

3D Gravity Modeling in Gocad

Rapport Projet/Stage

Grosser Beleg

Realized at the Ecole Nationale Supérieure de Géologie Nancy
with the Laboratoire Informatique et Analyse des Données
(LIAD)

Kristina Brottko

5 October 2000

Abstract

In this paper gravity modeling algorithms in Gocad, a geomodeling software, are presented. Two different approaches have been followed: modeling in the space domain, and modeling in the frequency domain. Object of study has been the impact structure of Chicxulub/Mexico, which has been modeled using the developed algorithm. Inversion algorithms have been tested for the space domain with a matrix inversion method and with a new method called Discrete Smooth Interpolation.

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 3 |
| 1.1 | The Gocad Software | 3 |
| 1.2 | Introduction to Gravity | 3 |
| 1.3 | Motivation of the Work | 5 |
| 2 | Gravity Modeling in the Space Domain | 6 |
| 2.1 | Algorithm of Subsurface Parameterization | 6 |
| 2.2 | Realization in Gocad | 6 |
| 2.3 | Case Study - The Impact Structure of Chicxulub | 9 |
| 2.3.1 | Building the Model | 9 |
| 2.3.2 | The Calculations | 9 |
| 2.3.3 | Results | 12 |
| 2.4 | Search of a Inversion Method | 12 |
| 2.4.1 | Inversion with Inverse Matrix | 13 |
| 2.4.2 | Inversion with the DSI Algorithm | 19 |
| 3 | Gravity Modeling in the Frequency Domain | 26 |
| 3.1 | Derivation of Parkers Formula | 26 |
| 3.2 | Realization in Gocad | 26 |
| 3.3 | Examples and Problems | 27 |
| 4 | Conclusions | 30 |
| A | Classes and Functions | 31 |
| | List of Figures | 37 |
| | References | 38 |

Chapter 1

Introduction

1.1 The Gocad Software

The name Gocad comes from 'Geological Objects Computer Aided Design'. This software offers the possibility to model geological objects (like lines, surfaces, bodies and their features) in 3D space. Gocad has been developed within the Gocad Research Program which was initiated by the Computer Science Department of the Ecole Nationale Supérieure de Géologie Nancy (ENSG) in 1989. The fields of usage include modeling in structural geology, reservoir engineering, geophysics as well as basin modeling for petroleum geology.

Gocad is written in C++ and uses Motif and X-Window as user interface. A number of classes are used to construct, visualize and edit a model. Some of them are used in this work and will be explained when necessary.

The Gocad software offers the possibility to adapt the facilities to the needs of the user. A development environment called parallel tree makes it possible for developers to write new applications reusing all the Gocad programming environment (configuration, make files) and extending the standard Gocad resources without working inside the Gocad development source tree.

1.2 Introduction to Gravity

In gravity surveying, subsurface geology is investigated on the basis of variations in the earth's gravitational field generated by differences of density between subsurface rocks. The concept is the idea of a causative body, which is a rock unit of different density from its surroundings. A causative body represents a subsurface zone of anomalous mass and causes a localized perturbation in the gravitational field that is the gravity anomaly. Many geological situations cause zones of anomalous mass that produce significant gravity anomalies. On a small scale these are buried bedrock or even man made cavities. On a larger scale, negative anomalies are caused by salt domes. Granite plutons or sedimentary basins can generate anomalies on an even larger scale. The interpretation of gravity anomalies allows an estimation of the

depth and shape of the causative body.

The basis of the gravity survey method is Newtons Law of Gravitation

$$F = \frac{\gamma m_1 m_2}{r^2} \quad (1.1)$$

where γ is the Gravitational Constant ($6.67e^{-11} \frac{m^3}{kg s^2}$). The gravitational attraction of a point mass at distance r in direction of the mass is given by

$$\Delta g_r = \frac{\gamma m}{r^2} \quad (1.2)$$

Only the vertical component is measured, thus the equation becomes

$$\Delta g_z = \frac{\gamma m}{r^2} \cos\theta \quad (1.3)$$

or

$$\Delta g_z = \frac{\gamma m z}{r^3} \quad (1.4)$$

The gravity anomaly of any body can be calculated by dividing it into small mass elements. The mass of this small element is given by (where ρ is the density) $\delta m = \rho \delta x' \delta y' \delta z'$. So the attraction of one element at a point outside the mass can be calculated by

$$\delta g = \gamma \rho \frac{(z' - z)}{r^3} \delta x' \delta y' \delta z' \quad (1.5)$$

After summation of all mass elements

$$\Delta g = \Sigma \Sigma \Sigma \gamma \rho \frac{(z - z')}{r^3} \delta x' \delta y' \delta z' \quad (1.6)$$

This becomes an integral when the dimensions approach zero

$$\Delta g = \int \int \int \gamma \rho \frac{(z' - z)}{r^3} dx' dy' dz' \quad (1.7)$$

Modelling comprises the computation of the gravity anomaly of an assumed causative body. By comparison of the so obtained anomaly with the actual observed anomaly conclusions concerning the subsurface can be made. Starting from 1.5 the gravity attraction of a point mass can be calculated. A volume element can be considered as a point mass if its dimensions are small compared with the distance to the point where the measurement is taken.

In general, the gravity anomaly of any body can be determined by summarizing attractions of all the mass elements which make up the body. For simply shaped bodies equation 1.7 can be solved. If we have irregular shaped bodies a numerical solution of 1.6 is suitable.

1.3 Motivation of the Work

Direct interpretation of geophysical surveys with potential field methods is often restricted to simple geometrical models. The indirect approach or modeling aims to find the geology by a trial and error method. Different models are calculated and compared to the real survey. In times of powerful computers the solution of the forward problem of complex geology can be obtained fast and conveniently.

The aim of this work was to provide tools for the modeling of potential field anomalies. All the programming and modeling in this project has been done with the Gocad-1.4.6. version on a Sun workstation.

Chapter 2

Gravity Modeling in the Space Domain

2.1 Algorithm of Subsurface Parameterization

As explained in the previous section, the gravity anomaly of any body can be calculated by subdividing it into small mass elements and summing over all elements (see equation 1.6). This applies also for larger scale geology. Any kind of three dimensional grid would be suitable, the easiest to handle is a parallelepipedic grid with constant cell volume. The density in each cell is constant, with the cell far enough from the reading point to be considered as point mass, the attraction of one cell is given by

$$\Delta g_z = \frac{\gamma V_{cell} \Delta \rho z}{r^3} \quad (2.1)$$

where V is the volume, z the depth and r the distance between reading point and center of cell.

The question arising is when can a cell be said far enough to be considered a point mass. In the first place it depends on the dimension of the cell, it should be small compared with the distance. In the second place it is a question of the general scale of the problem and the required precision. Cells that are too near or too big must be subdivided until they fit the conditions as pointed out.

2.2 Realization in Gocad

Gocad offers two possibilities to construct three dimensional grids: the Voxet and the SGrid (stratigraphic grid). The SGrid is well adapted to fit into geological structures as they occur in reservoir modeling. Its disadvantages are additional geometric information which make computing slow. Therefore the Voxet class has been chosen. A Voxet is a regular parallelepipedic volume which is defined by three vectors $\vec{u} \vec{v} \vec{w}$ and the number of cells along this axes nu, nv, nw. The three vectors need not to be parallel to the (x,y,z) space, they need

not to be orthogonal and can have different lengths. Each cell of the voxel can store floating point values in different properties. With this the subsurface for many geological situations can be modeled.

In practice the user will construct the voxel corresponding to his needs and initialize a density property. Furthermore the points at which the anomaly is calculated have to be given. They do not need to be regular spaced so any object of the class VSet (vertex set) can be used. Two other parameters concerning the point mass consideration are asked for. First is a parameter called "distance parameter". It defines at which ratio between distance and dimension the cube can no longer be considered a point mass and must be subdivided. This might lead to an infinite algorithm at cells near the surface, so a second parameter is introduced defining the minimum size at which the cube can be splitted up. It is called "minimum splitup size". Figure 2.1 shows a voxel which cells near the surface are split.

The most simple example is shown in figure 2.2 a sphere with a density contrast of $1000 \frac{kg}{m^3}$.

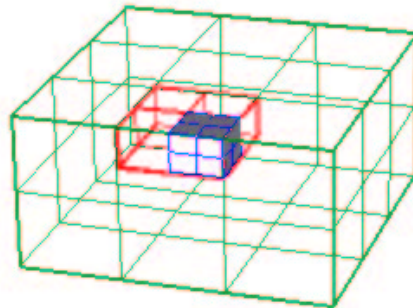


Figure 2.1: Subdivided voxel

The cube size in the voxel is 1m by 1m by 1m. There are 20 cubes in each direction. In light blue is shown the calculated response and in dark blue the well known analytic results of a sphere with diameter of 10m and the same density contrast. The maximum value corresponds to 3.4 gu. All discrepancies are due to the inadequate approximation of a sphere by small cubes. A bigger case study of the Chicxulub impact structure will be presented in the next section.

Complex geology Applying the algorithm as described in the last section will not lead to the right results when regarding a complex geology. As shown in figure 2.3 reading points near the border will suffer from border effects that is the lack of cubes at the border. A slightly different algorithm will be used, first the indices of the surrounding cubes are determined afterwards their attractions are summed. Two more parameters are introduced; `cube_u` and `cube_v`. This are the numbers of cubes in u direction and v direction which the user wants to be taken into account.

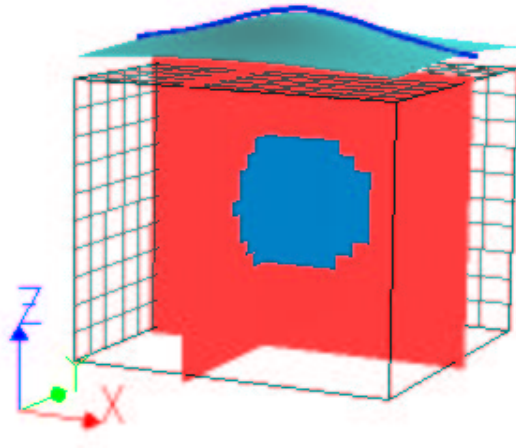


Figure 2.2: Simple sphere model

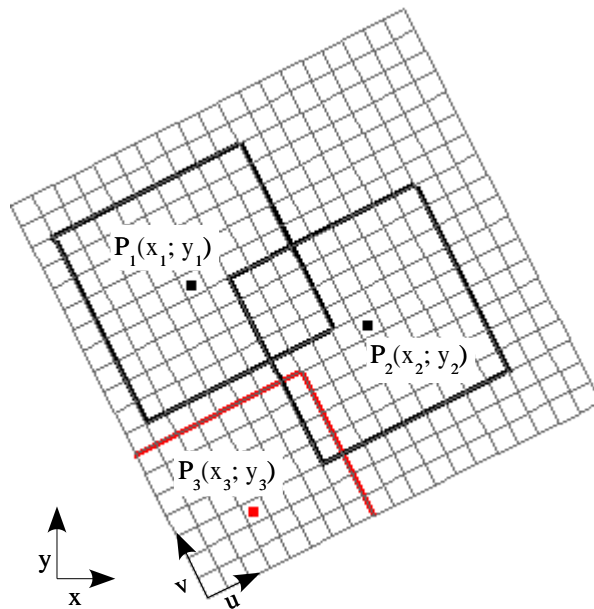


Figure 2.3: Grid

2.3 Case Study - The Impact Structure of Chicxulub

2.3.1 Building the Model

The buried geological structure of Chicxulub (Northern Yucatan, Mexico) has first been explained as an impact crater by Penfield and Camargo (1981). Recent studies strongly support this interpretation by providing geophysical and geochemical evidence. The Chicxulub impact has been linked to the mass extinction at the end of the Cretaceous period. Since the crater is partly situated offshore gravity data is unevenly distributed and has been interpolated in large areas. In addition seismic measurements have been made covering the circular anomaly.

Sharpton(1993)[8] first proposed a conceptual model for the buried structure based on gravity data. According to this model, the Chicxulub impact structure corresponds to a 300 km diameter multi-ring basin. Further modelling work based on a 2 1/2 dimensional model has been done by Espindola(1995)[3]. He constrains the diameter of the crater to 200 km. The density model here proposed follows this interpretation. This means, the structure corresponds to a crater with a inner peak of uplifted basement material and only one ring.

Available data Around the observed anomaly low eleven wells have been drilled. Their depths range between 200m and 3000m. Cores have been produced and analyzed. The location of the wells and geology are shown in figure 2.4.

First model First step in constructing the model is the choice of the area of interest. According to the size of the area an object of the Gocad class voxet is built. In the case of Chicxulub the voxet has an east-west extension of 500km (75 cells), a north-south extension of 600km (85 cells) and a depth of 14km (50 cells). A property "density" is created. To outline the different facies surfaces are constructed which define the geological structures and fit into available data like well markers. An example is shown in figure 2.5, the blue surface approximates the basement and the red one the crater structure. To initialize the density Gocad offers different possibilities. Here regions between surfaces have been created and assigned constant densities. An example for the region which presents the first estimate for the carbonate sediments is shown in figure 2.6.

2.3.2 The Calculations

Input parameters The calculations have been carried out with the algorithm described in section 2.2. In figure 2.7 an example of the input window is given. The parameters are

- Name of the voxet with density information
- Name of the density property
- Name of the pointset

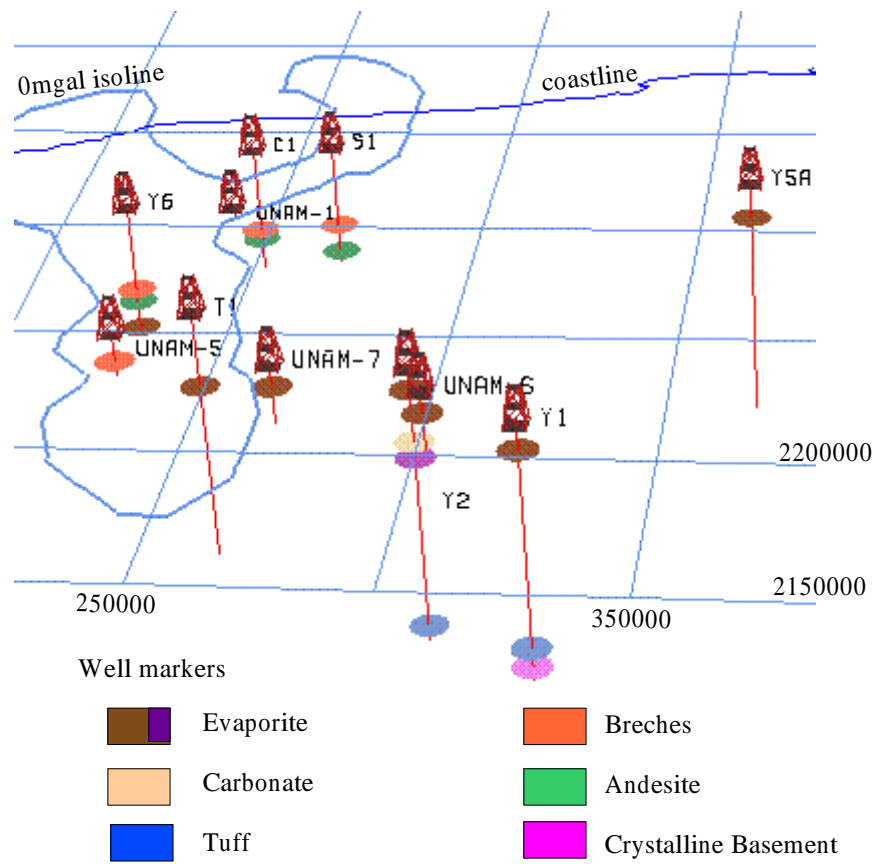


Figure 2.4: Wells in the Chicxulub area, z axis 20 times exaggerated

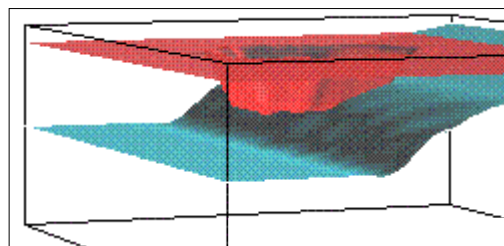


Figure 2.5: Two surfaces outlining geological structures, z axis 15 times exaggerated

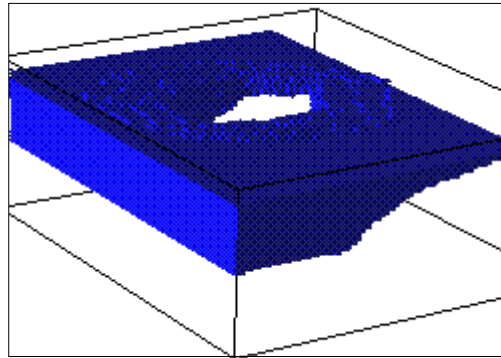


Figure 2.6: Region initialized between two surfaces, z axis 15 times exaggerated

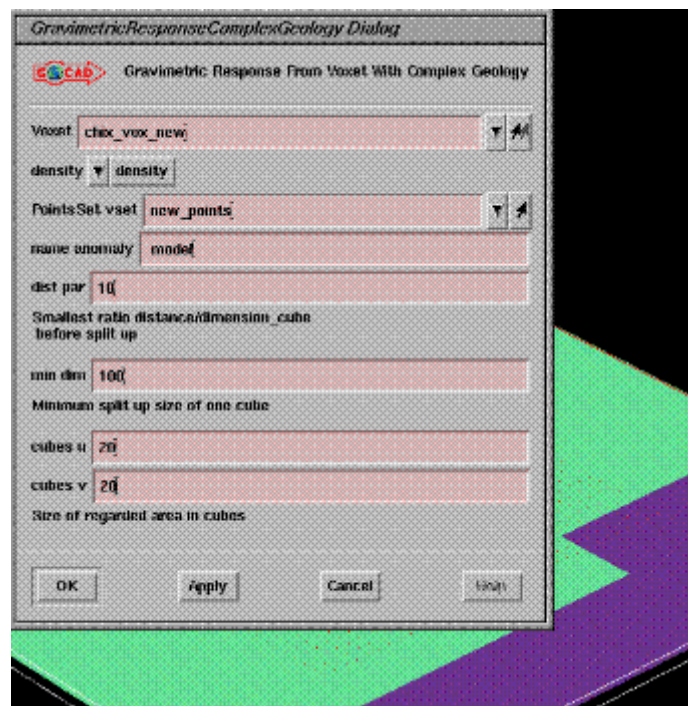


Figure 2.7: Interface for the calculation of gravity anomaly

- Name of the property for calculated anomaly
- "Distance parameter" (see section 2.2)
- "Minimum split up size" (see section 2.2)
- Number of cubes in u-direction (see section 2.2)
- Number of cubes in v-direction (see section 2.2)

As already mentioned in the last section the used voxet had a dimension of 75 by 85 by 50 cells. The anomaly was calculated on a pointset with 824 points. As distance parameter

10 was chosen and minimum split up size was 100 meters. In u and v direction 30 cubes respectively have been taken into account for each observation point. This equals an area of 44100 km^2 .

Changing the model Gocad offers many possibilities to access and change property values. Here only some few can be mentioned. A fast access offers a script. Scripts are small programmes in a C-like syntax which are interpreted at run time. They can be applied to the whole object or to specified regions. A typical syntax could be:

```
{if(U>20) density=100;}
```

which would apply a density of 100 $\frac{kg}{m^3}$ to all cubes with u-index superior to 20.

Surfaces can be altered and fit to new data easily. From this regions can always be reinitialized. The property value of one cube can also be accessed manually and the value can be changed by mouse click.

2.3.3 Results

In this section the results of the modeling are presented. In some cases they are not satisfying yet, but firstly a model is never a final one, it should be updated as further data are available. And the second point is the time consuming process of modeling which did not allow to test hundreds of models.

The geology of Chicxulub are mainly marine sediments which overlay a crystalline basement. A general view of the densities and geological features is given in figure 2.8. More detailed information on three profiles are given in figure 2.9. The calculated and measured gravity values are given as well. A diameter of 150 km was obtained for the crater. A comparison of the isoline maps of the anomalies is shown in figure 2.10. Especially in the northwest and the southeast large discrepancies are observed.

2.4 Search of a Inversion Method

The inversion aims to determine the parameters of the causative body such as density, shape, depth etc. in an iterative process. Generally it involves four steps:

1. Construction of a reasonable model.
2. Computation of its gravity responds.
3. Comparison of computed model with observation.
4. Alteration of model to improve correspondence and return to step 2.

Solution of step 4 depends on the chosen approach. Alteration can be performed manually or in the case of complex bodies automatically. To achieve this non-linear optimization is

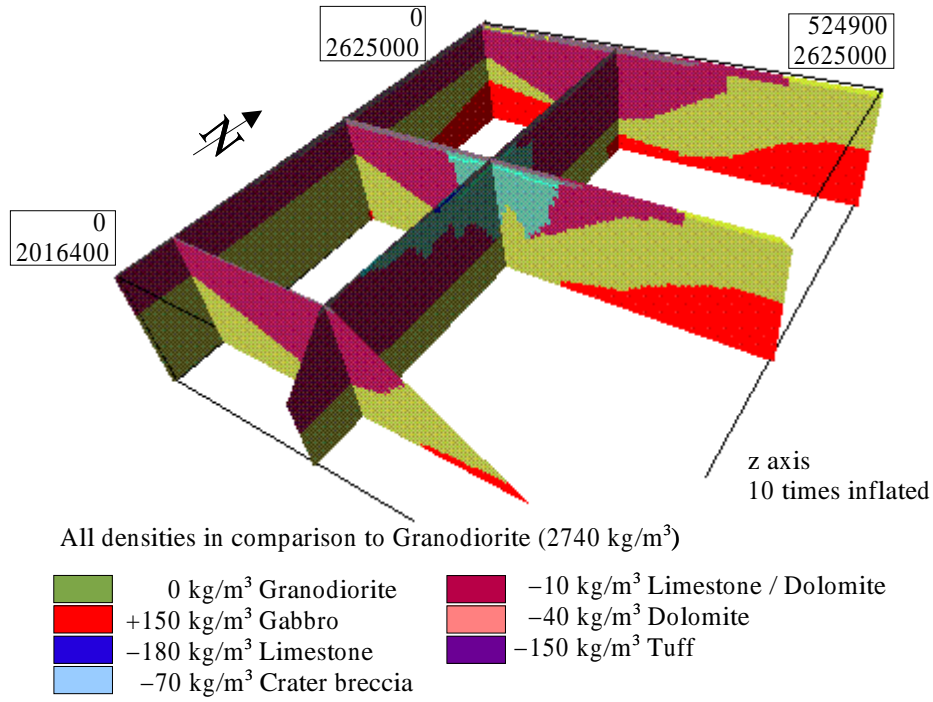


Figure 2.8: Overview of obtained densities in Chicxulub

used. All variables may be allowed to vary within defined limits. The method then attempts to minimize a function F with

$$F = \sum_{i=1}^n (\Delta g_{obs_i} - \Delta g_{calc_i})^2 \quad (2.2)$$

In the next section a method which makes use of an inverse matrix algorithm will be explained. In section 2.4.2 another approach called Discrete Smooth Interpolation (DSI) will be presented.

2.4.1 Inversion with Inverse Matrix

A model response can be either a linear or a nonlinear function of the model parameters. Recalling equation 2.1

$$\Delta g_z = \frac{\gamma V_{cell} \Delta \rho z}{r^3}$$

one can see that Δg_z is a linear function of the parameter density (ρ) and a nonlinear function of the distance parameter \vec{r}^3 . In the case of subparametrization the earth in small cubes the geometry is assumed fixed so the result will depend only on the density distribution. As we have already seen this is a linear problem so linear least-square inversion can be applied. The strategy is to minimize the sum of squares of the errors between the model response and the observations (see equation 2.2). The set of the α observations is represented by the vector

$$\vec{g}_z = col(g_{z_1}, g_{z_2}, \dots, g_{z_\alpha}) \quad (2.3)$$

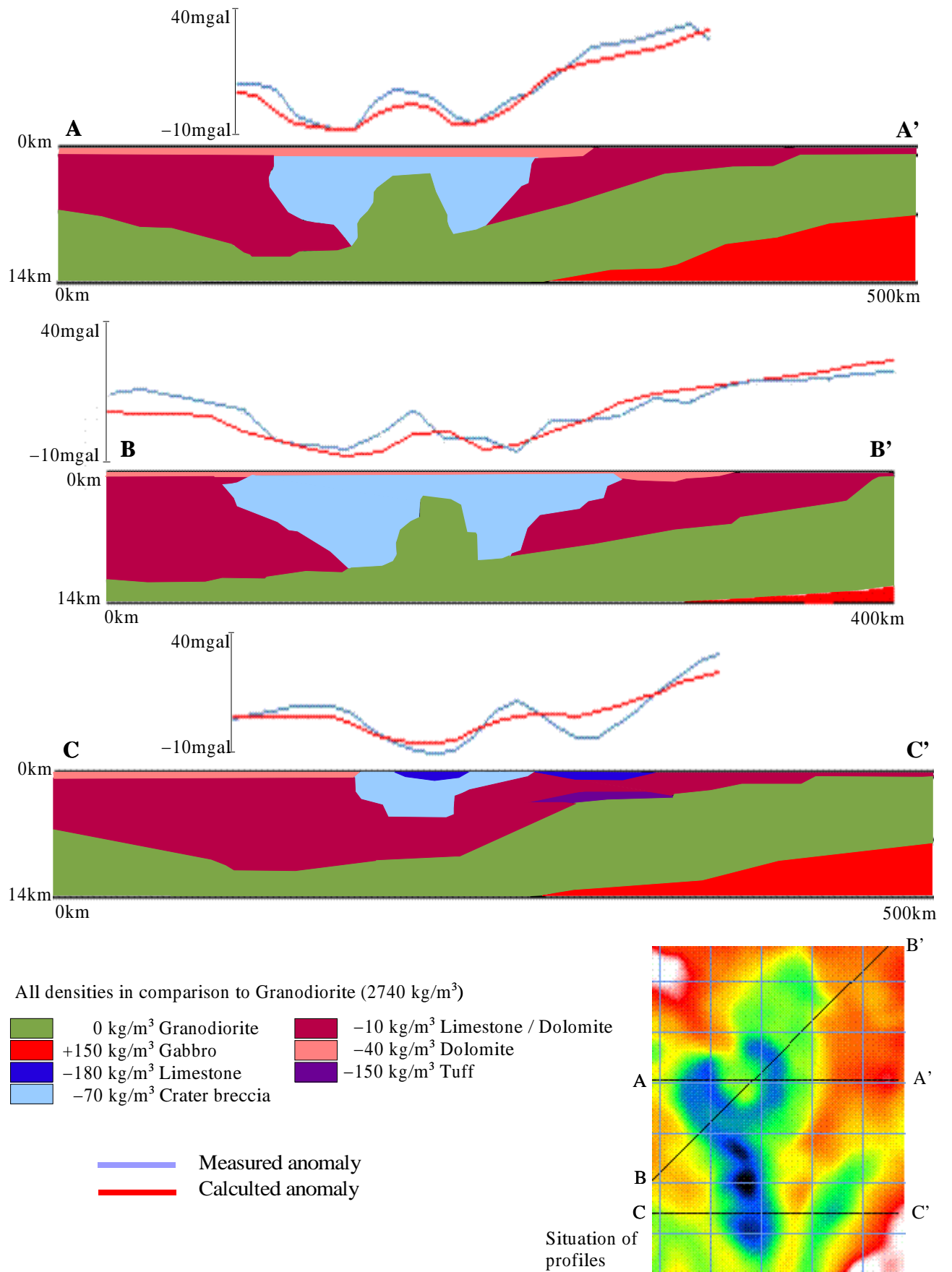


Figure 2.9: Three profiles over the Chicxulub area

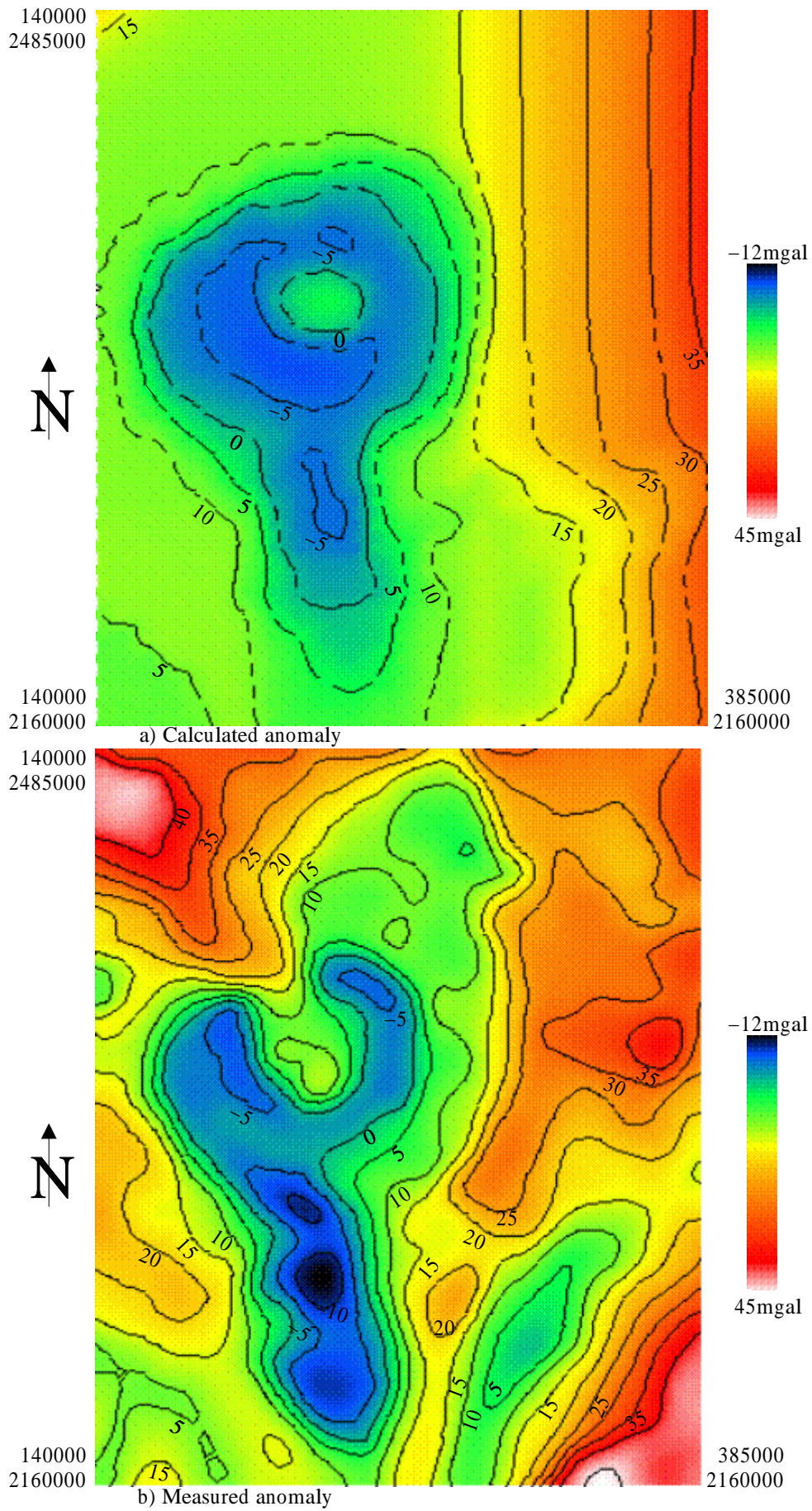


Figure 2.10: Results of modeled anomaly and measured anomaly

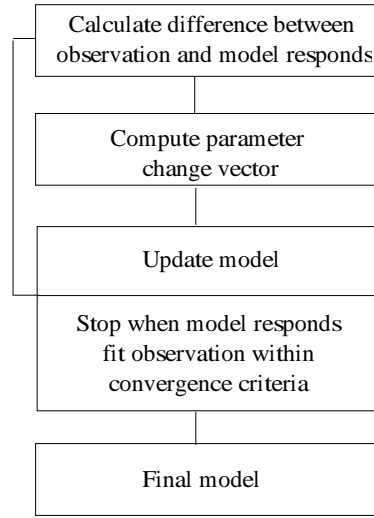


Figure 2.11: Workflow of inversion with inverse matrix

The response of the model is presented by a vector of the same length

$$\vec{f}_z = \text{col}(f_{z_1}, f_{z_2}, \dots, f_{z_\alpha}) \quad (2.4)$$

The model is a function of the density distribution. There are n parameters, corresponding to the n cubes in the model. This can be represented in a parameter vector θ

$$\vec{\theta} = \text{col}(\theta_1, \theta_2, \dots, \theta_n) \quad (2.5)$$

If we have initial parameters $\vec{\theta}^0$ and a initial model response \vec{f}_z^0 then the perturbation of the model response can be represented in matrix notation

$$\vec{f}_z = \vec{f}_z^0 + \mathcal{Z}\delta \quad (2.6)$$

where \mathcal{Z} is the $\alpha \times n$ Jacobien matrix of partial derivates with elements

$$Z_{ij} = \frac{\partial f_i}{\partial \theta_j} \quad (2.7)$$

and $\vec{\delta} = \vec{\theta} - \vec{\theta}_0$ is the parameter change vector. This is the vector we are looking for but the choice must be made so that the error will be minimized. If $\vec{e} = \vec{g}_z - \vec{f}_z$ is the difference between model and observation and combining it with 2.6 gives $\vec{g}_z - \vec{f}_z^0 = \mathcal{Z}\vec{\delta} + \vec{e}$. The vector $\vec{g}_z - \vec{f}_z^0$ is called discrepancy vector \vec{d} . In the least square approach the cumulative error $S = \vec{e}^T \vec{e}$ needs to be minimized with respect to δ . So we have

$$\frac{\partial}{\partial \vec{\delta}} (\vec{\delta}^T \mathcal{Z}^T \mathcal{Z} \vec{\delta} - \vec{d}^T \mathcal{Z} \vec{\delta} - \vec{\delta}^T \mathcal{Z}^T \vec{d} + \vec{d}^T \vec{d}) = 0. \quad (2.8)$$

After carrying out the differentiation the solution for the parameter change vector is obtained

$$\mathcal{Z}^T \mathcal{Z} \vec{\delta} = \mathcal{Z}^T \vec{d} \quad (2.9)$$

Unfortunately obtaining the final solution involves matrix inversion. This can cause numeric inaccuracies for large datasets. So it is better to deal with

$$\mathcal{Z}\vec{\delta} = \vec{d} \quad (2.10)$$

The inverse of a matrix is only defined if the matrix is square. But there is a definition for generalized inverse matrix and it can be found with singular value decomposition.

Singular value decomposition SVD The algorithm of SVD factors a matrix \mathcal{Z} into a product of three matrices.

$$\mathcal{Z} = \mathcal{U}\mathcal{W}\mathcal{V}^T \quad (2.11)$$

With \mathcal{Z} a $\alpha \times n$ matrix (more observations than parameters), \mathcal{U} is an column-orthogonal $\alpha \times n$ matrix, \mathcal{V}^T is the transposed of a orthogonal $n \times n$ matrix and \mathcal{W} is a $n \times n$ diagonal matrix which contains the singular values. Then the inverse of any matrix \mathcal{Z} is defined as

$$\mathcal{Z}^{-1} = \mathcal{V}\mathcal{W}^{-1}\mathcal{U}^T \quad (2.12)$$

With this an explicite solution for the parameter change vector can be found

$$\vec{\delta} = \mathcal{V}\mathcal{W}^{-1}\mathcal{U}^T\vec{d} \quad (2.13)$$

Realization Equation 2.13 has been implemented using an algorithm for the SVD presented in "Numerical Recipies in C" [7]. The workflow of the programme is shown in figure 2.11. The inversion is an iterative process which can oscillate strongly. To avoid this it must be damped, this can be achieved by editing the eigenvalues in \mathcal{W} . Small eigenvalues can be set to zero or a damping factor may be added.

Example So far the algorithm has only be tested with synthetic data which will be described here. The model consists of eight cubes with the size of 100x100x50m and have a uniform density of $100\frac{kg}{m^3}$. The gravity anomaly has been calculated using the algorithm described in section 2.2 with a distance parameter of 5 and a minimum split-up size of 10 meters. As an assumed starting model the same model with a density of $50\frac{kg}{m^3}$ was choosen. The first response of this model was calculated in the same manner as for the "real" model. Input parameters for the inversion are first density model, observed and first calculated anomaly, distance parameter and minimum split-up size to update the response and a property to store the difference. The user interface for this inversion is shown in figure 2.13. The model and the results are shown in figure 2.12. In the algorithm a damping factor θ has been added so that

$$\mathcal{W}^{-1} = (\mathcal{W}^2 + \theta\mathcal{I})^{-1} \quad (2.14)$$

where \mathcal{I} states the identity matrix.

The results depend on the choosen damping factor: if it is to small the solution will oscilate

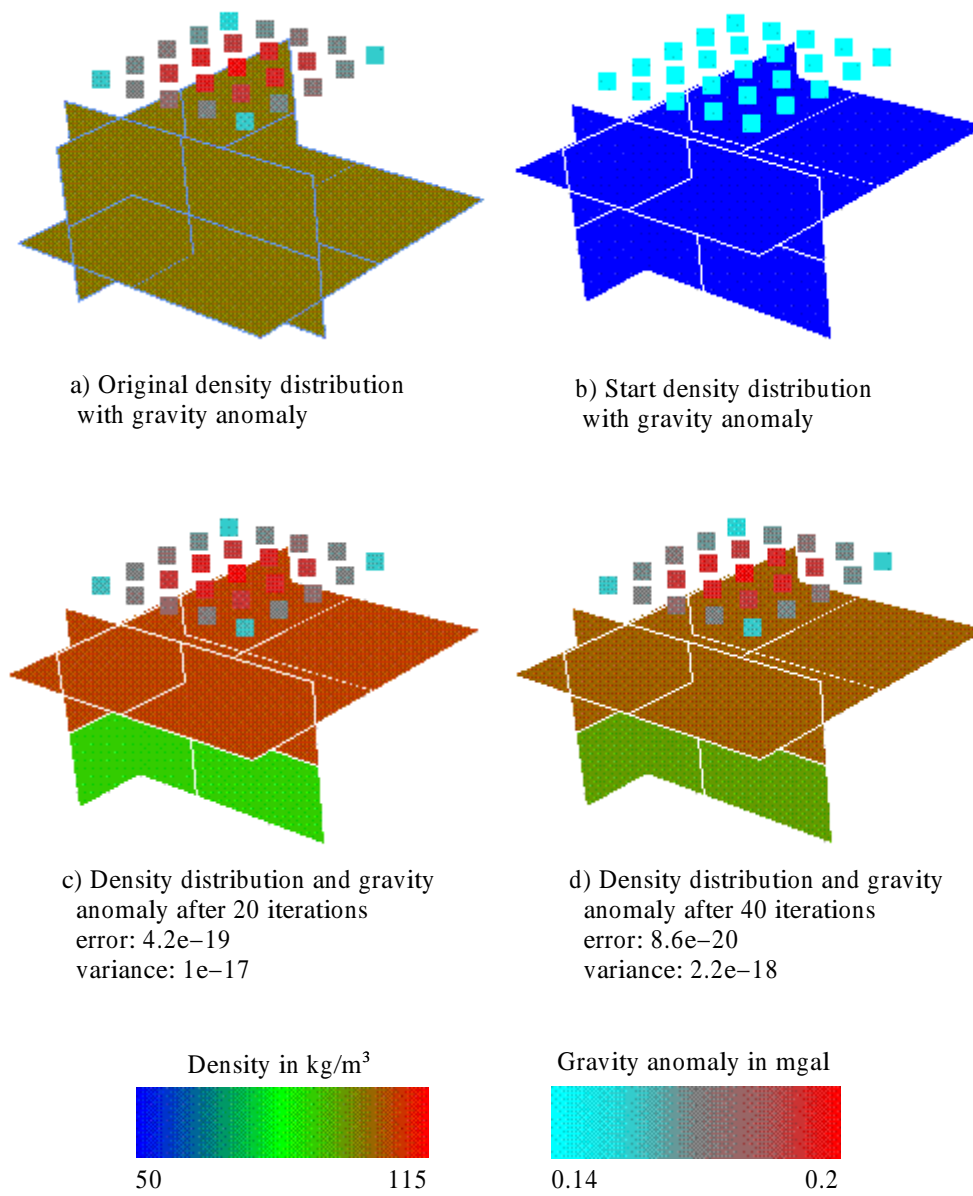


Figure 2.12: Inversion of synthetic data using SVD

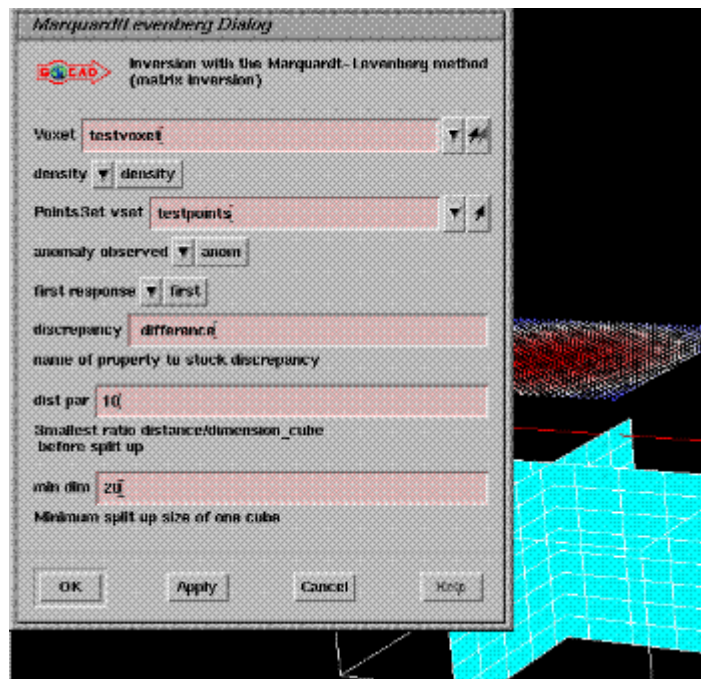


Figure 2.13: User interface for inversion with SVD

unbounded, if it is too high the eigenvalues will have no influence. In this example the damping value is adjusted with each iteration. In the first iteration the same value as the mean error is used afterwards it is decreased if no parameter change is observed or increased otherwise. As convergence criteria in this problem the mean error between observed and calculated anomaly has been used. The variance of error was calculated as well. For both parameters a decrease with increasing number of iterations can be observed. Bear(1995) [2] proposed as convergence criteria the resolution matrix of the given model. In contrast to his work convergence in five to eight iterations could not be found. Disadvantage of this method is the time consuming process of iterative computation of the model response and the damping factor which has to be found according to model parameters.

Since the gravity method is a potential field method no unique solution can be expected. Different density distributions can generate the same gravity anomaly. The solution needs to be constrained with additional information, e.g. expected density ranges in certain areas, known densities through well data or increasing density with depth. So far constraints with this method could not be included.

2.4.2 Inversion with the DSI Algorithm

Discrete Smooth Interpolation is an interpolation method developed by professor Mallet [5]. It has been developed to interpolate heterogeneous data as found in geomodeling. Heterogeneous data means unevenly distributed data, data of different magnitude and data which is more or less reliable. The DSI method aims to find in all the possible solutions the solution

which is the most smooth one. For this a criterion "roughness" has been introduced. The DSI method can take into account constraints. The algorithm of DSI has been fully implemented into the Gocad software. The advantage of the DSI method is the generic formulation of the constraints. This makes it easy for programmers to add new constraints for interpolations of various kinds.

Theory To describe a discrete topological model the notation $M^n(\Omega, N, \varphi, C)$ is introduced, where

- Ω is the set of all nodes of the model: $\Omega = \{1, 2, \dots, \alpha, \dots, M\}$
- N is a subset of Ω , $N(\alpha)$ is called the neighbourhood of the node α
- φ are the vectorial functions defined on Ω
 $\varphi(\alpha) = \{\varphi^1(\alpha), \varphi^2(\alpha), \dots, \varphi^\nu(\alpha), \dots\} \forall \alpha \in \Omega$
- C is the set of constraints attached to the model

Each constraint $c \in C$ is assumed to be linear and to have the general form, where $A_c^\nu(\alpha)$ and b_c are given coefficients defining the constraint c :

$$\sum_{\alpha \in \Omega} \sum_{\nu} A_c^\nu(\alpha) \cdot \varphi^\nu(\alpha) = \simeq > b_c \quad (2.15)$$

The three different forms correspond to three different kinds of constraints. The $=$ stands for a hard equality constraint which has to be strictly respected, $>$ is a hard inequality constraint which also has to be strictly respected and \simeq stands for a soft equality constraint which has to be respected in a least square sense.

The DSI has been introduced for interpolating the function φ on the model M while taking into account the constraints C . For this purpose, the DSI method aims to look for the function φ minimizing a criterion called global roughness $R^*(\varphi)$ defined by:

$$R^*(\varphi) = \sum_{k \in \Omega} \mu(k) \cdot R(\varphi|k) + (\phi \cdot \omega) \cdot \sum_{c \in C} \omega_c \cdot \rho(\varphi|c) \quad (2.16)$$

with

$$R(\varphi|k) = \left| \sum_{\alpha \in N(k)} v_\alpha(k) \cdot \varphi(\alpha) \right|^2 \quad (2.17)$$

(local roughness at node α) and

$$\rho(\varphi|c) = \left| \sum_{\alpha \in \Omega} A_c(\alpha) \cdot \varphi(\alpha) - b_c \right|^2 \quad (2.18)$$

The function $\mu(k)$ is a stiffness coefficient used to weight the relative importance of the local roughness at node α , ω is a factor to balance the relative contribution of the local roughness to the global roughness, ϕ is a fitting factor to specify how important it is to satisfy the constraints versus having a smooth DSI solution. A complete presentation of all the factors and their calculation is not possible and necessary in this report, for more information see Mallet[5].

Local DSI equation The DSI solution φ is such that $\partial R^*(\varphi)/\partial\varphi^\nu(\alpha) = 0$; hence the component $\varphi^\nu(\alpha)$ of φ must satisfy the following equation

$$\varphi^\nu(\alpha) = -\frac{G^\nu(\alpha|\varphi) + (\phi \cdot \omega) \cdot \Gamma^\nu(\alpha|\varphi)}{g^\nu(\alpha) + (\phi \cdot \omega) \cdot \gamma^\nu(\alpha)} \quad (2.19)$$

The terms $G^\nu(\alpha|\varphi)$ and $g^\nu(\alpha)$ depend on the function $\mu(k)$ and $v_\alpha^\nu(k)$, see Mallet[5]. It is

$$\Gamma^\nu(\alpha|\varphi) = \sum_{c \in C} \omega_c \cdot \Gamma_c^\nu(\alpha|\varphi) \quad (2.20)$$

with

$$\Gamma_c^\nu(\alpha|\varphi) = A_c^\nu(\alpha) \cdot \left\{ \sum_{\beta \neq \alpha} A_c^\nu(\beta) \cdot \varphi^\nu(\beta) - b_c \right\} \quad (2.21)$$

and

$$\gamma^\nu(\alpha) = \sum_{c \in C} \omega_c \cdot \gamma_c^\nu(\alpha) \quad (2.22)$$

with

$$\gamma_c^\nu(\alpha) = (A_c^\nu(\alpha))^2 \quad (2.23)$$

Since the DSI algorithm is fully implemented the programmer only needs to provide the terms $\Gamma_c^\nu(\alpha|\varphi)$ and $\gamma_c^\nu(\alpha)$ which depend on the constraint and its balancing factor.

Gravity constraint For the gravity inversion one global constraint has been developed. That means the possible density distribution will be constrained in such a way that the summed gravity effect of all cubes is the observed gravity anomaly for each measure point. Recalling equation 2.1 the gravity constraint has for one measure point the form

$$\sum_{\alpha} \frac{r_z(\alpha)}{\|r(\alpha)\|^3} \cdot \varphi(\alpha) = b_c \quad (2.24)$$

where α is the entirety of cubes, \vec{r} is the vector between cube and measure point, $\varphi(\alpha)$ is the density of one cube and b_c corresponds to the measured anomaly. The equation has been normalized by the gravity constant and the volume which is constant in the voxel. The constraint needs further to be normalized so that $\|A_c\|^2 = 1$. Therefore a normalizing factor is calculated for each measure point β :

$$a_c(\beta) = \left\{ \sum_{\alpha} \left(\frac{r_z(\alpha)}{\|r(\alpha)\|^3} \right)^2 \right\}^{\frac{1}{2}} \quad (2.25)$$

Now the terms $\Gamma_c^\nu(\alpha|\varphi)$ and $\gamma_c^\nu(\alpha)$ can be calculated for each cube α .

$$\Gamma_c^\nu(\alpha|\varphi) = \sum_{\beta} \frac{r_z(\alpha_\beta)}{\|r(\alpha_\beta)\|^3} \cdot \frac{\frac{r_z(\alpha_\beta)}{\|r(\alpha_\beta)\|^3} \cdot \varphi(\alpha) - b_c}{a_c(\beta)} \quad (2.26)$$

$$\gamma_c^\nu(\alpha) = \sum_{\beta} \left\{ \frac{\frac{r_z(\alpha_\beta)}{\|r(\alpha_\beta)\|^3}}{a_c(\beta)} \right\}^2 \quad (2.27)$$

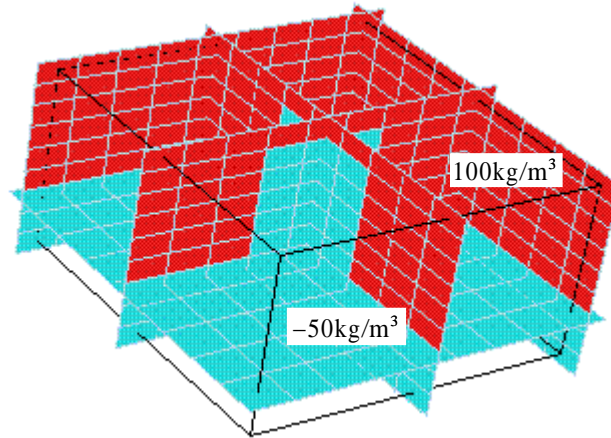


Figure 2.14: Assumed true density distribution

The DSI method is an iterative algorithm, so in every iteration the terms $\Gamma_c^\nu(\alpha|\varphi)$ and $\gamma_c^\nu(\alpha)$ are recomputed and the densities are updated. The normalizing factor is computed in the initializing process and stored.

To further constrain the density a range constraint has been developed. This means that the densities are forced to be located in distinctive ranges. They can be determined with additional geological information. There is no limit on how many ranges are used and they can overlap.

Test results So far the DSI gravity constraint has only been tested with synthetic data. One example is presented here. The assumed true density is shown in figure 2.14. The dimension of the model are 9x9x9 cubes with 200m x 200m x 100m for each cube. The anomaly has been calculated with the algorithm described in section 2.2 on a set of 324 points.

All inversions have been started with a uniform density of $50\frac{kg}{m^3}$. The user interface to set the constraint is shown in figure 2.15. The voxel contains the density information, the name of the property is given in the second parameter. The measured anomaly is stored in a property of the set of points. The calculated anomaly will be stored in another property. The last parameter "certainty factor" is a measurement of how much this constraint must be satisfied in comparison to other constraints. The voxel property is then interpolated in an existing menu: SGrid \rightarrow Interpolation \rightarrow InterpolateProperty.

The obtained density distribution after 50 iteration is shown in figure 2.16. The constraint has been observed, that means the gravity anomaly caused by this density distribution equals the observed anomaly (mean deviation 3%). The two anomalies are shown in figure 2.17. To test the convergence of the algorithm the densities have been compared between two succeeding iterations. The results are shown in table 2.1. It is difficult to estimate the correctness of the obtained densities, the inverse potential field problem will still rest ambiguous. The

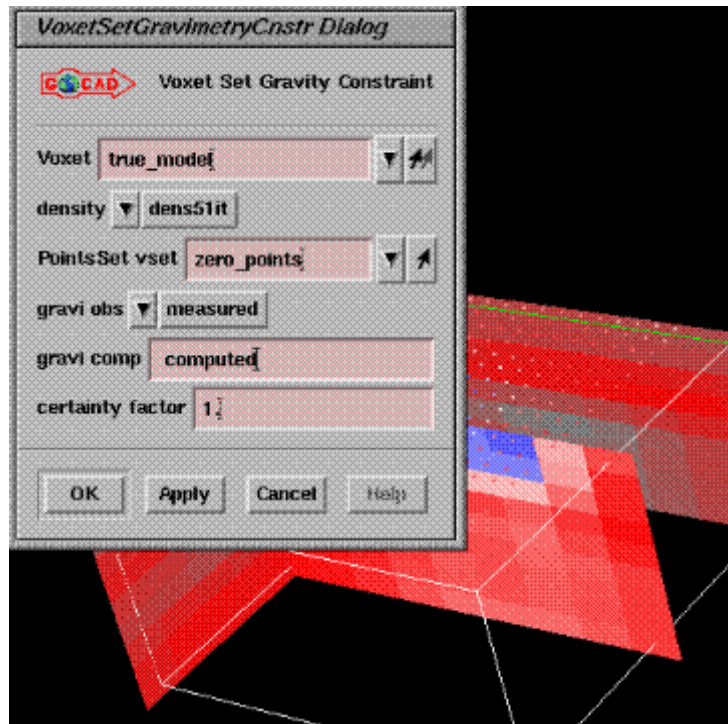


Figure 2.15: User interface for gravity constraint

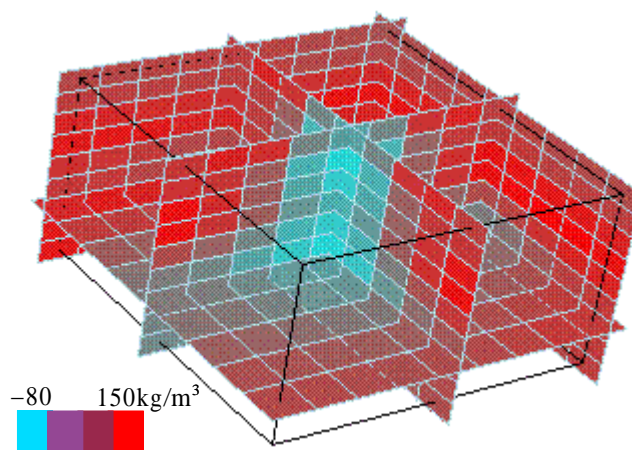


Figure 2.16: Densities after 50 iterations

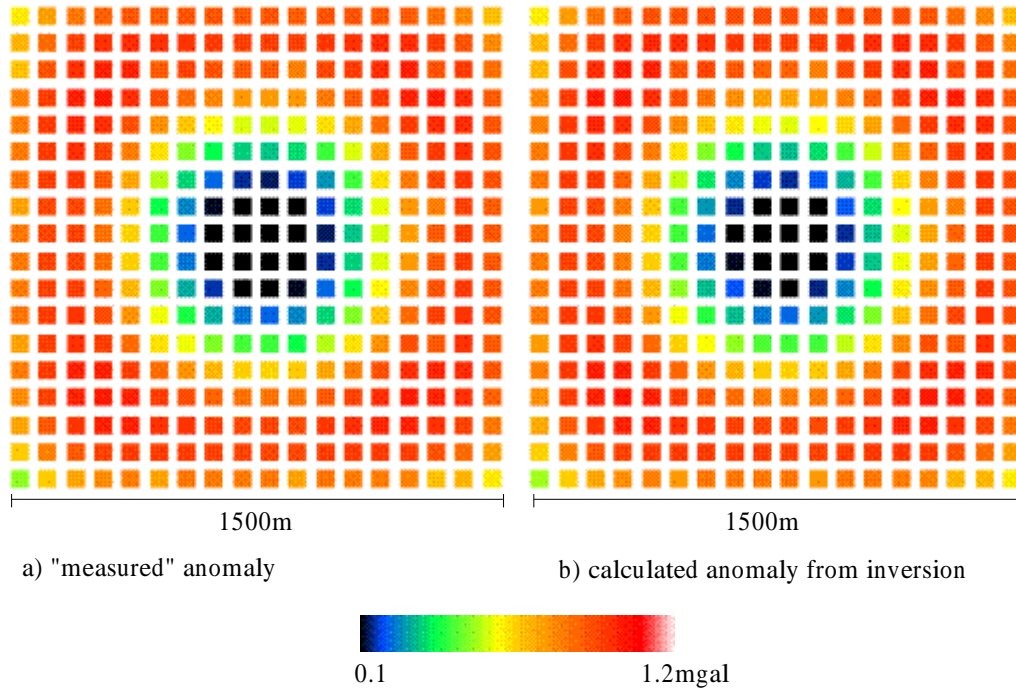


Figure 2.17: Comparison of response of true model and inverted model

| Iteration | $\sum_{\alpha}(\rho_{it+1}(\alpha) - \rho_{it}(\alpha))$ |
|-----------|--|
| 1 | 2.38e+06 |
| 5 | 5.06e+06 |
| 15 | 11015 |
| 25 | 4720 |
| 35 | 2870 |
| 50 | 1220 |
| 100 | 144 |

Table 2.1: Convergence of DSI algorithm

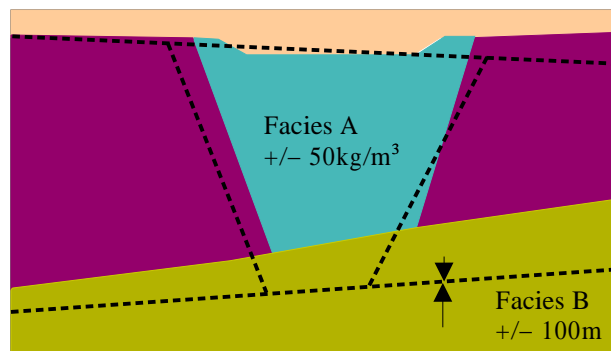


Figure 2.18: Possible implementation of DSI in gravity inversion

results are promising but the algorithm could not be applied yet to a case study.

Further work The problem of the DSI algorithm in the gravity inversion is the aim to obtain a smooth solution. In geology exist layer boundaries where the density changes abruptly. Therefore it might be more useful to initialize regions which are allowed to have densities in a certain range and in which a smooth solution is looked for. The geometry could be allowed to vary within same limites as well. The idea is shown in figure 2.18 for the situation of Chicxulub (see section 2.3). A number of other constraints as gradients or gaussian distribution are imaginable.

Chapter 3

Gravity Modeling in the Frequency Domain

3.1 Derivation of Parkers Formula

Parker(1972)[6] showed that by transforming the potential due to a layer into a Fourier integral and manipulating this expression one can obtain a sum of Fourier transforms. This sum defines the anomaly in the frequency domain or in this special case in the wavenumber domain. The desired expression is

$$\mathcal{F}[\Delta g] = -2\pi\gamma\rho \exp(-|\vec{k}|z_0) \sum_{n=1}^{\infty} \frac{|\vec{k}|^{n-1}}{n!} \mathcal{F}[h^n(\vec{r})] \quad (3.1)$$

where \vec{k} is the wave vector, $h(\vec{r})$ is the topography of the layer, z_0 is the observation level and $\mathcal{F}[\]$ denotes the Fourier transformation.

This equation can be generalized to include the case where the lower boundary of the layer is not flat, but given by $g(\vec{r})$ and the density is allowed to vary with depth $\rho(\vec{r})$

$$\mathcal{F}[\Delta g] = -2\pi\gamma \exp(-|\vec{k}|z_0) \sum_{n=1}^{\infty} \frac{|\vec{k}|^{n-1}}{n!} \mathcal{F}[\rho(\vec{r})\{h^n(\vec{r}) - g^n(\vec{r})\}] \quad (3.2)$$

There are different restrictions to this expressions. (3.1) and (3.2) hold only true if the observation level does not approach the layer of material nearer then the maximum dimension of the layer or nearer then the sampling spacing. To avoid artefacts in the transform the data has to be padded with zeros at the margins. Parker showed that for fast convergence of the sum the $z=0$ level should be chosen as average level of all topography.

3.2 Realization in Gocad

The formula of Parker (equation 3.1) has been implemented with an algorithm of the Fast Fourier Transformation as described in the book "Numerical Recipies in C" [7]. This is an algorithm well adapted for large data sets and performs the transformation and inverse

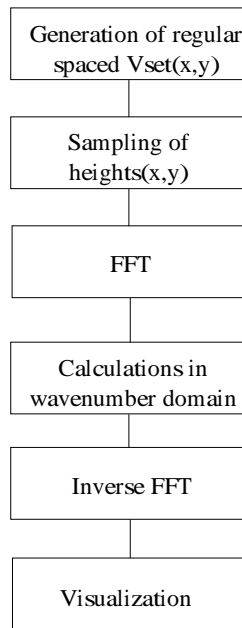


Figure 3.1: Workflow of the Parker algorithm

transformation in n-dimensional space. To calculate the gravity anomaly the steps as shown in figure 3.1 are performed. The algorithm is performed on object of the Gocad class Tsurf, which is a triangulated surface. The input parameters chosen by the user are the area of interest, number of sampling points in x and y direction and density contrast. To visualize the results an object of the Gocad class VSet (vertices set) is created. Its points are spaced in a regular grid and contain the results in a property. From this different visualization operations are easily performed, e.g. interpolation on a surface with isolines or conversation in pseudo heights.

3.3 Examples and Problems

The program can be applied to either a surface separating two layers of different density, a layer separated by two surfaces or a closed surface representing a body of different density.

Sphere First the program has been tested with the same spherical model as in section 2.2. The diameter of the sphere is 10m, the depth to its center is 10m and the density contrast is $1000 \frac{kg}{m^3}$. The result was calculated on a 128x128 point grid over an area of 100m by 100m. The user interface for this algorithm is shown in figure 3.2. Input parameters are the name(s) of the surface(s), the end points of the regular grid, number of sampling points in v direction and u direction, name of the new pointset, name of the new property and the density contrast. Figure 3.3 shows the result, in blue points the analytic results and as coloured surface the calculated anomaly.

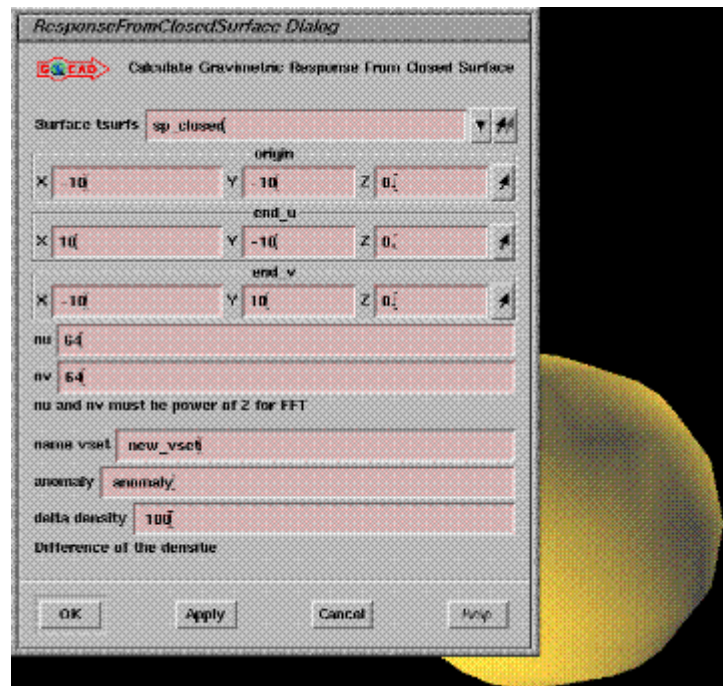


Figure 3.2: User interface for anomaly calculation in frequency domain

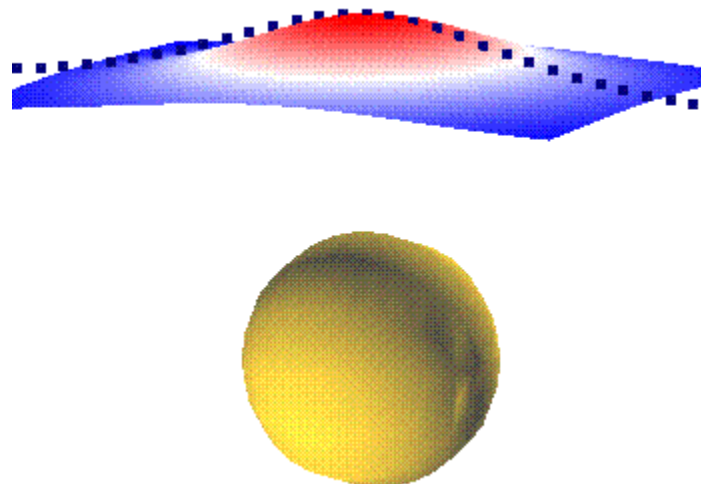


Figure 3.3: Gravity anomaly of sphere calculated with Parker algorithm

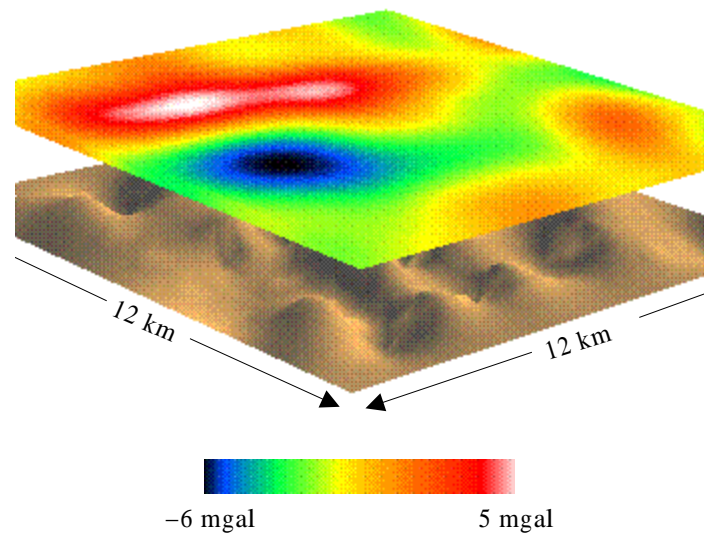


Figure 3.4: Gravity anomaly of ocean surface calculated with Parker algorithm

Layer The power of the algorithm is evident in calculating the anomaly over a complex surface. In figure 3.4 a typical ocean surface with a topography ± 400 m and its gravity anomaly is shown. The gravity anomaly was modeled over an area with the dimensions 12km by 12 km, average depth is 2100 m. In both direction 128 sample points were used. The density contrast between water and crust is $1700 \frac{kg}{m^3}$. This method is especially useful to calculate the topography correction in a field work.

Problems First restriction of this algorithm is that the anomaly can only be calculated at points of a regular grid. This can later cause problems in interpolation. The grid size has to be a power of two, so only distinctive spacing is possible.

The second problem arises from the deduction of equation 3.1. As already mentioned it can be used only if the layer of interest is far enough from the observation level. Otherwise the sum in equation 3.1 will show no convergence. This is very limiting in modeling large, not very deep structures because together with the first restriction it results in huge datasets which must be treated in a different way.

Chapter 4

Conclusions

Conclusions In this work the ability to include geophysical modeling in Gocad has been shown. For gravity modeling a number of methods have been developed. The direct modeling in the space domain has been the most useful algorithm. It is well adapted to model complex geology.

Outlook The methods so far presented should be extended with different algorithms. For the inversion different constraints as already explained in section 2.4.1 and 2.4.2 are necessary to constrain the solution. Especially the possibilities of the DSI method have been pointed out in section 2.4.2. The inverse method can also be extended as presented by Abdoh(1990)[1]. He used a linear approximation of Parkers formula (see section 3.1) to find the topography of a basin. This will be also an interative process with constant inverse matrix. The forward modeling with parametrization of subsurface needs to be accelerated, so an other method to run through the cubes has to be looked for. Another possible solution might be to transform the integral 1.7 with the help of Gaussian integral transformtion into a surface integral and to calculate the gravity anomaly from this.

A last step could be to enable interactivity for the user, so the effects of changes could be observed immediatly. All the methods described can easily be extended to model magnetic anomalies.

Appendix A

Classes and Functions

GOCAD SOFTWARE

Copyright, Association Scientifique pour la Geologie et ses Applications, 1989 - 1999, and T-Surf, 1999. Made available under license to T-Surf.

All rights reserved.

The software and information contained herein are proprietary to, and comprise valuable trade secrets of T-Surf and ASGA, which intend to preserve as trade secrets such software and information.

This software is furnished pursuant to a written license agreement and may be used, copied, transmitted, and stored only in accordance with the terms of such license and with the inclusion of the above copyright notice. This software and information or any other copies thereof may not be provided or otherwise made available to any other person.

class GravityResponse

GravityResponse::GravityResponse(Voxel* voxel, const CString& density, VSet* vset, const CString& model_anom, double dist_par, double min_dim)

Creates an object of the class GravityResponse. Voxel contains the density information, density is the name of the density property, vset are the locations at which to calculate, model_anom is the name of the new property, dist_par is the distance parameter and min_dim the minimum split up size.

void GravityResponse::calculate_response()

Calculates the gravity response by running through cubes of the voxel. Near cells are split up.

double GravityResponse::calculate_near_cubes(Cell3d near_cube, float density_of_cube, Point3d loc_obs)

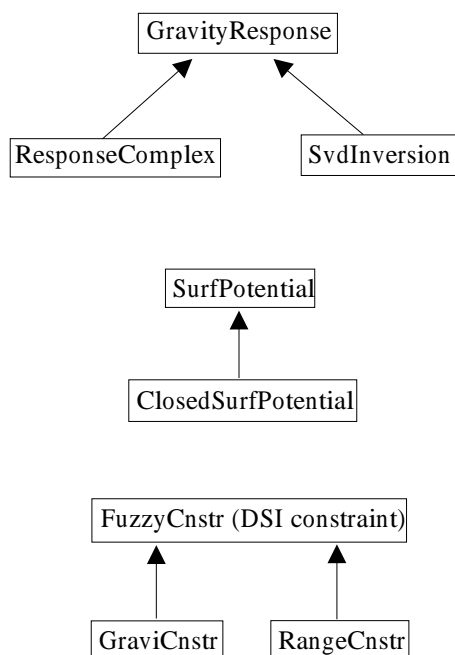


Figure A.1: Developed classes and their heritages

Calculates the gravity distribution of one cell to the anomaly at one observation point. Cells are virtually divided until they can be considered as point mass.

class ResponseComplex

The class ResponseComplex is derived from the class GravityResponse. It uses the same constructor.

ResponseComplex::ResponseComplex(Voxel* voxel, const CString& density, VSet* vset, const CString& model_anom, double dist_par, double min_dim)

ResponseComplex::response(int cube_u, int cube_v)

Calculates the gravity response in the same manner like GravityResponse::calculate_response, but takes into account only 2xcube_u in u-direction and 2xcube_v in v-direction.

GridIndex ResponseComplex::find_nearest_cube(Atom* atom)

Find to one atom of the vset the index of the nearest cube in the voxel.

List<GridIndex> ResponseComplex::list_of_index(int cube_u, int cube_v, GridIndex index)

Returns the list of indices of the cubes that are needed for the calculation of gravity anomaly at one point. Index is the index of the nearest cube.

void ResponseComplex::sum_attraction(Atom* atom, List;GridIndex; index_list)

Calculate the gravity anomaly at one point from list of indices. Anomaly is stored in the anomaly property of the class.

class SvdInversion

This class derives from GravityResponse. All matrices in functions of this class are stored in double*, so that for a nxm matrix double[n*m] is necessary. This equals the storage by rows.

SvdInversion::SvdInversion(Voxel* voxel, const CString& density, VSet* vset, const CString& model_anom, float dist_par, float min_dim, const CString& anom_obs, const CString& difference)

Creates an object of the class SvdInversion. Parameters are similar to those of the GravityResponse class. The property model_anom contains the first response of the model, the property anom_obs contains the observed anomaly and the property difference will store the difference between calculated and observed anomaly after the last iteration.

void SvdInversion::inversion()

Provides an gravity inversion based on matrix inversion with an SVD algorithm.

void SvdInversion::make_jacobien(double* j_mat)

Computes the jacobien matrix (partielle derivatives) and stores it in j_mat.

void SvdInversion::make_jacobien_trans_j(double* j_mat, double* j_trans_j)

Compute $Z^T Z$ where Z is the jacobien matrix. Results are stored in j_trans_j.

void SvdInversion::singular_value_decomposition(double* matrix, int obs, int par, double* w, double* v)

Performs the singular value decomposition of a nxm matrix. The algorithm was copied from "Numerical Recipes in C" [7]. Matrix is replaced by nxm matrix u, v is also of dimension mxm, w is of dimension n and will contain the singular values. Obs(ervation) points equal parameter n and par(ameters) equal m.

void SvdInversion::compute_delta(double* u, double* w, double* v, double* delta, double beta, int index_diff)

Computes the parameter change vector delta which is a SVD backsubstitution with damping factor delta.

double SvdInversion::compute_error(int index_diff, double& value)

Computes the error between calculated and observed anomaly. Results are stored in the property with the index `index_diff`. Variance of error is stored in `double& value`.

void SvdInversion::update_difference(int index_obs, int index_diff)

Updates the differences according to new modeled values after each iteration. `index_obs` is the property index of the observed anomaly and `index_diff` the property index of the difference.

void SvdInversion::update_density(double* delta)

Updates the density information in the voxel after each iteration with the calculated parameter change vector `delta`.

class SurfPotential

SurfPotential::SurfPotential(PtrList<TSurf> tsurfs, int nu, int nv, double dens_diff)

Creates an object of the class `SurfPotential`. One or two surfaces are stored in the `tsurfs` list. `Nu` and `nv` are the number of sampling points in `u` and `v` direction. `Dens_diff` is the density contrast between layer and surrounding material.

void SurfPotential::compute_response(VSet* vset, const CString& anomaly_name)

Computes the gravity anomaly of a layer between two surfaces or a layer which separates two layers. Calculations are performed in the wavenumber domain and results are stored in the `vset` in a property `anomaly_name`.

List<Point3d> SurfPotential::positions_of_points(Point3d origin, Point3d end_u, Point3d end_v)

Creates the regular spaced grid, parallel to the x-y plane.

double SurfPotential::sample_height(VSet* vset, double* height, TSurf* surf)

Samples the topography of surfaces `surf` as a function of average height at the points of `vset`. It returns average height. Heights are stored in `height[nuxnv]`.

void SurfPotential::prepare_data(double* data, double* height)

Add imaginary parts (zeros) for FFT in `data`, result is stored in `height` which must have two times dimension of `data`.

void SurfPotential::sum_of_fourier(double* sum, double* data_up, double* data_lo,

double obs_level, int nb_surfs, VSet* vset) Computes the sum of fourier transformed of topography until convergence (max 30 iterations). Data_up contains prepared data of upper surface, data_low contains prepared data for lower surface. The level of observation is relativ to average height of topographie.

void SurfPotential::fourier_transform(double* values, int isign, int nu, int nv)

Algorithm for 2D FFT from "Numerical Recipies in C"[7]. Transforms values of dimension nuxnvx2 (alternating real and complex part) into wavenumber domain (isign=1) and inverse(isign=-1). Nu and nv are number of samples in each direction. Must be 2^n !

void SurfPotential::display_response(double* values, const CString& ano_name, VSet* vset)

Displays the calculated values. in the created vset in a property ano_name.

class ClosedSurfPotential

Class herititates from SurfPotential. It uses the same constructor.

ClosedSurfPotential::ClosedSurfPotential(PtrList<TSurf> tsurfs, int nu, int nv, double dens_diff)

void ClosedSurfPotential::compute_response_of_body(VSet* vset, const CString& anomaly)

Computes gravity anomaly of closed surface with transformation in wavenumber domain. Closed surface is cut in two surfaces which are treated as in SurfPotential::compute_response().

double ClosedSurfPotential::sample_height_of_body(VSet* vset, double* height, TSurf* surf)

Samples topography as function of average height and returns average height. Upper and lower intersection are stored. Double* height must have dimension 2xnuxnv.

void ClosedSurfPotential::cut_at_medium(double* height_samples, double* height_up, double* height_lo, double average_h)

Sampled heights of height_samples are arranged around average and stored in two arrays which must have dimension nuxnv.

class GraviCnstr

The class GraviCnstr derives from the Gocad class FuzzyConstraint. It is set to obtain in

gravity inversion a density distribution which respects the measured anomaly.

GraviCnstr::GraviCnstr(Voxel* voxel, VSet* vset, const CString& g_obs, const CString& g_comp)

Creates an object of the class GraviCnstr. Voxel contains the density information, vset is the set of observation points. The measured anomaly is stored in a property g_obs. The calculated anomaly will be stored in a property g_comp.

virtual void GraviCnstr::init(const DsiArg&) Initializes the constraint by calculating the first response and the normalizing factors. It is called by the DSI algorithm, DsiArg is here the voxel.

virtual void GraviCnstr::fuzzy(DsiArg&)

Computes the fuzzy part of the DSI algorithm. It is called iteratively.

virtual void GraviCnstr::hard(DsiArg&)

Computes the hard part of the DSI algorithm. It is called iteratively.

virtual double GraviCnstr::weight(DsiArg&, double&)

Returns the weight for this constraint.

class GraviRangeCnstr

This class derives from the FuzzyCnstr class. It is a constraint set for gravity inversion which declares a number of possible density ranges.

GraviRangeCnstr::GraviRangeCnstr(Voxel* vox, int nb_classes, const List<RangeValue>& ranges, const CString& control_prop)

Constructs an object of the class GraviRangeCnstr. There are nb_classes number of classes, their min and max values are stored in ranges. Control_prop is the name of the property which contains the density information.

virtual void GraviRangeCnstr::init(DsiArg&)

Initializes the range constraint. The function is called by the Dsi algorithm.

virtual void GraviRangeCnstr::fuzzy(DsiArg&)

Computes the fuzzy part of the constraint. The function is called by the Dsi algorithm.

virtual void GraviRangeCnstr::hard(DsiArg&)

Computes the hard part of the constraint. The function is called by the Dsi algorithm.

virtual double GraviRangeCnstr::weight(DsiArg&, double&)

Returns the weight for this constraint. The function is called by the Dsi algorithm.

List of Figures

| | | |
|------|---|----|
| 2.1 | Subdivised voxel | 7 |
| 2.2 | Simple sphere model | 8 |
| 2.3 | Grid | 8 |
| 2.4 | Wells in the Chicxulub area, z axis 20 times exaggerated | 10 |
| 2.5 | Two surfaces outlining geological structures, z axis 15 times exaggerated | 10 |
| 2.6 | Region initialized between two surfaces, z axis 15 times exaggerated | 11 |
| 2.7 | Interface for the calculation of gravity anomaly | 11 |
| 2.8 | Overview of obtained densities in Chicxulub | 13 |
| 2.9 | Three profiles over the Chicxulub area | 14 |
| 2.10 | Results of modeled anomaly and measured anomaly | 15 |
| 2.11 | Workflow of inversion with inverse matrix | 16 |
| 2.12 | Inversion of synthetic data using SVD | 18 |
| 2.13 | User interface for inversion with SVD | 19 |
| 2.14 | Assumed true density distribution | 22 |
| 2.15 | User interface for gravity constraint | 23 |
| 2.16 | Densities after 50 iterations | 23 |
| 2.17 | Comparison of response of true model and inverted model | 24 |
| 2.18 | Possible implementation of DSI in gravity inversion | 25 |
| 3.1 | Workflow of the Parker algorithm | 27 |
| 3.2 | User interface for anomaly calculation in frequency domain | 28 |
| 3.3 | Gravity anomaly of sphere calculated with Parker algorithm | 28 |
| 3.4 | Gravity anomaly of ocean surface calculated with Parker algorithm | 29 |
| A.1 | Developed classes and their heritages | 32 |

Bibliography

- [1] Abdoh, A.; Cowan, D.; Pilkington, M. 1990. *3D Inversion of the Cheshire basin*. Geophysical Prospecting 38, 999-1011.
- [2] Bear, G.W.; Al-Shukri, H.J.; Rudman, A.J. 1995. *Linear inversion of gravity data for 3-D density distributions*. Geophysics 60, 1354-1364.
- [3] Espindola, J.M.; Mena, M.; de La Fuente, M; Campos-Enriquez, J.O. 1995. *A model of the Chicxulub impact structure (Yucatan. Mexico) based on its gravity and magnetic signatures*. Physics of the Earth and Planetary Interiors 92, 271-278.
- [4] Mallet, J.L. 1997. *Discrete Modeling for Natural Objects*. Journal of Mathematical Geology Vol. 29, No.2, 199-219.
- [5] Mallet, J.L.2000. *Geomodeling*.
- [6] Parker, R.L.1972. *The rapid calculation of potential anomalies*. Geophysical Journal of the Royal Astronomical Society 31, 447-455.
- [7] Press, W.H.and Teukolsky, S.A.; Vetterling, W.T.; Flannery, P.B.1992. *Numerical Recipes in C, 2nd ed*. Cambridge University Press.
- [8] Sharpton, V.L.; Burke, K.; Camargo-Zanoguera, A. 1993. *Chicxulub multiring impact basin. Size and other characteristics derived from gravity analysis*. Science 261: 1564-1567.
- [9] Treitel, S. and Lines, L.R. 1984. *Tutorial. A review of least-squares inversion and its application to geophysical problems*. Geophysical Prospecting 32, 159-186.