GZES Documentation

Release 1.1

UBC-GIF

Sep 05, 2017

Contents

GZES is a program library for carrying out the equivalent source processing of surface gravity data. The model can be forward modelled with gzfor3d. The contents of this manual are as follows:

GZES package overview

Description

GZES is a program library for carrying out the equivalent-source processing gravity data. The program library carries out the following functions:

- The inversion is solved as an optimization problem with the simultaneous goals of (i) minimizing a model objective function and (ii) generating synthetic data that match observations to within a degree of misfit consistent with the statistics of those data.

- By minimizing the model objective function, distributions of subsurface density contrast are found that are smooth in three dimensions creating a better solution when upward continuing data.

- The regularization parameter (controlling relative importance of objective function and misfit terms) is determined in either of twoe ways, depending upon how much is known about errors in the measured data.

- Implementation of parallel computing architecture (OpenMP) allows the user to take full advantage of multi-core processors on a CPU.

- The large size of 3D inversion problems is mitigated by the use of wavelet compression. Parameters controlling the implementation of this compression are available for advanced users. A full sensitivity matrix is also produced to remove the propagation of error into the data space from the lossiness of the wavelet transform.

NOTE: The forward modelling of the vertical component of the gravity response to a 3D volume of density contrast can be done after the inversion of a fictitious layer with gzfor3d.

The initial research underlying this program library was funded principally by the mineral industry consortium "Joint and Cooperative Inversion of Geophysical and Geological Data" (1991 - 1997) which was sponsored by NSERC and the following 11 companies: BHP Minerals, CRA Exploration, Cominco Exploration, Falconbridge, Hudson Bay Exploration and Development, INCO Exploration & Technical Services, Kennecott Exploration Company, Newmont Gold Company, Noranda Exploration, Placer Dome, and WMC.

The code itself has been produced and funded by the consortium "Potential fields and software for advanced inversion" (2012-2016) sponsored by Newmont, Teck, Glencore, BHP Billiton, Vale, Computational Geoscience Inc, Cameco, Barrick, Rio Tinto, and Anglo American.

Program library content

Executable programs

This package consists of two major programs:

- GZSENES: Calculates sensitivity matrix
- GZINVES: Performs inversion of gravity data for an equivalent source layer

Graphical user interfaces

GUI-based utilities for these codes include respective viewers for the data and models. They are only available on Windows platforms and can be freely downloaded through the UBC-GIF website:

- GM_DATA_VIEWER: a utility for viewing raw surface or airborne data (not borehole data), error distributions, and for comparing observed to predicted data directly or as difference maps.
- MeshTools3D: a utility for displaying resulting 3D models as volume renderings. Density contrast volumes can be sliced in any direction, or isosurface renderings can be generated.

Licensing

This equivalent source package is currently only available to the sponsors of "Potential fields and software for advanced inversion" consortium.

Installing

There is no automatic installer currently available for the software package. Please follow the following steps in order to use the software:

1. Extract all files provided from the given zip-based archive and place them all together in a new folder such as `C:\gzes\`

2. Add this directory as new path to your environment variables.

NOTE: Do not store anything in the "bin" directory other than executable applications and Graphical User Interface applications (GUIs).

Background theory

Introduction

The GZES suite of algorithms is based on the grav3d suit, developed at the UBC Geophysical Inversion Facility, is used to invert gravimetric responses over a three dimensional distribution of density contrast, or anomalous density. This manual is designed so that geophysicists who are familiar with the gravity experiment, but who are not necessarily versed in the details of inverse theory, can use the codes and invert their data. In the following, we describe the basics of the algorithm, but readers are referred to *[LO10]* for an in-depth discussion of various aspects of the algorithm. Note that an understanding of these components is necessary for the user to have a global view of the algorithms and to use the program library to its fullest extent.

A gravity experiment involves measuring the vertical components of the gravity field produced by anomalous (either excess or deficient) mass beneath the surface. A distribution of anomalous mass, characterized by anomalous density $\rho(x, y, z)$, produces its own gravity field, $\mathbf{g}_s$, which is superimposed on the ambient gravity field. By measuring the resultant field and removing the ambient field from the measurements through numerical processing, one obtains the field due to the anomalous mass.

The vertical component of the gravity field produced by the density $\rho(x, y, z)$ is given by

$$g_z(\mathbf{r}_o) = \gamma \int\limits_V \rho(\mathbf{r}) \frac{z - z_o}{|\mathbf{r} - \mathbf{r}_o|^3} dv, \tag{2.1}$$

where $\mathbf{r}_o = (x_o, y_o, z_o)$ is the vector denoting the observation location and $\mathbf{r} = (x, y, z)$ is the source location. The volume of the anomalous mass is V and γ is the gravitational constant. Here we have adopted a Cartesian coordinate system having its origin on the earth's surface and the $z-$axis pointing vertically downward. In the following, we outline the basics of the forward and inverse procedures used by the GRAV3D program library.

Forward Modelling

Forward modelling of gravity data is a linear problem and can be carried out by performing the integration in equation (2.1). We divide the region of interest into a set of 3D prismatic cells by using a 3D orthogonal mesh with a single layer and assume a constant density contrast within each cell. If topography is present, the layer is draped on topography.

Also, We discretize the density contrast model in this manner since it is best suited for our inversion methodology. Given such a discretization, the gravity field at the ith location can be written as:

$$d_i \equiv g_z(\mathbf{r}_{oi})$$

$$= \sum_{j=1}^{M} \rho_j \left\{ \gamma \int_{\Delta V_j} \frac{z - z_o}{|\mathbf{r} - \mathbf{r}_{oi}|^3} dv \right\}, \tag{2.2}$$

$$\equiv \sum_{j=0}^{M} \rho_j G_{ij}.$$

In equation (2.2), ρ_j and ΔV_j are the anomalous density and volume of the jth cell, d_i is introduced as a generic symbol for the ith datum, and G_{ij}, defined by the expression in brackets, quantifies the contribution of the jth cell to the ith datum. The solution for the integral in equation (2.2) can be found in *[Nag66]* and we have adopted the solution by *[Haa53]* here.

Inversion methodology

Let the set of extracted anomaly data be $\mathbf{d} = (d_1, d_2, ..., d_N)^T$ and the density contrast of cells in the model be $\rho = (\rho_1, \rho_2, ..., \rho_M)^T$. The two are related by the sensitivity matrix

$$\mathbf{d} = \mathbf{G}\rho. \tag{2.3}$$

The matrix has elements g_{ij} which quantify the contribution to the ith datum due to a unit density in the jth cell. The program performs the calculation of the sensitivity matrix, which is to be used by the subsequent inversion. The sensitivity matrix provides the forward mapping from the model to the data during the entire inverse process. We will discuss its efficient representation via the wavelet transform in a separate section.

For the inversion, the first question that arises concerns definition of the "model". We choose density contrast, ρ, as the model for since the anomalous field is directly proportional to the density contrast. The inverse problem is formulated as an optimization problem where a global objective function, ϕ, is minimized subject to the constraints in equation (2.3). The global objective functions consists of two components: a model objective function, ϕ_m, and a data misfit function, ϕ_d, such that

$$\min \phi = \phi_d + \beta \phi_m$$
$$\text{s. t. } \rho^l \leq \rho \leq \rho^u,$$

where β is a trade off parameter that controls the relative importance of the model smoothness through the model objective function and data misfit function. When the standard deviations of data errors are known, the acceptable misfit is given by the expected value ϕ_d and we will search for the value of β via an L-curve criterion *[Han00]* that produces the expected misfit. Otherwise, a user-defined value is used. Bound are imposed through the projected gradient method so that the recovered model lies between imposed lower (ρ^l) and upper (ρ^u) bounds.

We next discuss the construction of a model objective function which, when minimized, produces a model that is geophysically interpretable. However, here we stress that the functional is to create a smooth model since the layer of sources being solved for is completely fictional and a means to create a model that re-produces the data. In general, the objective function gives the flexibility to incorporate as little or as much information as possible. At the very minimum, this function drives the solution towards a reference model ρ_o and requires that the model be relatively smooth in the three spatial directions. Here we adopt a right handed Cartesian coordinate system with positive north and positive down. Let the model objective function be

$$\phi_m(\rho) =$$

$$\alpha_s \int_V \{\rho(\mathbf{r})\}^2 dv + \alpha_x \int_V \left\{ \frac{\partial \rho(\mathbf{r})}{\partial x} \right\}^2 dv + \alpha_y \int_V \left\{ \frac{\partial \rho(\mathbf{r})}{\partial y} \right\}^2 dv \tag{2.4}$$

where α_s, α_x, and α_y are coefficients, which affect the relative importance of different components in the objective function. Numerically, the model objective function in equation (2.4) is discretized onto the mesh defining the density contrast model using a finite difference approximation. This yields:

$$\phi_m(\rho) = \rho^T(\alpha_s \mathbf{W}_s^T \mathbf{W}_s + \alpha_x \mathbf{W}_x^T \mathbf{W}_x + \alpha_y \mathbf{W}_y^T \mathbf{W}_y)\rho,$$
$$\equiv \rho^T \mathbf{W}_m^T \mathbf{W}_m \rho,$$
$$= \|\mathbf{W}_m \rho\|^2,$$

where ρ is an M-length vector representing the recovered model. The next step in setting up the inversion is to define a misfit measure. Here we use the l_2-norm measure

$$\phi_d = \|\mathbf{W}_d(\mathbf{G}\rho - \mathbf{d})\|^2. \tag{2.5}$$

For the work here, we assume that the contaminating noise on the data is independent and Gaussian with zero mean. Specifying $\mathbf{W}_d$ to be a diagonal matrix whose i^{th} element is $1/\sigma_i$, where σ_i is the standard deviation of the ith datum makes ϕ_d a chi-squared distribution with N degrees of freedom. The optimal data misfit for data contaminated with independent, Gaussian noise has an expected value of $E[\chi^2] = N$, providing a target misfit for the inversion. We now have the components to solve the inversion as defined in equation (2.4).

To solve the optimization problem when constraints are imposed we use the projected gradients method *[CM87][Vog02]*. This technique forces the gradient in the Krylov sub-space minimization (in other words a step during the conjugate gradient process) to zero if the proposed step would make a model parameter exceed the bound constraints. The result is a model that reaches the bounds, but does not exceed them. This method is computationally faster than the log-barrier method because (1) model parameters on the bounds are neglected for the next iteration and (2) the log-barrier method requires the calculation of a barrier term. Previous versions of potential-field codes from UBC-GIF used the logarithmic barrier method *[Wri97][NW99]*.

Wavelet Compression of Sensitivity Matrix

The major obstacle to the solution of a large-scale gravity processing problem is the CPU time required for the application of the sensitivity matrix to model vectors. The GZES program library overcomes these difficulties by forming a sparse representation of the sensitivity matrix using a wavelet transform based on compactly supported, orthonormal wavelets. For more details, the users are referred to *[LO03][LO10]*. In the following, we give a brief description of the method necessary for the use of the GZES library.

Each row of the sensitivity matrix in a 3D gravity inversion can be treated as a 2D image and a 2D wavelet transform can be applied to it. By the properties of the wavelet transform, most transform coefficients are nearly or identically zero. When coefficients of small magnitudes are discarded (the process of thresholding), the remaining coefficients still contain much of the necessary information to reconstruct the sensitivity accurately. These retained coefficients form a sparse representation of the sensitivity in the wavelet domain. The need to store only these large coefficients means that the memory requirement is reduced. Further, the multiplication of the sensitivity with a vector can be carried out by a sparse multiplication in the wavelet domain. This greatly reduces the CPU time. Since the matrix-vector multiplication constitutes the core computation of the inversion, the CPU time for the inverse solution is reduced accordingly. The use of this approach increases the size of solvable problems by nearly two orders of magnitude. However, the model subtleties from the wavelet compression will have an impact on the forward modelled data (typically these are not worried about when 3D inversion is performed as the minute differences in the models won't affect intpertation). Therefore, the full sensitivity matrix is also written to file and can be used at the end of the inversion allowing the inversion code to solve the problem almost the entire way in the wavelet domain and then finish with the accuracy needed to create an accurate representation of the data.

Let $\mathbf{G}$ be the sensitivity matrix and $\mathcal{W}$ be the symbolic matrix-representation of the 3D wavelet transform. Then applying the transform to each row of $\mathbf{G}$ and forming a new matrix consisting of rows of transformed sensitivity is equivalent to the following operation:

$$\widetilde{\mathbf{G}} = \mathbf{G}\mathcal{W}^T, \tag{2.6}$$

where $\widetilde{\mathbf{G}}$ is the transformed matrix. The thresholding is applied to individual rows of $\mathbf{G}$ by the following rule to form the sparse representation $\widetilde{\mathbf{G}}^S$,

$$\widetilde{g}_{ij}^s = \begin{cases} \widetilde{g}_{ij} & \text{if } |\widetilde{g}_{ij}| \geq \delta_i \\ 0 & \text{if } |\widetilde{g}_{ij}| < \delta_i \end{cases}, \quad i = 1, \ldots, N, \tag{2.7}$$

where δ_i is the threshold level, and $\widetilde{g}_{ij}$ and $\widetilde{g}_{ij}^s$ are the elements of $\widetilde{\mathbf{G}}$ and $\widetilde{\mathbf{G}}^S$, respectively. The threshold level δ_i are determined according to the allowable error of the reconstructed sensitivity, which is measured by the ratio of norm of the error in each row to the norm of that row, $r_i(\delta_i)$. It can be evaluated directly in the wavelet domain by the following expression:

$$r_i(\delta_i) = \sqrt{\frac{\sum\limits_{|\widetilde{g}_{ij}|<\delta_i} \widetilde{g}_{ij}^2}{\sum\limits_{j} \widetilde{g}_{ij}^2}}, \quad i = 1, \ldots, N, \tag{2.8}$$

Here the numerator is the norm of the discarded coefficients and the denominator is the norm of all coefficients. The threshold level δ_{i_o} is calculated on a representative row, i_o. This threshold is then used to define a relative threshold $\epsilon = \delta_{i_o}/\max\limits_{j}|\widetilde{g}_{ij}|$. The absolute threshold level for each row is obtained by

$$\delta_i = \epsilon\max\limits_{j}|\widetilde{g}_{ij}|, \quad i = 1, \ldots, N. \tag{2.9}$$

The program that implements this compression procedure is GZSENES. The user is asked to specify the relative error r^* and the program will determine the relative threshold level δ_i. Usually a value of a few percent is appropriate for r^*. When both surface and borehole data are present, two different relative threshold levels are calculated by choosing a representative row for surface data and another for borehole data. For experienced users and ones that are re-inverting the data, the program also allows the direct input of the relative threshold level.

Elements of the program GZES

Introduction

The program library consists of the programs:

1. **GZSENES**: calculates sensitivity for the inversion.

2. **GZINVES**: performs gravity equivalent-source processing through inversion.

Each of the above programs requires input files and the specification of parameters in order to run. Although these programs are greatly simplified compared to 3D inversion, we first first present information about general file formats needed.

General files for GZES programs

There are four general files which are used in GZES. All are in ASCII text format. Input files can have any user-defined name. *Program output files have restricted file names that will be over-written if already in the directory*. Also the filename extensions are not important. Many prefer to use the filename convention so that files are more easily read and edited in the Windows environment. File and file locations may have spaces in the name or path, but it is discouraged. The file name (absolute or relative path) must be 500 characters or less in length. The files contain components of the inversion:

Mesh file

The mesh defines the model region. The two choices for the mesh is to (1) only have a single vertical layer or (2) to have the same cell thickness throughout the mesh. If the latter is chosen and topography is given, *GZSENES* will choose the layer based on topography. The mesh file has the following structure:

- NE: Number of cells in the East direction.

- NN: Number of cells in the North direction

- NZ: Number of cells in the vertical direction

```
NE      NN      NZ
Eₒ      Nₒ      Zₒ
ΔE₁     ΔE₂     ⋯      ΔE_NE
ΔN₁     ΔN₂     ⋯      ΔN_NN
ΔZ₁     ΔZ₂     ⋯      ΔZ_NZ
```

- E_o, N_o, Z_o: Coordinates, in meters, of the southwest top corner, specified in (Easting, Northing, Elevation). The elevation can be relative to a reference elevation other than the sea level, but it needs to be consistent with the elevation used to specify the locations, observations, and topography files.

- ΔE_n: n^{th} cell width in the easting direction (ordered W to E).

- ΔN_n: n^{th} cell width in the northing direction (ordered S to N).

- ΔZ_n: n^{th} cell thickness. **These must be all the same or NZ=1**.

The mesh can be designed in accordance with the area of interest and the spacing of the data available in the area. In general, the mesh consists of a core region which is directly beneath the area of available data, and a padding zone surrounding this core mesh. Padding is allowed with the equivalent source, but only horizontally. Within the core mesh, the size of the cells should be at least half the spacing of the data. There is no restriction on the relative position of data location and nodal points in horizontal direction. The cell width in this area is usually uniform. Beyond the core region, the mesh should be padded with cells that increase (typically no more than 40% of the previous length).

The vertical position of the mesh is specified in elevation. This is to accommodate the inversion of a data set acquired over a topographic surface. When there is strong topographic relief, which the user wishes to incorporate it into the inversion, a 2D layer will be built internally. Otherwise, the top of the mesh will be chosen as its layer. Only calculated sensitivities are stored.

Example

This example shows a mesh that consists of 26 cells in easting, 27 cells in the northing, and 23 cells in the vertical directions. The top of the mesh is located at 1610 m of elevation and the southwest corner is at -350 m easting and -400 m northing. The cells in the core portion of the mesh are all 40 m × 40 m × 25 m. There are three cells in the padding zone in every direction except vertically. This mesh is valid because each cell has a vertical cell width of 25 m.

```
26      27      23
-350    -400    1610
200     -100    50      20*40 50 100 200
200     -100    50      21*40 50 100 200
23*40
```

Topography file

This file is used to define the surface topography of a mesh/model by the elevation at different locations. Lines starting with ! are comments. The topography file has the following general structure:

Parameter definitions:

```
!       comment
npt
E₁        N₁        ELEV₁
E₂        N₂        ELEV₂
⋮         ⋮         ⋮
Eₙ        Nₙ        ELEVₙ
```

- npt: Number of points defining the topographic surface.

- E_i: Easting of the ith point on the surface.

- N_i: Northing of the ith point on the surface.

- $ELEV_i$: Elevation (metres) of the ith point on the profile.

The lines in this file can be in any order as long as the total number is equal to npt. The topographic data need not be supplied on a regular grid. GIF inversion codes assume a set of scattered points for generality and use a triangulation-based interpolation to determine the surface elevation above each column of cells. To ensure the accurate discretization of the topography, it is important that the topographic data be supplied over the entire area above the model and that the supplied elevation data points are not too sparse.

NOTE 2: Only the cells completely below the (interpolated) topographic surface are kept. The cells above or at the topographic surface are removed from the model, although these must still be included in the as if they are a part of the model. For input model files these cells can be assigned any value. The recovered model produced by inversion program also includes the cells that are excluded from the model, but these cells will have unrealistic values and be set to -100.

Example:

The following is an example of a topography file:

```
2007
12300.00   9000.00   0.109411E+04
12300.00   9025.00   0.109545E+04
12300.00   9050.00   0.109805E+04
12300.00   9075.00   0.110147E+04
12300.00   9100.00   0.110555E+04
12300.00   9125.00   0.111011E+04
12300.00   9150.00      1114.9
12300.00   9175.00   0.111971E+04
```

Observations file

This file is used to specify the observed gravity anomalies with estimated standard deviation. Lines starting with ! are comments. The following is the GIF-formatted file structure of a gravity observations file:

```
!  comment
ndat
  E₁      N₁      ELEV₁      Grav₁      Err₁
  E₂      N₂      ELEV₂      Grav₂      Err₂
  ⋮       ⋮        ⋮          ⋮          ⋮
  E_ndat  N_ndat  ELEV_ndat  Grav_ndat  Err_ndat
```

Parameter definitions:

- ndat: Number of observations.

- E, N, ELEV: Easting, northing and elevation of the observation, measured in meters. Elevation should be above the topography for surface data, and below the topography for borehole data. The observation locations can be listed in any order.

- $Grav_i$: Anomalous gravity of ith datum measured in mGal.

- Err_i: Standard deviation of $Grav_n$. This represents the absolute error. It must be positive and non-zero.

NOTE: It should be noted that the data are **extracted anomalies**, which are derived by removing the regional from the field measurements. Furthermore, the inversion program assumes that the anomalies are produced by a density contrast distribution in g/cm ^{3} with mesh cells in meters. Therefore, it is crucial that the data be prepared in `mGal`.

Predicted data file

The predicted data file is the exact same format as above, but omitting the uncertainty column. The inversion code will output the predicted data in this format. This format is also the same as gzfor3d

Example

```
!  surface data
144 !!   Number of data
   0.00      0.00    1.00    0.174732E+02    0.598678E+01
   0.00     50.00    1.00    0.265550E+02    0.613890E+01
   0.00    100.00    1.00    0.311366E+02    0.629117E+01

   ⋮         ⋮        ⋮         ⋮               ⋮

1000.00    900.00    1.00   -0.109595E+01    0.530682E+01
1000.00    950.00    1.00   -0.902209E+01    0.523738E+01
1000.00   1000.00    1.00   -0.397501E+01    0.518496E+01
```

Model file

This file contains the cell property values (g/cc) of the model and is the most common of the model files. Inversion output models are in this format. The following is the file structure of the model file

$$
\begin{array}{l}
m_{1,1,1} \\
m_{1,1,2} \\
\vdots \\
m_{1,1,NZ} \\
m_{1,2,1} \\
\vdots \\
m_{1,j,k} \\
\vdots \\
m_{NN,NE,NZ}
\end{array}
$$

Each $m_{i,j,k}$ is the property in the $[i,j,k]^{th}$ model cell. $[i, j, k]=[1, 1, 1]$ is defined as the cell at the top, south-west corner of the model. The total number of lines in this file should equal $NN \times NE \times NZ$, where NN is the number of cells in the north direction, NE is the number of cells in the east direction, and NZ is the number of cells in the vertical direction. The model ordering is performed first in the z-direction (top-to-bottom), then in the easting, and finally in the northing.

NOTE: Only the cells within the equivalent layer are kept within an inversion. All other cells will be populated with a value of -100.

Running the programs

The software package GZES uses two general codes:

- `GZSENES`: calculates sensitivity.
- `GZINVES`: performs gravity equivalent-source processing through inversion.

This section discusses the use of these codes individually.

Introduction

All programs in the package can be executed under Windows or Linux environments. They can be run by typing the program name followed by a control file in the `command prompt` (Windows) or `terminal` (Linux). They can be executed directly on the command line or in a shell script or batch file. When a program is executed without any arguments, it will print the usage to screen.

Execution

The command format and the control, or input, file format on a single machine are described below. Within the command prompt or terminal, any of the programs can be called using:

program arg_1 [$\text{arg}_2 \cdots \text{arg}_i$]

where:

- program: the name of the executable

- arg_i: a command line argument, which can be a name of corresponding required or optional file. Typing as the control file, serves as a help function and returns an example input file. Some executables do not require control files and should be followed by multiple arguments instead. This will be discussed in more detail later in this section. Optional command line arguments are specified by brackets: *[ ]*

Each control file contains a formatted list of arguments, parameters and filenames in a combination and sequence specific for the executable, which requires this control file. Different control file formats will be explained further in the document for each executable.

Input and output files

4.2.1 GZSENES

This program performs the sensitivity calculation. Command line usage:

```
gzsenes gzsenes.inp [nThreads]
```

For a sample input file type:

```
gzsenes -inp
```

The argument specifying the number of CPU threads used in the OpenMP format is optional. If this argument is not given to the program, chooses to use all of the CPU threads on the machine. This argument allows the user to specify half, for example, of the threads so that the program does not take all available RAM. Note that this option is not available in the MPI-based code used for clusters.

Input files

Format of the control file:

```
mesh       !  mesh file
obs        !  locations file
topo       !  topography file | null
wvltx      !  wavelet type:  daubx | symmx | NONE
itol, eps  !  wavelet parameters | null
```

The input parameters for the control file are:

- `mesh`: Name of modified 3D *mesh file*.

- `obs`: The *data file* that contains the observation locations. Note for sensitivity calculations, standard deviations are not required, but this file may be the observations that will be used in the inversion (with uncertainties).

- `topo`: Surface *topography*. If `null` is entered, the surface will be treated as being flat on top of the mesh.

- `wvltx`: A five-character string identifying the type of wavelet used to compress the sensitivity matrix. The types of wavelets available are Daubechies wavelet with 1 to 6 vanishing moments (`daub1`, `daub2` and so on) and Symmlets with 4 to 6 vanishing moments (`symm4`, `symm5`, `symm6`). Note that `daub1` is the Haar wavelet and `daub2` is the Daubechies-4 wavelet. The Daubechies-4 wavelet is suitable for most inversions, while the others are provided for user's experimentation. If `NONE` is entered, the program does not use wavelet compression.

- **`itol, eps`: An integer and real number that specify how the wavelet threshold level is to be determined. This line is**

    `itol=1`: program calculates the relative threshold and `eps` is the relative reconstruction error of the sensitivity. A reconstruction error of 0.05 (95%) is usually adequate.

    `itol=2`: the user defines the threshold level and `eps` is the threshold to be used. If `null` is entered on this line, a default relative reconstruction error of 0.05 (e.g. 5%) is used and the relative threshold level is calculated (i.e., `itol=1`, `eps=0.05`).

    NOTE The detailed explanation of threshold level and reconstruction error can be found in the *wavelet section* of this manual.

Example of input file

Below is an example of an input file using wavelets.

```
es.msh     !  mesh file
obs.grv    !  locations file
topo.dat   !  topography file
daub2      !  wavelet type
1 0.05     !  95% accuracy
```

Output files

The program `gzsenes` outputs three files. They are:

1. `gzsenes.log`: Log file with input file, wavelet parameters, and CPU time.

2. `gzinves.mtx`: The sensitivity matrix file to be used in the inversion. This file contains the sensitivity matrix, mesh, and discretized surface topography. It is produced by the program and it's name is not adjustable. It can get quite large and may be deleted once the work is completed.

3. `gzinves_full.mtx`: The full sensitivity matrix file (no wavelet compression) to be used in the inversion. This file contains the sensitivity matrix, mesh, and discretized surface topography. It is produced by the program and it's name is not adjustable. It can get quite large and may be deleted once the work is completed. It can be used on its own or in conjunction with the previous matrix file. This will allow the inversion to solve quickly in the wavelet domain and then accurately with the non-compressed sensitivity.

GZINVES

This program actually performs the equivalent source re-construction through the inversion of gravity data. Command line usage is:

`gzinves gzinves.inp [nThreads]`

For a sample input file type:

`gzinves -inp`

The argument specifying the number of CPU threads used in the OpenMP format is optional. If this argument is not given to the program, chooses to use all of the CPU threads on the machine. This argument allows the user to specify half, for example, of the threads so that the program does not take all available RAM. Note that this option is not available in the MPI-based code used for clusters.

Input files

Input files can be any file name. If there are spaces in the path or file name, you *MUST* use quotes around the entire path (including the filename). Files that may be used by the inversion are:

1. `obs`: Mandatory *observations file*.

2. `gzinves.mtx`: Mandatory sensitivity matrix from *GZSENES*

3. `gzinves.inp`: The control file containing the options. Does not need to be specifically called "gzinves.inp".

The input file should be formatted as follows:

```
mode               !  1 (Find beta), 2 (Fixed beta)
par, tolc          !  beta parameter and tolerance
obs                !  observed data
matrixFile         !  sensitivity matrix
VALUE lowerBound   !  lower bound bˡ
VALUE upperBound   !  upper bound bᵘ
αₛ, αₓ, α_y        !  alpha's | Le Ln | null
fullMatrixFile     !  non-compressed mtxFile | null
```

The parameters within the control file are:

- `mode`: An integer specifying one of three choices for determining the trade-off parameter.

  1. `mode=1`: the program chooses the trade off parameter by carrying out a line search so that the target value of data misfit is achieved (e.g., $\phi_d^* = N$).

  2. `mode=2`: the user inputs the trade off parameter.

- `par, tolc` Two real numbers that are dependent upon the value of mode.

  1. `mode=1`: the target misfit value is given by the product of `par` and the number of data N, i.e., `par=1` is equivalent to $\phi_d^* = N$ and `par=0.5` is equivalent to $\phi_d^* = N/2$. The second parameter, `tolc`, is the misfit tolerance in fractional percentage. The target misfit is considered to be achieved when the relative difference between the true and target misfits is less than `tolc`. Normally, `par=1` is ideal if the true standard deviation of error is assigned to each datum. When `tolc=0`, the program assumes a default value of `tolc=0.02` since this number must be positive.

  2. `mode=2`: `par` is the user-input value of trade off parameter. In this case, `tolc` is not used by the program.

  NOTE: When both `par` and `tolc` are used. When only `par` is used. When `mode=3`, neither nor `tolc` are used. However, the third line should always have two values.

- `obs`: Input *data file*. The file must specify the standard deviations of the error. By definition these values are greater than zero.

- `matrixFile`: The binary file of sensitivities created by *GZSENES*.

- `VALUE lowerBound`: The lower bounds model is defined globally as a value (e.g., `VALUE -1`).

- `VALUE upperBound`: The upper bounds model is defined globally as a value (e.g., `VALUE 1`).

- $\alpha_s, \alpha_x, \alpha_y$: Coefficients for the each model component. α_s is the smallest model component. Coefficient for the derivative in the easting direction. α_y is the coefficient for the derivative in the northing direction.

  If `null` is entered on this line, then the above four parameters take the following default values: $\alpha_s = 0.0001, \alpha_x = \alpha_y = 1$. All alphas must be positive and they cannot be all equal to zero at the same time. There is no vertical length scale because the model is a single layer of cells.

NOTE: The three coefficients may be substituted for three corresponding *length scales* L_x and L_y and are in units of metres. To understand the meaning of the length scales, consider the ratios α_x/α_s and α_y/α_s. They generally define smoothness of the recovered model in each direction. Larger ratios result in smoother models, smaller ratios result in blockier models. The conversion from α's to length scales can be done by:

$$L_x = \sqrt{\frac{\alpha_x}{\alpha_s}}; \ L_y = \sqrt{\frac{\alpha_y}{\alpha_s}}$$

When user-defined, it is preferable to have length scales exceed the corresponding cell dimensions. Typically having length scales of four cell widths are a good starting point.

- `fullMatrixFile`: The binary file of full sensitivities created by *GZSENES* (i.e., no wavelet compression). If this file is given, the program will go to 90% of the desired misfit with the compressed sensitivity matrix (line 4) and then finish the inversion with this matrix. An error will occur if the wavelet transform is applied to this matrix. This matrix could be used above and then this input in unnecessary. `null` will have the program solve to the desired misfit using the first matrix file.

Example of control file

Below is an example of a control file:

```
1                    !  find beta
1, 0.05              !  beta parameter and tolerance
obs.grv              !  observed data
gzinves.mtx          !  compressed sensitivity file
VALUE -10            !  lower bound b^l
VALUE 10             !  upper bound b^u
50 50                !  length scales (Le Ln)
gzinves_full.mtx     !  Non-compressed sensitivity file
```

Output files

Five general output files are created by the inversion. They are:

1. `gzinves.log`: The log file containing the minimum information for each iteration and summary of the inversion.

2. `gzinves.out`: The "developers" log file containing the details of each iteration including the model objective function values for each component, number of conjugate gradient iterations, etc.

3. `gzinves_xxx.den`: Density contrast *model files* output after each "xxx" iteration (i.e., `gzinves_012.den`). Model cells not used in gzinves will be given the value of -100.

4. `gzinves_xxx.pre`: *Predicted data files* (without uncertainties) output after each "xxx" iteration.

References

[CM87] P H Calamai and J J Moré. Projected gradient methods for linearly constrained problems. *Mathematical programming*, 39:96–116, 1987.

[Han00] P C Hansen. *The L-curve and its use in the numerical treatment of inverse problems*. WIT Press, Southampton, 2000.

[Haa53] I B Haáz. Relations between the potential of the attraction of the mass contained in a finite rectangular prism and its first and second derivatives. *Geofizikai Közlemenyek*, 2(7):57–66, 1953. in Hungarian.

[LO03] Y Li and D W Oldenburg. Fast inversion of large-scale magnetic data using wavelet transforms and a logarithmic barrier method. *Geophysics Journal International*, 152:251–265, 2003.

[LO10] Y Li and D W Oldenburg. Rapid construction of equivalent sources using wavelets. *Geophysics*, 75:L51–L59, 2010.

[Nag66] D Nagy. The gravitational attraction of a right rectangular prism. *Geophysics*, 31:361–371, 1966.

[NW99] J Nocedal and S J Wright. *Numerical optimatization*. Springer Science, New York, NY, 1999.

[Vog02] Curtis R Vogel. *Computational Methods for Inverse Problems (Frontiers in Applied Mathematics)*. Society for Industrial Mathematics, 2002. URL: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/0898715504.

[Wri97] S J Wright. *Primal-Dual Interior-Point Methods*. SIAM, Philadelphia, PA, 1997.