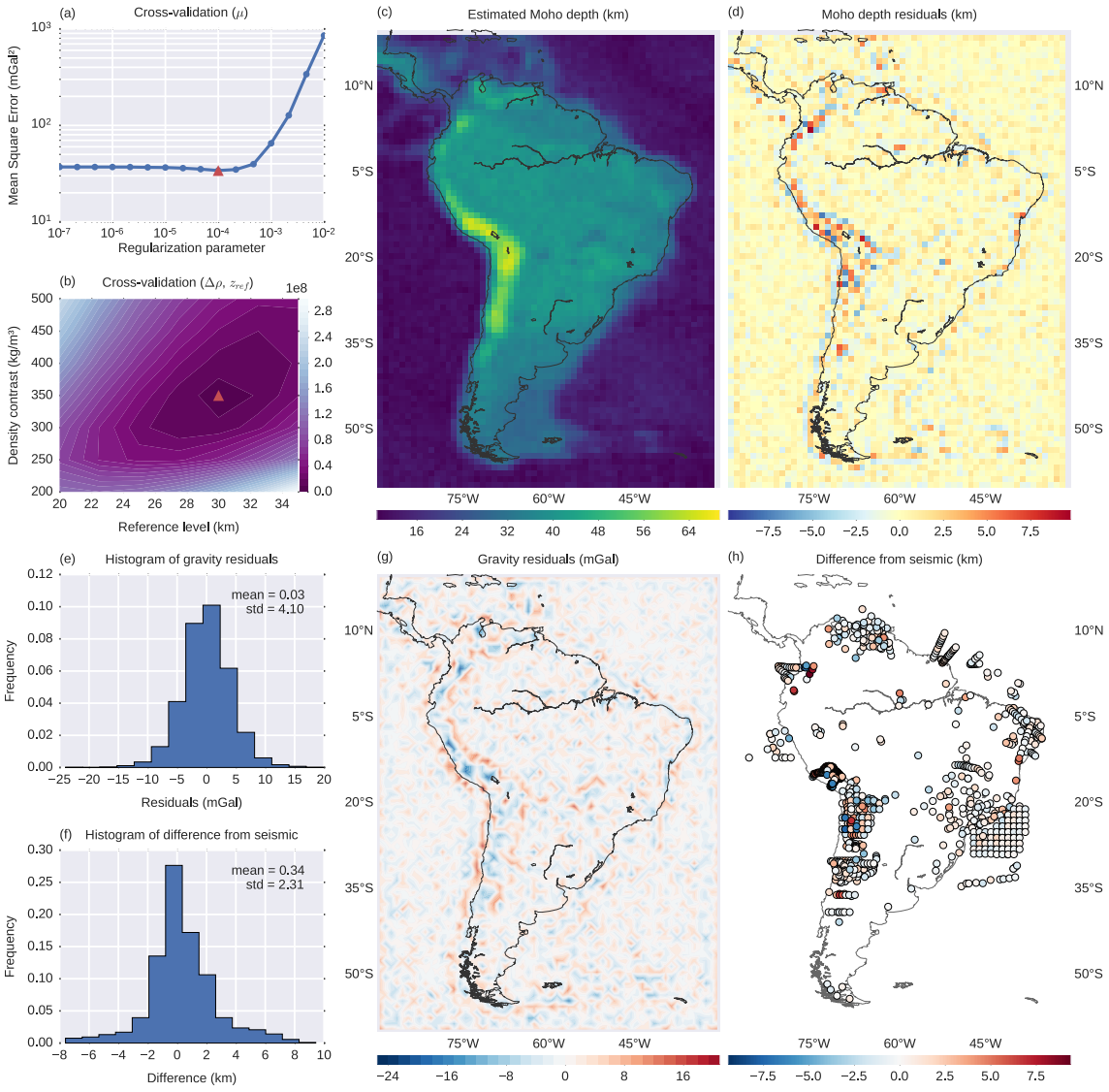

Figure 4.7: Inversion results from the CRUST1.0 synthetic data. (a) Cross-validation curve used to determine the regularization parameter (Eq. 4.10). The minimum Mean Square Error (Eq. 4.17) is found at $\mu = 0.0001$ (red triangle). (b) Cross-validation results used to determine the reference level (z_{ref}) and the density-contrast ($\Delta\rho$). The colored contours represent the Mean Square Error (Eq. 4.18) in km^2. The minimum (red triangle) is found at $z_{ref} = 30\ km$ and $\Delta\rho = 350\ kg/m^3$. (c) The estimated Moho depth. (d) Difference between the CRUST1.0 model depths (Fig. 4.6a) and the estimated depths. (e) Histogram of the inversion residuals (observed minus predicted data). (f) Histogram of the differences between the synthetic seismic observations (Fig. 4.6c) and the estimated depths. (g) The inversion residuals. (h) Difference between the seismic and the estimated depths.

struct a tesseroid model with $M_{lat} \times M_{lon} = 80 \times 60$ juxtaposed elements, 4800 in total, using the Moho depths shown in Fig. 4.6a. In our model, the Normal Earth Moho is $z_{ref} = 30\ km$ and the density-contrast is $\Delta\rho = 350\ kg/m^3$. We produce the synthetic data at a constant height of 50 km and on a regular grid of $N_{lat} \times N_{lon} = 159 \times 119$ points (a total of 18921 observations). We contaminate the synthetic data with normally distributed pseudo-random noise with zero mean and 5 mGal standard deviation (Fig. 4.6b).

The cross-validation procedure to determine $\Delta\rho$ and z_{ref} requires knowledge of the Moho depth at certain points ($\mathbf{z}_s^o$ in Eq. 4.18), usually from seismic experiments. Thus, we must also generate synthetic seismic data about the Moho depth. We produce such data by interpolating the Moho depth shown in Fig. 4.6a on the same 937 geographic coordinates pinpointed in the data set of ASSUMPÇÃO *et al.* (2013). The resulting synthetic seismic data is shown in Fig. 4.6c.

We perform the cross-validation procedures in two parts. First, we run the cross-validation to estimate an optimal regularization parameter (μ). The starting estimate for all inversions is 60 km depth for all model parameters. For this cross-validation, we keep z_{ref} and $\Delta\rho$ fixed to 20 km and 500 kg/m^3, respectively. Second, we use the estimated μ to run the cross-validation to estimate z_{ref} and $\Delta\rho$, thus obtaining the final estimated Moho depths. Fig. 4.7 summarizes the results from both cross-validation runs and the final inversion results.

For the first cross-validation, we separate the synthetic data (Fig. 4.3) into a training set with twice the grid spacing of the original data ($N_{lat} \times N_{lon} = 80 \times 60$) and a testing set with 14,121 observations. We run the inversion for 16 different values of μ equally spaced in a logarithmic scale between 10^{-7} and 10^{-2}. For each of the 16 estimates we compute the MSE (Eq. 4.17), shown in Fig. 4.7a as function of μ. The optimal regularization parameter that minimizes the MSE is $\mu = 10^{-4}$ (red triangle in Fig. 4.7a).

In the second cross-validation, we use the estimated value of μ in all inversions. We test seven values of z_{ref} from 20 to 35 km with 2.5 km intervals and seven values of $\Delta\rho$ from 200 to 500 kg/m^3 with 50 kg/m^3 intervals. We run the inversion for every combination of z_{ref} and $\Delta\rho$, totaling 49 inversions. Finally, we calculate the Mean Square Error (Eq. 4.18) for each of the 49 estimates and choose the values of z_{ref} and $\Delta\rho$ that minimize the MSE. Fig. 4.7b shows a colored-contour map of the MSE with a minimum (marked by the red triangle) at $z_{ref} = 30\ km$ and $\Delta\rho = 350\ kg/m^3$.

Fig. 4.7c shows the final solution after both cross-validation procedures. The recovered model is smooth, indicating that the cross-validation procedure was effective in estimating an optimal regularization parameter. Fig. 4.7d shows the difference between the true Moho depths (Fig. 4.6a) and the estimated depths (Fig. 4.7c). The

maximum and minimum differences are, respectively, 9.8 and -8.2 km. The largest absolute differences are located along the central and northern Andes, where there is a sharp increase in the true Moho depth (Fig. 4.6a). Positive differences (indicating a too shallow estimate) appear along the central portion of the Andes, flanked by regions of negative differences (indicating a too deep estimate) on the continental and Pacific sides. Figs. 4.7e and g show the gravity residuals, defined as the difference between the observed and predicted gravity data. The residuals appear normally distributed, with 0.03 mGal mean and a standard deviation of 4.10 mGal. The gravity residuals follow a similar, though reversed, pattern to the differences shown in Fig. 4.7d. The largest residuals (in absolute value) are along the Andes, with the central portion being dominated by negative residuals and flanked by positive residuals on both sides. Figs. 4.7f and h show the differences between the synthetic seismic data (Fig. 4.6c) and the estimated Moho depths. Once more, the largest differences are concentrated along the Andes, particularly in the central Andes and near Ecuador and Colombia. The differences are smaller along the Atlantic coast of South America, with notable larger differences in a few points of northeastern Brazil and along the Amazon river. In general, large residuals are associated with sharp increases in Moho depth.

4.5 Application to the South American Moho

We apply the inversion method proposed here to invert for the Moho depth of the South American continent. We follow the application of VAN DER MEIJDE *et al.* (2013) but with some differences, mainly using a different data set and performing all modeling in spherical coordinates using tesseroids. The data are corrected of the effects of topography and sedimentary basins. Crust and mantle heterogeneities cannot be properly accounted for in regions where information coverage is sparse and readily accessible models are not available, like in South America and Africa. Hence, for the purposes of this study, we will assume to be negligible all other crustal and mantle sources, including lateral variations in density along the Moho.

4.5.1 Gravity and seismic data

The raw gravity data are generated from the satellite only spherical harmonic model GOCO5S MAYER-GUERR *et al.* (2015). The GOCO5S model combines data from 15 satellites, including the complete mission data from the GOCE satellite. The data were downloaded from the International Centre for Global Earth Models (ICGEM) web-service (BARTHELMES e KÖHLER, 2012, `http://icgem.gfz-potsdam.de/ICGEM/`)) in the form of the complete gravity field on a regular grid with 0.2° grid

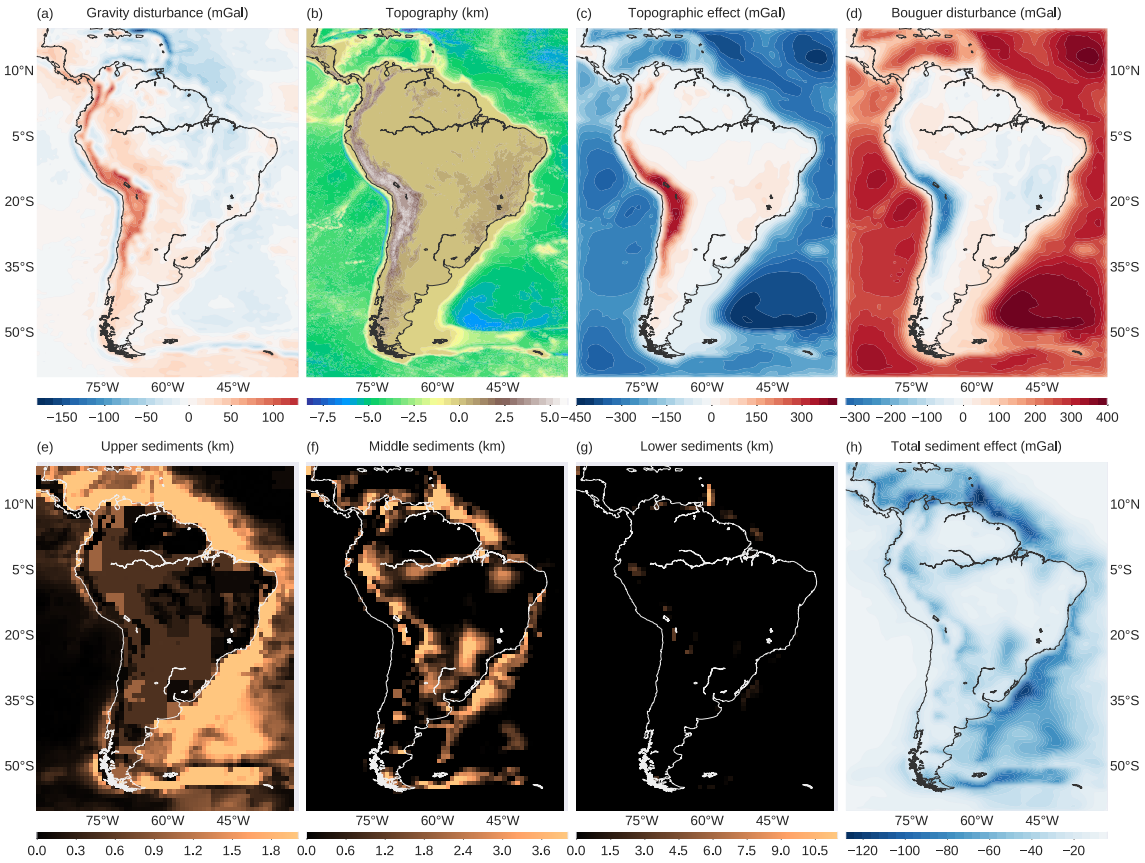

Figure 4.8: Gravity data for South America and the models used in the data corrections. (a) The gravity disturbance (Eq. 4.1) calculated from the raw gravity data. (b) Topography from ETOPO1. (c) Gravitational attraction of the topography calculated at the observation height using tesseroids. (d) The Bouguer disturbance (Eq. 4.2) obtained by subtracting (c) from (a). The upper (e), middle (f), and lower (g) sediment layer thicknesses from the CRUST1.0 model. (h) The total gravitational attraction of the sediment layers shown in (e), (f), and (g), calculated using tesseroids.

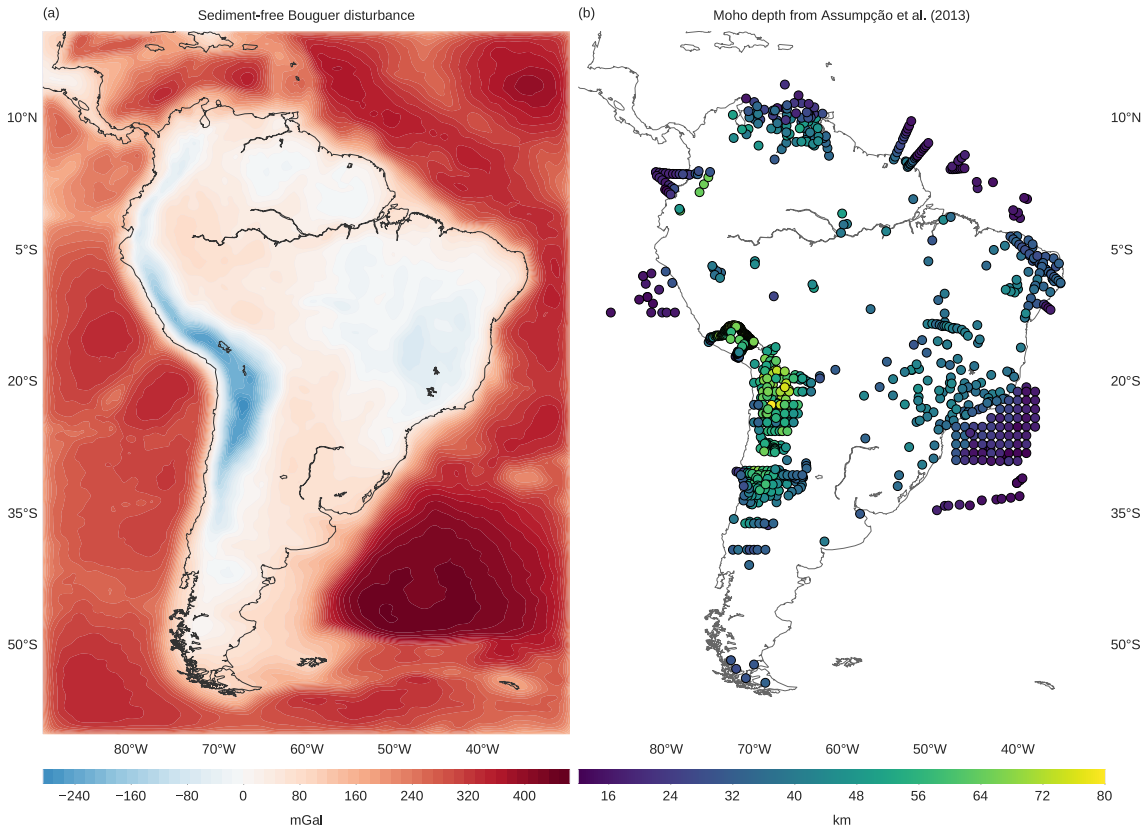

Figure 4.9: Input data for the South American Moho inversion. (a) Sediment-free Bouguer disturbance for South America. Obtained by subtracting the total sediment gravitational effect (Fig. 4.8h) from the Bouguer disturbance (Fig. 4.8d). (b) Seismological Moho depth estimates from ASSUMPÇÃO *et al.* (2013).

spacing at ellipsoidal height 50 km. We calculate the gravity disturbance ($\delta(P)$ in Eq. 4.1) by subtracting from the raw data the normal gravity of the WGS84 reference ellipsoid ($\gamma(P)$) using the formula of LI e GÖTZE (2001). Fig. 4.8a show the calculated gravity disturbance of South America.

We remove the gravitational effect of the topography from the gravity disturbance by modeling the ETOPO1 digital terrain model (AMANTE e EAKINS, 2009, `http://dx.doi.org/10.7289/V5C8276M`) using tesseroids (Fig. 4.8b). We used the standard densities of 2670 kg/m^3 for continents and -1630 kg/m^3 for the oceans. Fig. 4.8c shows the calculated gravitational attraction of the topographic masses at 50 km height. Fig. 4.8d shows the Bouguer disturbance (Eq. 4.2) obtained after subtracting the topographic effect from the gravity disturbance.

The effect of sedimentary basins is removed using tesseroid models of the three sedimentary layers present in the CRUST1.0 model (LASKE *et al.*, 2013, `http://igppweb.ucsd.edu/~gabi/rem.html`). Each sedimentary layer model includes the density of each $1^\circ \times 1^\circ$ model cell. Figs. 4.8e-g show the thickness of the upper, middle, and lower sedimentary layers, respectively. The density-contrasts of the tesseroid model is obtained by subtracting 2670 kg/m^3 from the density of each model element. Fig. 4.8h shows the combined gravitational attraction of the sedimentary basin tesseroid model. We subtract the total effect of sediments from the Bouguer disturbance in Fig. 4.8d to obtain the sediment-free Bouguer disturbance (Fig. 4.9a), which will be used as input for the inversion.

Fig. 4.9b shows the 937 known Moho depths (colored dots) which were estimated from seismological data by ASSUMPÇÃO *et al.* (2013). This data set is used in the cross-validation procedure.

4.5.2 Inversion and cross-validation

As in the CRUST1.0 synthetic data test (section 4.4.2), we perform the cross-validation in two parts. First, we run the cross-validation to estimate an optimal regularization parameter (μ). The starting estimate for all inversions is 60 km depth for all model parameters. For this cross-validation, we keep z_{ref} and $\Delta\rho$ fixed to 20 km and 500 kg/m^3, respectively. Second, we use the estimated μ to run the cross-validation to estimate z_{ref} and $\Delta\rho$, thus obtaining the final estimated Moho depth model.

We split the sediment-free gravity disturbance (Fig. 4.9a) into the training and testing data sets. The training data set is a regular grid with 0.4° grid spacing (twice the spacing of the original data grid) and $N_{lat} \times N_{lon} = 201 \times 151$ grid points, a total of 30,351 observations. The remaining 90,350 points compose the testing data set. We test 16 values of the regularization parameter (μ) equally spaced on a

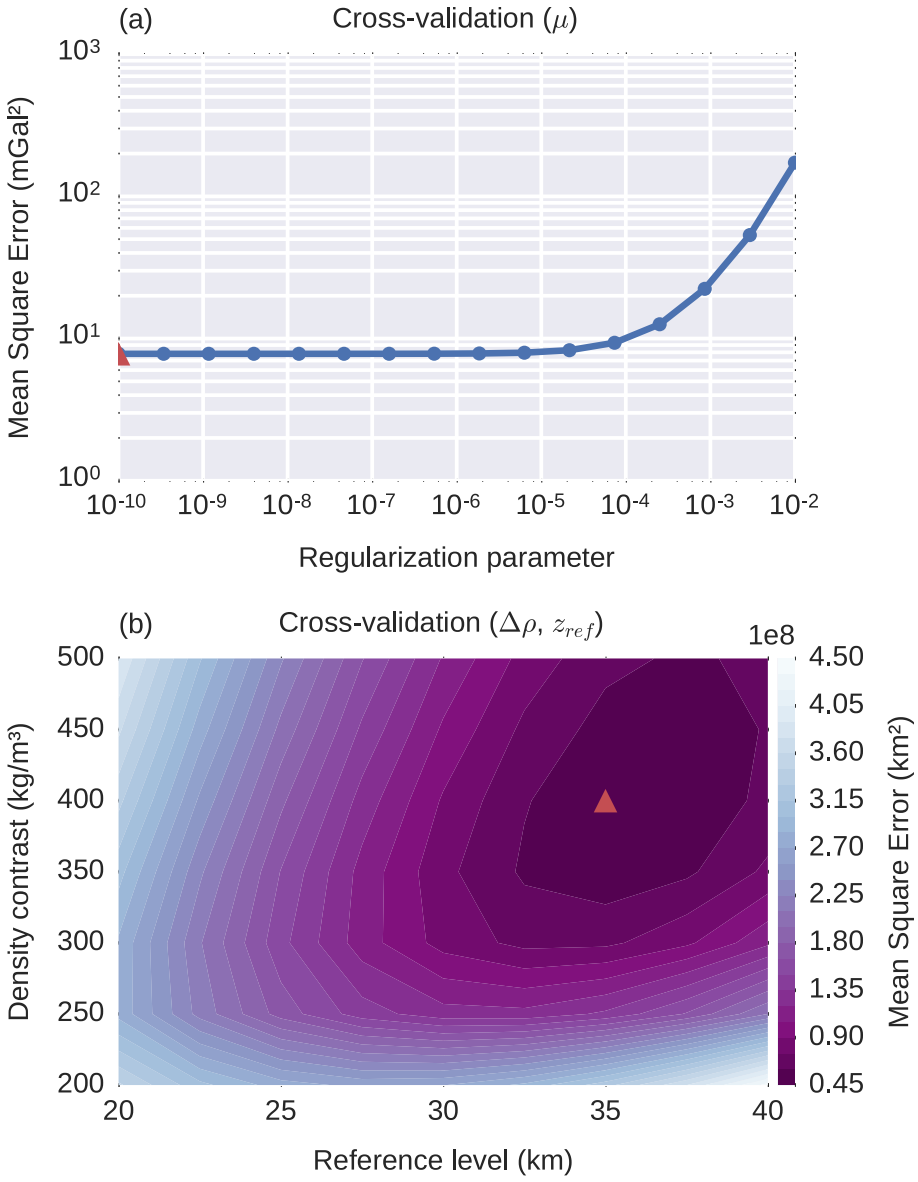

Figure 4.10: Cross-validation results for the South American Moho inversion. (a) Cross-validation to determine the regularization parameter μ (Eq. 4.10). The minimum Mean Square Error (Eq. 4.17), shown as a red triangle, corresponds to $\mu = 10^{-10}$. (b) Cross-validation to determine the reference level (z_{ref}) and the density-contrast ($\Delta\rho$). The colored contours represent the Mean Square Error (Eq. 4.18). The minimum (red triangle) is found at $z_{ref} = 35\ km$ and $\Delta\rho = 400\ kg/m^3$.

logarithmic scale between 10^{-10} and 10^{-2}. Fig. 4.10a shows the Mean Square Error (MSE) as a function of μ. The minimum MSE is found at $\mu = 10^{-10}$, the lowest value of μ tested, suggesting that little or no regularization is required.

We proceed with the second cross-validation using $\mu = 10^{-10}$ in all inversions. We test all combinations of seven values of z_{ref}, from 20 to 40 km with 2.5 km intervals, and seven values of $\Delta\rho$, from 200 to 500 kg/m^3 with 50 kg/m^3 intervals. Fig. 4.10b shows a map of the MSE with respect to the ASSUMPÇÃO *et al.* (2013) data set. The MSE has a well-defined minimum, indicated by the red triangle, at $z_{ref} = 35\ km$ and $\Delta\rho = 400\ kg/m^3$.

4.5.3 Moho model for South America

The final Moho depth model for South America is shown as a pseudo-color map in Fig. 4.11. The model is available in the online repository that accompanies this contribution (see section 4.3.7). Each model element is a $0.4^\circ \times 0.4^\circ$ tesseroid, represented by the pixels in the pseudo-color map.

Our model differs significantly from CRUST1.0 (Fig. 4.6a) but contains most of the large-scale features present in the GMSA12 gravity-derived model of VAN DER MEIJDE *et al.* (2013). The deepest Moho is along the central Andes, reaching depths upward of 70 km. The oceanic areas present the shallowest Moho, ranging approximately from 7.5 to 20 km. The Brazilian and Guyana Shields have a deeper Moho (greater than 35 km), with the deepest portions in the area around the São Francisco Craton and the northern border of the Parecis Basin. The Moho is shallower than 35 km along the Guyana Basin, the Andean foreland basins, the Chaco Basin, and along the centers of the Solimões, Amazonas, and Paraná Basins.

Fig. 4.12a shows the gravity residuals, defined as the difference between the observed and predicted gravity data. Fig. 4.12b shows the differences between the seismic-derived Moho depths of ASSUMPÇÃO *et al.* (2013) (Fig. 4.9b) and the depths of our gravity-derived model (Fig. 4.11). The differences shown in Fig. 4.12b range from approximately -23 to 23 km and have a mean of 1.18 km and a standard deviation of 6.84 km. The gravity residuals and Moho depth differences from seismic are smallest in the oceanic areas, southern Patagonia, and the eastern coast of the continent. The largest gravity residuals are located along the Andes and correlate with the deepest Moho depths. These large residuals follow a pattern of a negative value in the center flanked by positive values to the east and west. This same pattern is observed in the CRUST1.0 synthetic test results (Fig. 4.7). In general, larger gravity residuals appear to be associated with sharp variations in the estimated Moho depth. Along the Andes, large differences with seismic data are correlated with the larger gravity residuals. Conversely, this correlation is absent from the large

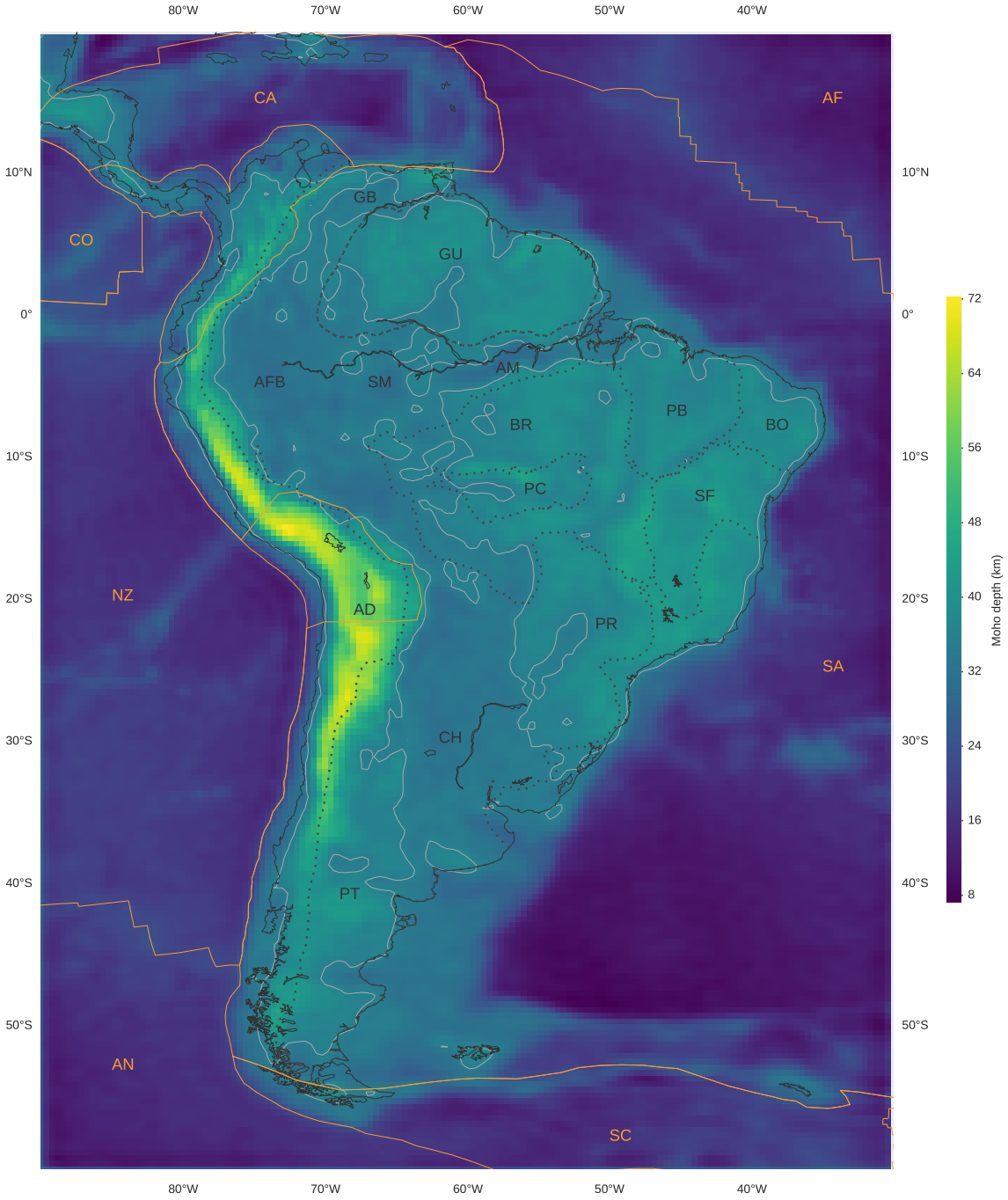

Figure 4.11: The estimated Moho depth of South America. Dotted lines represent the boundaries between major geologic provinces (after ASSUMPÇÃO *et al.*, 2013; GOUTORBE *et al.*, 2015); AD: Andean Province, AFB: Andean foreland basins, AM: Amazonas Basin, BR: Brazilian Shield, BO: Borborema province, CH: Chaco Basin, GB: Guyana Basin, GU: Guyana Shield, PB: Parnaíba Basin, PC: Parecis Basin, PR: Paraná Basin, PT: Patagonia province, SF: São Francisco Craton, SM: Solimões Basin. Solid orange lines mark the limits of the main lithospheric plates (BIRD, 2003); AF: Africa Plate, AN: Antarctica Plate, CA: Caribbean Plate, CO: Cocos Plate, SA: South America Plate, SC: Scotia Plate, NZ: Nazca Plate. The solid light grey line is the 35 km Moho depth contour.

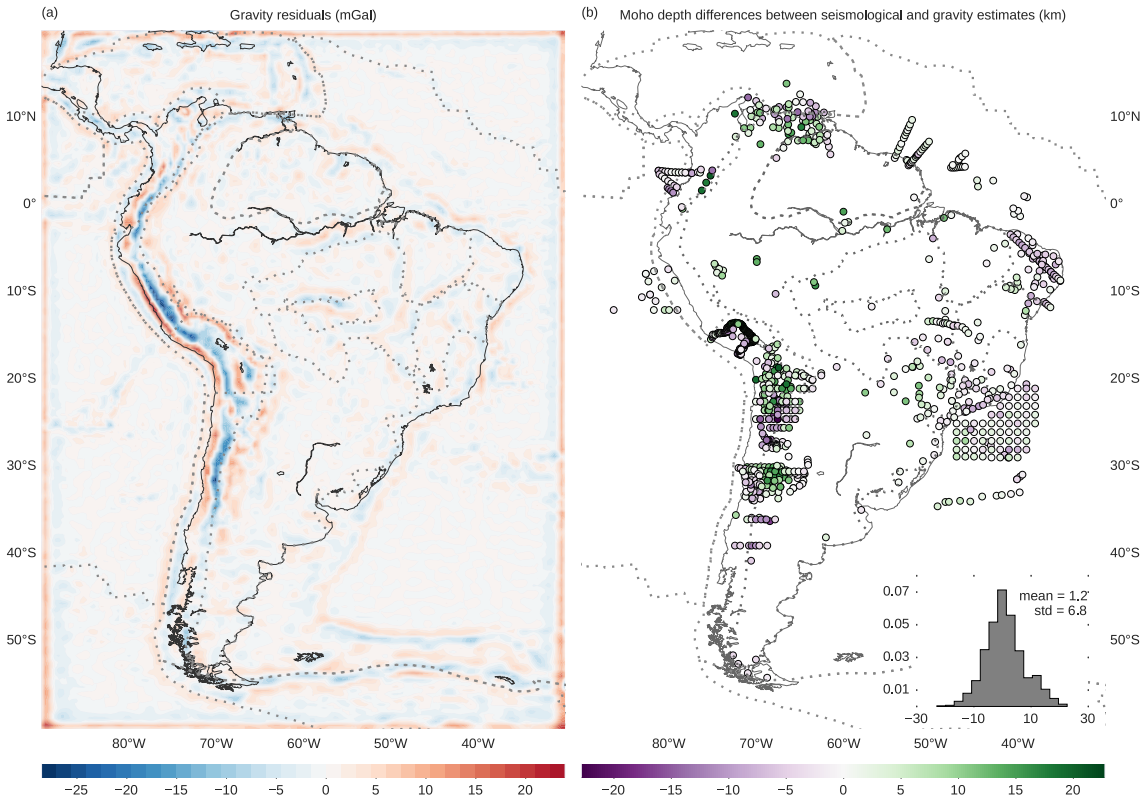

Figure 4.12: Residuals for the estimated South American Moho depth in Fig. 4.11. (a) Gravity residuals, defined as the difference between the observed data in Fig. 4.9a and the data predicted by the estimate in Fig. 4.11. (b) Differences between the seismological depth estimates of ASSUMPÇÃO *et al.* (2013) and our gravity-derived Moho depth estimate. The inset in b shows a histogram of the differences along with their calculated mean and standard deviation (std). Dotted lines mark the limits of major geologic provinces and lithospheric plates.

differences seen in the Guyana, Paraná, and the Solimões Basins. In the Borborema province, northeastern Brazil, our model slightly overestimates the Moho depth. On the other hand, our model underestimates the Moho depths in the Amazonas, Solimões, and Paraná Basins. Particularly in the Amazonas and Solimões Basins, where our model predicts a Moho depth of approximately 30 km, the differences with the seismological estimates can reach 10 km or more.

4.6 Conclusions

We have developed a computationally efficient gravity inversion method in spherical coordinates. Our method extends the Gauss-Newton formulation of Bott's method (SILVA *et al.*, 2014) to use tesseroids as model elements and smoothness regularization. We retain the computational efficiency of Bott's method by taking advantage of the sparse nature of all matrices involved. We employ two cross-validation techniques to estimate the hyper-parameters of the inversion: the regularization parameter, the Moho density-contrast, and the Normal Earth Moho depth.

The test on simple synthetic data shows that our inversion method is able to recover a smooth Moho relief with a homogeneous density-contrast. The inversion was not able to fully recover the shortest wavelength feature in the model, possibly due to the smoothness constraints which tends to soften high-frequency (sharp) variations. The cross-validation Mean Square Error curve in Fig. 4.5e has a well-defined minimum, indicating a value of the regularization parameter (μ) whose corresponding estimate best predicts data that were not included in the inversion. Using this value of μ in the inversion leads to a stable solution characterized by a smooth Moho relief with an acceptable data misfit.

The source code profiling results presented in Table 4.1 confirm the efficiency of the proposed method. When using sparse matrices, solving linear systems and performing matrix multiplications together account for a mere 0.067% of the total computation time required for a single inversion. The majority of the computation time (99.824%) is spent on forward modeling. Thus, we are able to retain the high computational efficiency of Bott's method and use a classic Tikhonov regularization formulation. This approach could, in theory, be extended to other types of regularization (e.g., total variation) and misfit functions (e.g., re-weighted least squares) already available in the literature. For example, the total variation approach used by MARTINS *et al.* (2011) could potentially be implemented in a more straight forward manner than done by SANTOS *et al.* (2015).

The more complex synthetic data test based on CRUST1.0 (Fig. 4.7) shows that the cross-validation using pointwise Moho depth information is able to correctly estimate the density-contrast ($\Delta\rho$) and Normal Earth Moho depth (z_{ref}). This test

indicates that the inversion neither correctly estimates Moho depth nor adequately fits the gravity and pointwise data when sharp variations in Moho depth occur. This phenomenon is particularly strong in the region below the Andes. A likely explanation is that the smoothness regularization is intrinsically unable to produce sharp variations in Moho depth. These effects might be mitigated with the use of sharpness-inducing regularization, like the weighted smoothness inversion (BARBOSA *et al.*, 1999b), Cauchy norm regularization (PILKINGTON, 2008; SACCHI e ULRYCH, 1996), entropic regularization (SILVA *et al.*, 2010), total variation regularization (MARTINS *et al.*, 2011), or an adaptive mixed smoothness-sharpness regularization (SUN e LI, 2014).

We applied the method proposed here to estimate the Moho depth for South America. Our estimated Moho depth model is in accordance with previous results by VAN DER MEIJDE *et al.* (2013). The model fits well the gravity and seismic data in all oceanic regions, the central portion of the Andean foreland, Patagonia, and coastal and central parts of Brazil. However, the model is unable to fit the gravity and seismic data in places with sharp variations in Moho depth, particularly below the Andes and in the boundaries of the main geotectonic provinces of the South American Plate, like the Borborema province, the Parnaiba Basin, and the São Francisco Craton. This might indicate that smoothness regularization should not be applied indiscriminately to the whole model, as suggested by the CRUST1.0 synthetic data test. Another reason for the observed misfit might be the presence of crustal or mantle density anomalies whose gravitational effects were not removed during the data corrections. In the Guyana Basin on the coastal region of Venezuela, along the central Amazonas and Solimões Basins, and in the Paraná Basin, our Moho depth model is able to fit the gravity data but differs significantly from the seismic data. MARIANI *et al.* (2013) and NUNN e AIRES (1988) explain these discrepancies in the Paraná and Amazonas Basins, respectively, as high density rocks in the lower crust. In general, differences between a gravity and a seismically derived Moho model may indicate the presence of crustal or mantle density anomalies that were unaccounted for in the data processing. Such locations warrant further detailed investigation.

4.7 Online repository

The Moho depth model for South America estimated here can be downloaded from the online repository `https://github.com/pinga-lab/paper-moho-inversion-tesseroids`. The repository also contains all data and source code used to produce the results and figures presented here.

Chapter 5

Conclusions

We have developed two open-source software projects that implement the modules required for building an inversion method: forward modeling, optimization, and regularization. We used these modules to develop a fast gravity inversion method to estimate the depth of the crust-mantle interface (the Moho) in a spherical approximation. We then applied the proposed method to estimate the Moho depth for South America.

The *Tesseroids* software is a collection of command-line programs developed in the C programming language. The programs calculate the gravitational potential and its first and second derivatives of a tesseroid (spherical prism) model. We implemented and improved upon an adaptive discretization algorithm to guarantee the accuracy of the computations. The adaptive discretization is controlled by a scalar called the distance-size ratio (D). Higher values of D result in finner discretization and vice-versa. Furthermore, we investigated the accuracy of the calculations as a function of D. Contrary to previous assumptions, our results showed that the first and second derivatives require finner discretization than the gravitational potential to achieve the same accuracy level. The values of the distance-size ratio that yield a maximum error of 0.1% are $D = 1$ for the gravitational potential, $D = 1.5$ for the first derivatives, and $D = 8$ for the second derivatives. These values are included as defaults starting in version 1.2 of the software. Previous versions of *Tesseroids* have been used in published research by, for example, BRAITENBERG *et al.* (2011), ÁLVAREZ *et al.* (2012), BOUMAN *et al.* (2013b), BOUMAN *et al.* (2013a), MARIANI *et al.* (2013), BRAITENBERG (2015), and FULLEA *et al.* (2015).

Fatiando a Terra is a software library implemented in the Python programming language. The library contains functions and classes for data processing, visualization, inversion, and forward modeling. The inverse problems package of the library offers generic classes for optimization and regularization. These classes can be extended and combined with the existing forward modeling functions to implement new inversion methods. Using these tools, the amount of code required to imple-

ment a new method is reduced, increasing the speed of the cycle of prototyping a new algorithm, testing, and then refining it. The project has been used in scholarly works such as CARLOS *et al.* (2014), HIDALGO-GATO e BARBOSA (2015), NICCOLI (2015), OLIVEIRA JR. *et al.* (2015), and BASSETT *et al.* (2016). To date, the project has received contributions from nine developers in three different countries.

We used the tesseroid forward modeling and inverse problem toolkit of *Fatiando a Terra* to implement a new non-linear gravity inversion method to estimate the depth of the Moho. The inversion method uses the Gauss-Newton formulation of Bott's method proposed by SILVA *et al.* (2014). In this formulation, the Jacobian (sensitivity) matrix is approximated by a linear diagonal matrix. This eliminates the high computational cost of calculating the full dense Jacobian matrix for every iteration of the Gauss-Newton method. We stabilize the inverse problem through smoothness regularization. The use of regularization required that we abandon the elimination of linear systems of the traditional Bott's method. However, we maintained computational efficiency by taking advantage of the fact that all matrices involved are sparse. Our benchmarks suggest that less than 0.1% of the computation time required for an inversion is spent on matrix operations and solving linear systems. We also estimate the regularization parameter and two other hyper-parameters of the inversion through cross-validation procedures.

The applications to synthetic data show the efficiency of the proposed inversion method in retrieving smooth Moho depth variations. The cross-validation procedures are able to correctly estimate an optimal regularization parameter, anomalous Moho density-contrast, and the reference level (the depth of the Normal Earth Moho). We applied the proposed method to estimate the Moho depth of the South American continent. The estimated Moho depths are in agreement with previous models and with the major tectonic provinces of the continent. Notable misfits between the observed and predicted gravity data are in the Andes, northern Venezuela, and the Paraná, Solimões, and Amazon Basins. These regions also present a high misfit with Moho depth estimates from seismology. Discrepancies between gravity-derived and seismological estimates of Moho depth may indicate the presence of other density anomalies with the crust or upper mantle. For example, MARIANI *et al.* (2013) and NUNN e AIRES (1988) explain the discrepancies in the Paraná and Amazon Basins, respectively, as high density material in the lower crust.

In conclusion, we have successfully applied the algorithms and software foundation developed here to solve a geophysical inverse problem. This application demonstrates that the same approach can be used to aid the development of new inversion methods in the future. In addition, both software packages were successful in gaining third-party users and have been applied to solve real world scientific problems.

Fatiando a Terra, in particular, was able to gather other developers to contribute source code to the software, enabling the continued growth of the project in the future. We emphasize that this is only possible because the project is open-source and freely available.

Bibliography

ÁLVAREZ, O., GIMENEZ, M., BRAITENBERG, C., et al., 2012, "GOCE satellite derived gravity and gravity gradient corrected for topographic effect in the South Central Andes region: GOCE derivatives in the South Central Andes", *Geophysical Journal International*, v. 190, n. 2 (ago.), pp. 941–959. ISSN: 0956540X. doi: 10.1111/j.1365-246X.2012.05556.x.

AMANTE, C., EAKINS, B. W., 2009, "ETOPO1 1 Arc-Minute Global Relief Model: Procedures, Data Sources and Analysis", *NOAA Technical Memorandum NESDIS NGDC-24. National Geophysical Data Center, NOAA.* doi: 10.7289/V5C8276M.

ASGHARZADEH, M. F., VON FRESE, R. R. B., KIM, H. R., et al., 2007, "Spherical prism gravity effects by Gauss-Legendre quadrature integration", *Geophysical Journal International*, v. 169, n. 1 (abr.), pp. 1–11. ISSN: 0956540X, 1365246X. doi: 10.1111/j.1365-246X.2007.03214.x.

ASSUMPÇÃO, M., FENG, M., TASSARA, A., et al., 2013, "Models of crustal thickness for South America from seismic refraction, receiver functions and surface wave tomography", *Tectonophysics*, v. 609 (dez.), pp. 82–96. ISSN: 0040-1951. doi: 10.1016/j.tecto.2012.11.014.

ASTER, R. C., BORCHERS, B., THURBER, C. H., 2012, *Parameter Estimation and Inverse Problems, Second Edition.* 2 edition ed. Waltham, MA, Academic Press. ISBN: 978-0-12-385048-5.

BACKUS, G. E., GILBERT, J. F., 1967, "Numerical Applications of a Formalism for Geophysical Inverse Problems", *Geophysical Journal International*, v. 13, n. 1-3 (jan.), pp. 247–276. ISSN: 0956-540X, 1365-246X. doi: 10.1111/j.1365-246X.1967.tb02159.x.

BACKUS, G., GILBERT, F., 1968, "The Resolving Power of Gross Earth Data", *Geophysical Journal International*, v. 16, n. 2 (jan.), pp. 169–205. ISSN: 0956-540X, 1365-246X. doi: 10.1111/j.1365-246X.1968.tb00216.x.

BARBOSA, V., SILVA, J., MEDEIROS, W., 1999a, "Stable inversion of gravity anomalies of sedimentary basins with nonsmooth basement reliefs and arbitrary density contrast variations", *GEOPHYSICS*, v. 64, n. 3 (maio), pp. 754–764. ISSN: 0016-8033. doi: 10.1190/1.1444585.

BARBOSA, V. C. F., SILVA, J. B. C., MEDEIROS, W. E., 1999b, "Gravity inversion of a discontinuous relief stabilized by weighted smoothness constraints on depth", *GEOPHYSICS*, v. 64, n. 5 (set.), pp. 1429–1437. ISSN: 0016-8033, 1942-2156. doi: 10.1190/1.1444647.

BARNES, G., BARRAUD, J., 2012, "Imaging geologic surfaces by inverting gravity gradient data with depth horizons", *GEOPHYSICS*, v. 77, n. 1 (jan.), pp. G1–G11. ISSN: 0016-8033, 1942-2156. doi: 10.1190/geo2011-0149.1.

BARRERA-FIGUEROA, V., SOSA-PEDROZA, J., LÓPEZ-BONILLA, J., 2006, "Multiple root finder algorithm for Legendre and Chebyshev polynomials via Newton's method". In: *Annales Mathematicae et Informaticae*, v. 33, pp. 3–13.

BARTHELMES, F., KÖHLER, W., 2012, "International Centre for Global Earth Models (ICGEM)", *Journal of Geodesy*, v. 86, n. 10 (set.), pp. 932–934. ISSN: 0949-7714, 1432-1394. doi: 10.1007/s00190-012-0584-1.

BASSETT, D., SANDWELL, D. T., FIALKO, Y., et al., 2016, "Upper-plate controls on co-seismic slip in the 2011 magnitude 9.0 Tohoku-oki earthquake", *Nature*, v. 531, n. 7592 (mar.), pp. 92–96. ISSN: 0028-0836, 1476-4687. doi: 10.1038/nature16945.

BEHNEL, S., BRADSHAW, R., CITRO, C., et al., 2011, "Cython: The Best of Both Worlds", *Computing in Science Engineering*, v. 13, n. 2, pp. 31–39. ISSN: 1521-9615. doi: 10.1109/MCSE.2010.118.

BIRD, P., 2003, "An updated digital model of plate boundaries", *Geochemistry, Geophysics, Geosystems*, v. 4, n. 3 (mar.), pp. 1027. ISSN: 1525-2027. doi: 10.1029/2001GC000252.

BOTT, M. H. P., 1960, "The use of Rapid Digital Computing Methods for Direct Gravity Interpretation of Sedimentary Basins", *Geophysical Journal International*, v. 3, n. 1 (jan.), pp. 63–67. ISSN: 0956-540X, 1365-246X. doi: 10.1111/j.1365-246X.1960.tb00065.x.

BOUMAN, J., EBBING, J., FUCHS, M., 2013a, "Reference frame transformation of satellite gravity gradients and topographic mass reduction", *Journal of*

Geophysical Research: Solid Earth, v. 118, n. 2 (fev.), pp. 759–774. ISSN: 2169-9356. doi: 10.1029/2012JB009747.

BOUMAN, J., EBBING, J., MEEKES, S., et al., 2013b, "GOCE gravity gradient data for lithospheric modeling", *International Journal of Applied Earth Observation and Geoinformation*, v. 35, pp. 16–30. ISSN: 0303-2434. doi: 10.1016/j.jag.2013.11.001.

BRAITENBERG, C., 2015, "Exploration of tectonic structures with GOCE in Africa and across-continents", *International Journal of Applied Earth Observation and Geoinformation*, v. 35, Part A (mar.), pp. 88–95. ISSN: 0303-2434. doi: 10.1016/j.jag.2014.01.013.

BRAITENBERG, C., MARIANI, P., EBBING, J., et al., 2011, "The enigmatic Chad lineament revisited with global gravity and gravity-gradient fields", *Geological Society, London, Special Publications*, v. 357, n. 1 (jan.), pp. 329–341. ISSN: 0305-8719, 2041-4927. doi: 10.1144/SP357.18.

CARLOS, D. U., UIEDA, L., BARBOSA, V. C. F., 2014, "Imaging iron ore from the Quadrilátero Ferrífero (Brazil) using geophysical inversion and drill hole data", *Ore Geology Reviews*, v. 61 (set.), pp. 268–285. ISSN: 0169-1368. doi: 10.1016/j.oregeorev.2014.02.011.

CHEN, J. L., WILSON, C. R., TAPLEY, B. D., 2011, "Interannual variability of Greenland ice losses from satellite gravimetry", *Journal of Geophysical Research: Solid Earth*, v. 116, n. B7 (jul.), pp. B07406. ISSN: 2156-2202. doi: 10.1029/2010JB007789.

FARQUHARSON, C. G., OLDENBURG, D. W., 2004, "A comparison of automatic techniques for estimating the regularization parameter in non-linear inverse problems", *Geophysical Journal International*, v. 156, n. 3 (jan.), pp. 411–425. ISSN: 0956-540X, 1365-246X. doi: 10.1111/j.1365-246X.2004.02190.x.

FEDI, M., PILKINGTON, M., 2012, "Understanding imaging methods for potential field data", *GEOPHYSICS*, v. 77, n. 1 (jan.), pp. G13–G24. ISSN: 0016-8033. doi: 10.1190/geo2011-0078.1.

FULLEA, J., RODRÍGUEZ-GONZÁLEZ, J., CHARCO, M., et al., 2015, "Perturbing effects of sub-lithospheric mass anomalies in GOCE gravity gradient and other gravity data modelling: Application to the Atlantic-Mediterranean transition zone", *International Journal of Applied Earth*

Observation and Geoinformation, v. 35 (mar.), pp. 54–69. ISSN: 0303-2434. doi: 10.1016/j.jag.2014.02.003.

GORDON, A. C., MOHRIAK, W. U., BARBOSA, V. C. F., 2013, "Crustal architecture of the Almada Basin, NE Brazil: an example of a non-volcanic rift segment of the South Atlantic passive margin", *Geological Society, London, Special Publications*, v. 369, n. 1 (jan.), pp. 215–234. ISSN: 0305-8719, 2041-4927. doi: 10.1144/SP369.1.

GOUTORBE, B., COELHO, D. L. D. O., DROUET, S., 2015, "Rayleigh wave group velocities at periods of 6–23 s across Brazil from ambient noise tomography", *Geophysical Journal International*, v. 203, n. 2 (jan.), pp. 869–882. ISSN: 0956-540X, 1365-246X. doi: 10.1093/gji/ggv343.

GROMBEIN, T., SEITZ, K., HECK, B., 2013, "Optimized formulas for the gravitational field of a tesseroid", *Journal of Geodesy*, v. 87, n. 7 (jul.), pp. 645–660. ISSN: 0949-7714, 1432-1394. doi: 10.1007/s00190-013-0636-1.

HANSEN, P., 1992, "Analysis of Discrete Ill-Posed Problems by Means of the L-Curve", *SIAM Review*, v. 34, n. 4 (dez.), pp. 561–580. ISSN: 0036-1445. doi: 10.1137/1034115.

HECK, B., SEITZ, K., 2007, "A comparison of the tesseroid, prism and point-mass approaches for mass reductions in gravity field modelling", *Journal of Geodesy*, v. 81, n. 2 (fev.), pp. 121–136. ISSN: 0949-7714, 1432-1394. doi: 10.1007/s00190-006-0094-0.

HIDALGO-GATO, M., BARBOSA, V., 2015, "Edge detection of potential-field sources using scale-space monogenic signal: Fundamental principles", *GEOPHYSICS*, v. 80, n. 5 (jul.), pp. J27–J36. ISSN: 0016-8033. doi: 10.1190/geo2015-0025.1.

HILDEBRAND, F. B., 1987, *Introduction to Numerical Analysis*. New York, NY, USA, Dover Publications. ISBN: 978-0-486-65363-1.

HUMPHREY, V., GUDMUNDSSON, L., SENEVIRATNE, S. I., 2016, "Assessing Global Water Storage Variability from GRACE: Trends, Seasonal Cycle, Subseasonal Anomalies and Extremes", *Surveys in Geophysics*, (fev.), pp. 1–39. ISSN: 0169-3298, 1573-0956. doi: 10.1007/s10712-016-9367-1.

HUNTER, J. D., 2007, "Matplotlib: A 2D graphics environment", *Computing in Science & Engineering*, v. 9, n. 3, pp. 90–95. doi: 10.1109/MCSE.2007.55.

JONES, E., OLIPHANT, T., PETERSON, P., et al., 2001. "SciPy: Open source scientific tools for Python". `http://www.scipy.org/`. Accessed August 2015.

KELLEY, C. T., 1987, *Iterative Methods for Optimization*. 1 edition ed. Philadelphia, Society for Industrial and Applied Mathematics. ISBN: 978-0-89871-433-3.

KIM, J.-H., 2009, "Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap", *Computational Statistics & Data Analysis*, v. 53, n. 11 (set.), pp. 3735–3745. ISSN: 0167-9473. doi: 10.1016/j.csda.2009.04.009.

KU, C. C., 1977, "A direct computation of gravity and magnetic anomalies caused by 2-and 3-dimensional bodies of arbitrary shape and arbitrary magnetic polarization by equivalent-point method and a simplified cubic spline", *Geophysics*, v. 42, n. 3, pp. 610–622. doi: 10.1190/1.1440732.

LAFEHR, T., 1991, "An exact solution for the gravity curvature (Bullard B) correction", *GEOPHYSICS*, v. 56, n. 8 (ago.), pp. 1179–1184. ISSN: 0016-8033. doi: 10.1190/1.1443138.

LASKE, G., MASTERS, G., MA, Z., et al., 2013, "Update on CRUST1.0 - A 1-degree Global Model of Earth's Crust". In: *EGU General Assembly Conference Abstracts*, v. 15, pp. EGU2013–2658.

LEÃO, J., MENEZES, P., BELTRÃO, J., et al., 1996, "Gravity inversion of basement relief constrained by the knowledge of depth at isolated points", *GEOPHYSICS*, v. 61, n. 6 (nov.), pp. 1702–1714. ISSN: 0016-8033. doi: 10.1190/1.1444088.

LI, X., GÖTZE, H., 2001, "Ellipsoid, geoid, gravity, geodesy, and geophysics", *GEOPHYSICS*, v. 66, n. 6 (nov.), pp. 1660–1668. ISSN: 0016-8033. doi: 10.1190/1.1487109.

LI, Z., HAO, T., XU, Y., et al., 2011, "An efficient and adaptive approach for modeling gravity effects in spherical coordinates", *Journal of Applied Geophysics*, v. 73, n. 3 (mar.), pp. 221–231. ISSN: 09269851. doi: 10.1016/j.jappgeo.2011.01.004.

MADAGASCAR DEVELOPMENT TEAM, 2013. "Madagascar". `http://www.ahay.org`. Accessed May 2013.

MARIANI, P., BRAITENBERG, C., USSAMI, N., 2013, "Explaining the thick crust in Paraná basin, Brazil, with satellite GOCE gravity observations", *Journal of South American Earth Sciences*, v. 45 (ago.), pp. 209–223. ISSN: 08959811. doi: 10.1016/j.jsames.2013.03.008.

MARTINS, C., BARBOSA, V., SILVA, J., 2010, "Simultaneous 3D depth-to-basement and density-contrast estimates using gravity data and depth control at few points", *GEOPHYSICS*, v. 75, n. 3 (maio), pp. I21–I28. ISSN: 0016-8033. doi: 10.1190/1.3380225.

MARTINS, C., LIMA, W., BARBOSA, V., et al., 2011, "Total variation regularization for depth-to-basement estimate: Part 1 — Mathematical details and applications", *GEOPHYSICS*, v. 76, n. 1 (jan.), pp. I1–I12. ISSN: 0016-8033. doi: 10.1190/1.3524286.

MAYER-GUERR, T., PAIL, R., GRUBER, T., et al., 2015, "The combined satellite gravity field model GOCO05s". In: *EGU General Assembly Conference Abstracts*, v. 17, pp. EGU2015–12364, abr.

MCKINNEY, W., 2010, "Data Structures for Statistical Computing in Python". pp. 51–56.

MENKE, W., 1984, *Geophysical Data Analysis: Discrete Inverse Theory*. San Diego, California, Academic Press. ISBN: 978-0-08-050732-3.

MIKHAILOV, V. O., PANET, I., HAYN, M., et al., 2014, "Comparative study of temporal variations in the earth's gravity field using GRACE gravity models in the regions of three recent giant earthquakes", *Izvestiya, Physics of the Solid Earth*, v. 50, n. 2 (mar.), pp. 177–191. ISSN: 1069-3513, 1555-6506. doi: 10.1134/S1069351314020062.

MIKUŠKA, J., PAŠTEKA, R., MARUŠIAK, I., 2006, "Estimation of distant relief effect in gravimetry", *GEOPHYSICS*, v. 71, n. 6 (out.), pp. J59–J69. ISSN: 0016-8033. doi: 10.1190/1.2338333.

MILNER, K., BECKER, T. W., BOSCHI, L., et al., 2009, "New Software Framework to Share Research Tools", *Eos, Transactions American Geophysical Union*, v. 90, n. 12 (mar.), pp. 104–104. ISSN: 2324-9250. doi: 10.1029/2009EO120005.

NICCOLI, M., 2015, "Mapping and validating lineaments", *The Leading Edge*, v. 34, n. 8 (jul.), pp. 948–950. ISSN: 1070-485X. doi: 10.1190/tle34080948.1.

NUNN, J. A., AIRES, J. R., 1988, "Gravity anomalies and flexure of the lithosphere at the Middle Amazon Basin, Brazil", *Journal of Geophysical Research*, v. 93, n. B1, pp. 415. ISSN: 0148-0227. doi: 10.1029/JB093iB01p00415.

OLDENBURG, D., 1974, "The inversion and interpretation of gravity anomalies", *GEOPHYSICS*, v. 39, n. 4 (ago.), pp. 526–536. ISSN: 0016-8033. doi: 10.1190/1.1440444.

OLIVEIRA JR., V. C., SALES, D. P., BARBOSA, V. C. F., et al., 2015, "Estimation of the total magnetization direction of approximately spherical bodies", *Nonlin. Processes Geophys.*, v. 22, n. 2 (abr.), pp. 215–232. ISSN: 1607-7946. doi: 10.5194/npg-22-215-2015.

PANISOVA, J., FRAŠTIA, M., WUNDERLICH, T., et al., 2013, "Microgravity and Ground-penetrating Radar Investigations of Subsurface Features at the St Catherine's Monastery, Slovakia", *Archaeological Prospection*, v. 20, n. 3 (jul.), pp. 163–174. ISSN: 1099-0763. doi: 10.1002/arp.1450.

PARKER, R. L., 1973, "The Rapid Calculation of Potential Anomalies", *Geophysical Journal International*, v. 31, n. 4 (jan.), pp. 447–455. ISSN: 0956-540X, 1365-246X. doi: 10.1111/j.1365-246X.1973.tb06513.x.

PEDERSEN, L., 1991, "Relations between potential fields and some equivalent sources", *GEOPHYSICS*, v. 56, n. 7 (jul.), pp. 961–971. ISSN: 0016-8033. doi: 10.1190/1.1443129.

PÉREZ, F., GRANGER, B. E., 2007, "IPython: A System for Interactive Scientific Computing", *Computing in Science & Engineering*, v. 9, n. 3 (maio), pp. 21–29. ISSN: 1521-9615. doi: 10.1109/MCSE.2007.53.

PILKINGTON, M., 2008, "3D magnetic data-space inversion with sparseness constraints", *GEOPHYSICS*, v. 74, n. 1 (dez.), pp. L7–L15. ISSN: 0016-8033. doi: 10.1190/1.3026538.

PLOUFF, D., 1976, "Gravity and magnetic fields of polygonal prisms and application to magnetic terrain corrections", *GEOPHYSICS*, v. 41, n. 4 (ago.), pp. 727–741. ISSN: 0016-8033. doi: 10.1190/1.1440645.

RAMACHANDRAN, P., VAROQUAUX, G., 2011, "Mayavi: 3D visualization of scientific data", *Computing in Science & Engineering*, v. 13, n. 2, pp. 40–51. doi: 10.1109/MCSE.2011.35.

RAMILLIEN, G., LOMBARD, A., CAZENAVE, A., et al., 2006, "Interannual variations of the mass balance of the Antarctica and Greenland ice sheets

from GRACE", *Global and Planetary Change*, v. 53, n. 3 (set.), pp. 198–208. ISSN: 0921-8181. doi: 10.1016/j.gloplacha.2006.06.003.

REGUZZONI, M., SAMPIETRO, D., SANSO, F., 2013, "Global Moho from the combination of the CRUST2.0 model and GOCE data", *Geophysical Journal International*, (jul.). ISSN: 0956-540X, 1365-246X. doi: 10.1093/gji/ggt247.

SACCHI, M. D., ULRYCH, T. J., 1996, "Estimation of the discrete Fourier transform, a linear inversion approach", *Geophysics*, v. 61, n. 4, pp. 1128–1136.

SANTOS, D., SILVA, J., MARTINS, C., et al., 2015, "Efficient gravity inversion of discontinuous basement relief", *GEOPHYSICS*, (maio), pp. G95–G106. ISSN: 0016-8033. doi: 10.1190/geo2014-0513.1.

SILVA, J., MEDEIROS, W., BARBOSA, V., 2001, "Potential-field inversion: Choosing the appropriate technique to solve a geologic problem", *GEOPHYSICS*, v. 66, n. 2 (mar.), pp. 511–520. ISSN: 0016-8033. doi: 10.1190/1.1444941.

SILVA, J., COSTA, D., BARBOSA, V., 2006, "Gravity inversion of basement relief and estimation of density contrast variation with depth", *GEOPHYSICS*, v. 71, n. 5 (set.), pp. J51–J58. ISSN: 0016-8033. doi: 10.1190/1.2236383.

SILVA, J., OLIVEIRA, A., BARBOSA, V., 2010, "Gravity inversion of 2D basement relief using entropic regularization", *GEOPHYSICS*, v. 75, n. 3 (maio), pp. I29–I35. ISSN: 0016-8033. doi: 10.1190/1.3374358.

SILVA, J., SANTOS, D., GOMES, K., 2014, "Fast gravity inversion of basement relief", *Geophysics*, v. 79, n. 5 (ago.), pp. G79–G91. ISSN: 0016-8033. doi: 10.1190/geo2014-0024.1.

STOCKWELL JR., J. W., 1999, "The CWP/SU: Seismic Un$_*$x package1,11", *Computers & Geosciences*, v. 25, n. 4 (maio), pp. 415–419. ISSN: 0098-3004. doi: 10.1016/S0098-3004(98)00145-9.

SUN, J., LI, Y., 2014, "Adaptive Lp inversion for simultaneous recovery of both blocky and smooth features in a geophysical model", *Geophysical Journal International*, (mar.), pp. ggu067. ISSN: 0956-540X, 1365-246X. doi: 10.1093/gji/ggu067.

TIKHONOV, A. N., ARSENIN, V. I., 1977, *Solutions of ill-posed problems*. Scripta series in mathematics. Washington : New York, Winston. ISBN: 978-0-470-99124-4.

UIEDA, L., 2015. "A tesserioid (spherical prism) in a geocentric coordinate system with a local-North-oriented coordinate system". http://dx.doi.org/10.6084/m9.figshare.1495525. Accessed November 2015.

UIEDA, L., BARBOSA, V. C. F., 2012, "Robust 3D gravity gradient inversion by planting anomalous densities", *Geophysics*, v. 77, n. 4, pp. G55–G66. ISSN: 00168033. doi: 10.1190/geo2011-0388.1.

UIEDA, L., JR, V. C. O., BARBOSA, V. C. F., 2013a. "Code samples in "Modeling the Earth with Fatiando a Terra"". http://dx.doi.org/10.6084/m9.figshare.708390, a. Accessed May 2013.

UIEDA, L., OLIVEIRA JR, V. C., BARBOSA, V. C. F., 2013b, "Modeling the Earth with Fatiando a Terra". In: van der Walt, S., Millman, J., Huff, K. (Eds.), *Proceedings of the 12th Python in Science Conference*, pp. 91 – 98, b.

UIEDA, L., BARBOSA, V. C. F., BRAITENBERG, C., 2016, "Tesseroids: forward modeling gravitational fields in spherical coordinates", *Geophysics*. In press (accepted).

VAN DER MEIJDE, M., JULIÀ, J., ASSUMPÇÃO, M., 2013, "Gravity derived Moho for South America", *Tectonophysics*, v. 609 (mar.), pp. 456–467. ISSN: 00401951. doi: 10.1016/j.tecto.2013.03.023.

VAN DER MEIJDE, M., FADEL, I., DITMAR, P., et al., 2015, "Uncertainties in crustal thickness models for data sparse environments: A review for South America and Africa", *Journal of Geodynamics*, v. 84 (mar.), pp. 1–18. ISSN: 0264-3707. doi: 10.1016/j.jog.2014.09.013.

WASKOM, M., BOTVINNIK, O., HOBSON, P., et al., 2015. "seaborn: v0.6.0". http://dx.doi.org/10.5281/zenodo.19108, jun. Accessed November 2015.

WESSEL, P., SMITH, W. H. F., 1991, "Free software helps map and display data", *Eos, Transactions American Geophysical Union*, v. 72, n. 41 (out.), pp. 441–446. ISSN: 2324-9250. doi: 10.1029/90EO00319.

WIECZOREK, M. A., PHILLIPS, R. J., 1998, "Potential anomalies on a sphere: Applications to the thickness of the lunar crust", *Journal of Geophysical Research: Planets*, v. 103, n. E1 (jan.), pp. 1715–1724. ISSN: 2156-2202. doi: 10.1029/97JE03136.

WILD-PFEIFFER, F., 2008, "A comparison of different mass elements for use in gravity gradiometry", *Journal of Geodesy*, v. 82, n. 10 (abr.), pp. 637–653. ISSN: 0949-7714, 1432-1394. doi: 10.1007/s00190-008-0219-8.