## Contents

1 MVI Package Overview 3
2 Background theory 9
3 Elements of the program MVI 17
4 Running the programs 19
5 Examples 27
6 References 31
Bibliography 33
MIV is a program library for carrying out forward modelling and inversion of magnetic data solving for the full magnetization vector, either in Cartesian or Spherical coordinate systems. Please visit the Example provided and the AtoZ Tutorial for more details. The contents of this manual are as follows:
The magnetic vector inversion (MVI) package is a program library for carrying out 3D forward modelling and inversion of magnetic data for the full magnetization vector. The program library is comprised of the following executables:

1. **MVIFWD**: Forward modelling of the magnetic anomaly response for a 3D distribution of magnetized volumes.

2. **MVIINV**: Performs the 3D magnetic vector inversion for magnetic anomaly data. The model is defined on a mesh of rectangular cells and is comprised of the effective susceptibilities in 3 orthogonal directions for each cell.

3. **MVISEN**: Calculates the sensitivities for the inversion.

**Note:**

- This code recovers the total magnetization vector in terms of effective susceptibility.
- The code models the combined contribution of induced fields (susceptibility), self-demagnetization and remanence.
- Inversion can be carried out in Cartesian \((p,s,t)\) and Spherical \((a,t,p)\) coordinate systems, but sparsity constraints can only be applied on the Spherical \((a,t,p)\) formulation.

### 1.1 Licensing

Licensing for an unconstrained academic version is available - see the Licensing policy document.

**NOTE:** All academic licenses will be **time-limited to one year**. You can re-apply after that time. This ensures that everyone is using the most recent versions of codes.

Licensing for commercial use is managed by third party distributors. Details are in the Licensing policy document.
1.2 Installing

There is no automatic installer currently available for this package. Please carry out 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
2. Add the path to the new folder to your environment variables.

Two additional notes about installation:

• Do not store anything in the “bin” directory other than executable applications and Graphical User Interface applications (GUIs).

1.3 Highlights of changes from version 2.0

• Distance weights are calculated directly from the sensitivity matrix; as a result, the user is no longer required to run the PFWEIGHT program.

• Length scales used in differential operators are set internally based on the mesh cell dimensions. The default values for \( \alpha_s \) is now 1.

• An approximated sensitivity calculation is now used to speed up the code for the MVI-Spherical formulation.

• Compression of the sensitivities both the MVI-Cartesian and MVI-Spherical formulations is now available. Default threshold tolerances are determined iteratively, and favoring lowest compression error.

Note: Download this Three Blocks Example
1.3. Highlights of changes from version 2.0
CHAPTER 2

Background theory

2.1 Introduction

Here we present the theoretical background of the magnetic problem, numerical examples, and details regarding the algorithms used by the MVI program library. This suite of algorithms, developed at the UBC-Geophysical Inversion Facility, is used to invert magnetic responses over a three-dimensional vector magnetization model. The manual is designed to help geophysicist who may be familiar with the magnetic experiment but not necessarily versed in the details of inverse theory.

Note: For more general information about the magnetic experiment, the reader is invited to visit the GPG site.

2.2 Magnetic Data

Magnetic survey data are generally comprised of a set of total magnetic intensity (TMI) measurements acquired above the Earth’s surface; although borehole data are sometime collected. The observed magnetic datum \( \hat{b}^{TMI} \) can be written as:

\[
\hat{b}^{TMI} = |\mathbf{B}_0 + \mathbf{B}_A|
\]

where both the Earth’s field \( \mathbf{B}_0 \) and the anomalous fields \( \mathbf{B}_A \) from magnetized bodies are recorded. The goal of the magnetic inversion is to obtain information about the distribution of subsurface magnetization from the data. The assumption is usually made that the anomalous field is small compared to Earth’s field, thus the following is true:

\[
\mathbf{B}_A \approx (\mathbf{B}_A \cdot \hat{\mathbf{B}}_0) \hat{\mathbf{B}}_0 \tag{2.1}
\]

where \( \hat{\mathbf{B}}_0 \) is the unit vector in the direction of Earth’s field. The Total Magnetic Anomaly can therefore be defined as

\[
\hat{b}^{TMA} = \hat{b}^{TMI} - |\mathbf{B}_0|
\]
2.3 Forward modelling

2.3.1 General formulation

The anomalous field produced by a distribution of magnetization $M$ is given by the following integral equation, where the integrand is comprised of a dyadic Green’s function:

$$\mathbf{B}_A(\mathbf{r}) = \frac{\mu_0}{4\pi} \int_V \nabla \nabla \frac{1}{|\mathbf{r} - \mathbf{r}_0|} \cdot \mathbf{M} \, dv; \quad (2.2)$$

where $\mathbf{r}$ is the observation location and $V$ represents the volume of magnetization at source locations $\mathbf{r}_0$. The above equation is valid for observation locations above the Earth’s surface; i.e. outside the region of magnetization.

The total magnetization exhibited by a rock $\mathbf{M}$ is comprised of three components:

$$\mathbf{M} = \kappa (\mathbf{H}_0 + \mathbf{H}_s) + \mathbf{M}_r, \quad (2.3)$$

where $\kappa$ is the magnetic susceptibility of the rock, $\mathbf{H}_0$ is the Earth’s primary field, $\mathbf{H}_s$ represents any ambient secondary (self-demagnetizing) fields and $\mathbf{M}_r$ represents the contribution due to magnetic remanence. The total induced magnetization is given by $\kappa (\mathbf{H}_0 + \mathbf{H}_s)$.

The magnetization of an object produces a magnetic flux. Outside the magnetized body, the magnetic flux density $\mathbf{B}$ and the magnetic field $\mathbf{H}$ are related by the magnetic permeability of free-space $\mu_0$ such that:

$$\mathbf{H} = \frac{\mathbf{B}}{\mu_0}$$

When the susceptibility is constant within a volume of source region, the Eq. (2.2) can be written in matrix form as:

$$\mathbf{B}_A = \mu_0 \begin{pmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{pmatrix} \mathbf{M} \equiv \mu_0 \mathbf{T} \mathbf{M}. \quad (2.4)$$

The tensor $T_{ij}$ is given by

$$T_{ij} = \frac{1}{4\pi} \int_V \frac{\partial}{\partial x_i} \frac{\partial}{\partial x_j} \frac{1}{|\mathbf{r} - \mathbf{r}_0|} \, dv, \text{ for } i = 1, 3; j = 1, 3, \quad (2.5)$$
where \( x_1, x_2, \) and \( x_3 \) represent \( x-, y-, \) and \( z- \)directions, respectively. The expressions of \( T_{ij} \) for a cuboidal source volume can be found in [Bha64] and [Sha66]. Since \( T \) is symmetric and its trace is equal to \(-1\) when the observation is inside the cell and is \(0\) when the observation is outside the cell, only five independent elements need to be calculated.

Once \( T \) is formed, the magnetic anomaly \( B_A \) is easily obtained. Furthermore, its projection along any measurement direction is easily obtained by taking the inner product with the directional vector. The projection of the field \( B_A \) along different directions yields different anomalies commonly obtained in magnetic surveys. For instance, the vertical anomaly is simply \( B_A z \), the vertical component of \( B_A \), whereas the total field anomaly is, to first order, the projection of \( B_A \) onto the direction of the inducing field \( B_0 \).

### 2.3.2 Numerical implementation of forward modelling

We use a right-handed coordinate system with \( z \)-axis pointing down. By equation (2.3), we divide the region of interest into a set of 3D prismatic cells and assume a constant magnetization within each cell from which we calculate the total anomalous field using equations (2.1) and (2.4). As input parameters within the data file, the coordinates, inclination and declination of the anomaly direction must be specified for each datum.

We can define the magnetization vector in terms of an effective susceptibility \( \kappa_e \) along the Cartesian directions such that

\[
M = H_0 \kappa_e
\]

and

\[
\kappa_e = \begin{bmatrix} \kappa_x \\ \kappa_y \\ \kappa_z \end{bmatrix}
\]

Let the set of extracted anomaly data be \( d = (d_1, d_2, ..., d_N)^T \) and the effective susceptibilities of cells in the model be \( \kappa_e = (\kappa_{x_1}, \kappa_{x_2}, ..., \kappa_{x_M})^T \). The two are related by the forward matrix

\[
d = G m.
\] (2.6)

The matrix has elements \( g_{ij} \) which quantify the contribution to the \( i^{th} \) datum due to a unit susceptibility in the \( j^{th} \) cell. The calculation involves the evaluation of equation (2.5) in a 3D rectangular domain defined by each cell. This operation can be done by MVIFWR if only the data is required, or by MVISEN if the forward matrix is stored on disk for the inversion. The \( G \) 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.

### 2.4 Inversion methodology

The inverse problem is formulated as an optimization problem where a global objective function, \( \phi \), is minimized subject to the constraints in equation (2.6). The global objective functions consists of two components: a model objective function, \( \phi_m \), and a data misfit function, \( \phi_d \), such that

\[
\min \phi = \phi_d + \beta \phi_m
\]

s. t. \( m^l \leq m \leq m^u \),

where \( \beta \) 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 \( \phi_d \) and we will search for the value of \( \beta \) via an L-curve criterion [Han00] that
produces the expected misfit. Otherwise, a user-defined $\beta$ value is used. Bounds are imposed through the projected gradient method so that the recovered model lies between imposed lower ($m_l$) and upper ($m_u$) bounds.

In discrete matrix form, the objective function in (2.7) can be written as

$$\phi = \phi_d + \beta \phi_m = \|W_d(F(m) - d^{obs})\|^2 + \beta \sum_{i=s,x,y,z} \|W_i(m - m_{ref})\|^2,$$

where $W_i$ are functions measuring the deviation of the model $m$ from a reference $m_{ref}$ or the roughness measured along three orthogonal directions. The following sections provide additional details about the misfit and the regularization function.

### 2.5 Misfit function $\phi_d$

The first term in (2.7) defines a measure of how well the observed data are reproduced by a model $m$. Here we use the $l_2$-norm measure

$$\phi_d = \|W_d(F(m) - d)\|^2.$$  \hspace{1cm} (2.7)

For the work here, we assume that the contaminating noise on the data is independent and Gaussian with zero mean. Specifying $W_d$ to be a diagonal matrix whose $i^{th}$ element is $1/\sigma_i$, where $\sigma_i$ is the standard deviation of the $i^{th}$ datum, makes $\phi_d$ 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 $E[\chi^2] = N$, thus providing a target misfit for the inversion. We now have the components to solve the inversion as defined in equation (2.7).

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.

### 2.5.1 Sensitivities

A solution to (2.7) is found by the second order Gauss-Newton method, such that a model update is calculated by iteratively solving

$$\frac{\partial \phi(m)}{\partial m} = J^T W_d^T W_d [F(m) - d^{obs}] + \beta W^T W (m - m_{ref})$$  \hspace{1cm} (2.8)

where $J$, also known as the sensitivity matrix, holds the derivatives of the forward operation with respect to the model

$$J = \frac{\partial F(m)}{\partial m}$$

The first question that arises during the inversion of magnetic data concerns the definition of the “model”. The MVI program allows for the inversion of a magnetization vector defined in either Cartesian or Spherical coordinate systems \[Lel09\]. We define both systems below.

### Cartesian (PST)

The first choice is to define a model $m$ in terms of effective magnetic susceptibility $\kappa_e$ along a rotated coordinate system such that one of the components is aligned with the inducing field $H_0$. Thus

$$M = |H_0| \begin{bmatrix} \kappa_p \\ \kappa_S \\ \kappa_T \end{bmatrix},$$

$$m_{PST} = \Omega_\phi \Omega_\theta \kappa_{xyz}$$
where \( p \) (primary), \( s \) (secondary) and \( t \) (tertiary) defines an orthogonal system that describes the magnetization vector in 3D. The matrices \( \Omega_{\phi} \) and \( \Omega_{\theta} \) define the rotation around the \( z \)-axis and \( y \)-axis respectively so that the \( x \)-axis points along the inducing field direction.

Fig. 2.2: Cartesian PST rotated coordinate system.

The sensitivity matrix \( J \) simplifies to
\[
J = \frac{\partial F(m)}{\partial m} = \tilde{G} \\
\tilde{G} = G \Omega_{\phi} \Omega_{\theta}
\]

The main advantage of this formulation is that the inversion remains linear. The drawback is that both the direction and the magnitude of magnetization are coupled in the vector components, which makes it harder to impose constraints on the magnetization vector through sparsity and/or petrophysical constraints.

**Spherical (ATP)**

As an alternative to the Cartesian formulation, the magnetization vector can be expressed in terms of an amplitude (\( \alpha \)) and two orientation angles (\( \theta, \phi \)) (ATP).

\[
m_{ATP} = \begin{bmatrix} \alpha \\ \theta \\ \phi \end{bmatrix}
\]

\[
x = \alpha \cos(\theta) \cos(\phi) \\
y = \alpha \cos(\theta) \sin(\phi) \\
z = \alpha \sin(\theta)
\]

(2.9)

The sensitivity matrix becomes non-linear due to the trigonometric transformation such that
\[
J = \frac{\partial F(m)}{\partial m} = G \, S
\]

where the matrix \( S \) holds the partial derivatives of (2.9)

\[
S = \begin{bmatrix} 
\cos \phi \cos \theta & -\alpha \sin \phi \cos \theta & -\alpha \cos \phi \sin \theta \\
\cos \phi \sin \theta & -\alpha \sin \phi \sin \theta & \alpha \cos \phi \cos \theta \\
\sin \phi & \alpha \cos \phi & 0 
\end{bmatrix}
\]

**2.5. Misfit function \( \phi_d \)**
Up until recently, solving the spherical formulation had proven to be prohibitively difficult. Issues regarding the convergence of the non-linear problem have now been addressed through an automated sensitivity re-weighting strategy. Solving for model parameters in spherical coordinates comes with the increased flexibility, as the user constrains the amplitude and orientation independently. The reader is encouraged to visit the examples section.

2.5.2 Regularization

We next discuss the construction of a model objective function which, when minimized, produces a model that is geophysically interpretable. This function gives the flexibility to incorporate as little or as much information as possible. At minimum, it drives the solution towards a reference model $m_0$ and requires that the model be relatively smooth in the three spatial directions. Let the model objective function expressed as

$$\phi_m(m) = \alpha_s \left| \int_V w_s w(r) |m(r) - m_{\text{ref}}|^{p_s} \, dv + \sum_{i=x,y,z} \alpha_i \int_V w_i w(r) \left| \frac{\partial[m(r) - m_{\text{ref}}]}{\partial i} \right|^{p_i} \, dv \right|$$

where the functions $w_s, w_x, w_y$ and $w_z$ are user defined weights, while $\alpha_s, \alpha_x, \alpha_y$ and $\alpha_z$ are coefficients which affect the relative importance between the penalty functions. Each function can use different $l_p$-norm measures to enforce sparsity on model or its gradients. The reference model $m_{\text{ref}}$ can be included optionally in the gradient penalty through the $\text{SMOOTH\_MOD\_DIF}$ option. The generalized sensitivity weighting function $w(r)$ counteract the geometrical decay of the sensitivities with respect to the distances from the observation locations. The details of the sensitivity weighting function will be discussed in the next section.

The $l_p$-norm is approximated by a Scaled-IRLS procedure such that for some general model function $f(m)$ (model or its gradients)

$$\int |f(m)|^p \, dV \approx \int \frac{f(m)^2}{(f(m(k-1))^2 + \epsilon^2)^{1-p/2}} \, dV$$

where $k$ stands for the iteration step. Numerically, the model objective function in equation Eq. (2.10) is discretized onto the mesh using a finite difference approximation. This yields:

$$\phi_m(m) = \alpha_s \| W_s R_s (m - m_{\text{ref}}) \|_2^2 + \sum_{i=x,y,z} \alpha_i \| W_i R_i G_i (m - m_{\text{ref}}) \|_2^2. \quad (2.10)$$
where \( \mathbf{m} \) and \( \mathbf{m}_0 \) are vectors of length \( 3M \) representing the recovered and reference models, respectively. The individual matrices \( \mathbf{W}_x, \mathbf{W}_y, \mathbf{W}_y, \) and \( \mathbf{W}_z \) contain user-defined weights as well as the sensitivity weighting functions \( w(\mathbf{r}) \). The gradient matrices \( \mathbf{G}_x, \mathbf{G}_y \) and \( \mathbf{G}_z \) are finite difference operators measuring the change in model values.

The diagonal matrices \( \mathbf{R}_x, \mathbf{R}_x, \mathbf{R}_y \) and \( \mathbf{R}_z \) contain sparsity weights applied on the model and its gradients:

\[
\mathbf{R} = \text{diag}\left(\gamma(f(\mathbf{m}^{(k-1)})^2 + \epsilon^2)^{p/2-1}\right)^{1/2}
\]

where \( \gamma \) is a normalization factor. More details about the sparse inversion can be found here. By default (and for PST formulation) the inversion uses the \( l_2 \)-norm penalty, in which case the \( \mathbf{R} \) matrices reduce to the identity matrix. Note that \( p \in [0, 2] \) can be defined on a cell-by-cell basis.

**Important: Change from previous versions** - The difference operators \( \mathbf{G}_i \) are now unitless, removing the need to alter scaling between the smallness and smoothness terms. By default, \( \alpha_x = \alpha_y = \alpha_z = 1 \).

### 2.6 Sensitivity Weighting

It is a well-known fact that static magnetic data have no inherent depth resolution. Thus when an inversion is performed which minimizes \( \int m(\mathbf{r})^2 d\mathbf{v} \) subject to fitting the data, the constructed susceptibility is concentrated close to the observation locations. This is a direct manifestation of the kernel’s decay with respect to the distance between the cell and observation locations. Because of the rapidly diminishing amplitude, the kernels of magnetic data are not sufficient to generate a function that possess significant structure at locations that are far away from observations.

Moreover, the trigonometric transformation associated with the spherical formulation introduces rapid changes in the sensitivity function, which affects the convergence of the algorithm.

In order to overcome these issues, we opt for an iterative re-weighting of the regularization to adjust the relative influence of the misfit and regularization functions. While previous version of the MAG3D and MVI made use of a depth or distance weighting, in this version we calculate the weights directly from the sensitivity matrix. We define the sensitivity weights as follow:

\[
\begin{align*}
\mathbf{W}_r &= \text{diag} \left( \hat{\mathbf{w}}_r \right)^{1/2} \\
\hat{\mathbf{w}}_r &= \frac{\mathbf{w}_r}{\max(\mathbf{w}_r)} \\
w_{r_j} &= \left[ \frac{\sum_{i=1}^{nD} f^{(k)}_{ij}^2 + \delta}{nD} \right]^{1/2},
\end{align*}
\]

where the superscript \( (k) \) is an iteration index and \( \delta \) is a small number added to avoid singularity.

### 2.7 Wavelet Compression of Sensitivity Matrix

The two major obstacles to the solution of a large-scale magnetic inversion problem are the large amount of memory required for storing the sensitivity matrix and the CPU time required for the application of the sensitivity matrix to model vectors. This 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]. Here, we give a brief description of the method necessary for the use of the MVI library.

Each row of the sensitivity matrix in a 3D magnetic inversion can be treated as a 3D image and a 3D 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 magnitude 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. Furthermore, 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.

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

\[
\tilde{G} = G \mathcal{W}^T,
\]

where \( \tilde{G} \) is the transformed matrix. The thresholding is applied to individual rows of \( G \) by the following rule to form the sparse representation \( \tilde{G}^S \):

\[
\tilde{g}_{ij} = \begin{cases} 
\bar{g}_{ij} & \text{if } |\bar{g}_{ij}| \geq \delta_i \ , \ i = 1, \ldots, N, \\
0 & \text{if } |\bar{g}_{ij}| < \delta_i \ .
\end{cases}
\]

(2.13)

where \( \delta_i \) is the threshold level, and \( \bar{g}_{ij} \) and \( \tilde{g}_{ij} \) are the elements of \( \bar{G} \) and \( \tilde{G}^S \), respectively. The threshold levels \( \delta_i \) 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:

\[
r_i(\delta_i) = \sqrt{\frac{\sum_{j} |\tilde{g}_{ij}|^2}{\sum_{j} g_{ij}^2}} , \ i = 1, \ldots, N, 
\]

(2.14)

Here the numerator is the norm of the discarded coefficients and the denominator is the norm of all coefficients. The threshold level \( \delta_{i_0} \) is calculated on a representative row, \( i_0 \). This threshold is then used to define a relative threshold \( \epsilon = \delta_{i_0} / \max_j |\bar{g}_{ij}| \). The absolute threshold level for each row is obtained by

\[
\delta_i = \epsilon \max_j |\bar{g}_{ij}| , \ i = 1, \ldots, N.
\]

(2.15)

The program that implements this compression procedure is MVISEN. For experienced users, the program allows the direct input of the relative threshold level. However it is recommended newer users let the program determine the optimal compression accuracy.
Elements of the program MVI

The program library consists of three executables:

1. **MVIFWD**: Performs forward modeling of magnetic data from a 3-components effective susceptibility model
2. **MVISEN**: Calculates sensitivities for the inversion
3. **MVIINV**: Performs magnetic vector inversion of magnetic data in 3D.

Each of the above programs requires an input file, supporting files and the specification of certain parameters in order to run. Before detailing the procedures for running each of the above programs, we first present information about the formats of the supporting files. Some files pertaining to this program library are formatted the same as files used by other UBC programs.

### 3.1 General files used by MVI

The MVI programs rely on UBC-formatted supporting files. These files can have any user-defined name and extension. Formatting for each of the following supporting files is explained within the GIFtools cookbook:

- Mesh
- Topography
- Observation
- Vector Model
- 3D Model

**Note:** For additional learning material, please visit our GIFtools Cookbook site:

**Caution:** Program output files have restricted file names that will be over-written if already in the directory.
The MVI software package uses three main programs:

4.1 MVIFWD

This program performs forward modelling; i.e. predicts magnetic data for a magnetic vector model. Command line syntax for running the program is:

```
mvifwd mesh.msh obs.loc model.fld [topo.dat]
```

Predicted data are output into a file called `mvifwd.mag`.

4.1.1 Input files

All files are in ASCII text format, thus they can be read with any text editor. Input files can have any name the user specifies. Details regarding the format of each file can be found in the *elements section*. The files associated with MVIFWD are:

- `mesh.msh`: The 3D mesh
- `obs.loc`: The observation file
- `model.fld`: The magnetic vector model in X,Y,Z (Cartesian) coordinates.
- `topo.dat`: Surface topography (optional). If omitted, the surface will be treated as being flat and the top of the 3D mesh.

4.1.2 Output file

Predicted data are output into a file called `mvifwd.mag`. The file format is that of the observation file without the associated standard deviations. The forward modelled data are in nT.
4.2 MVISEN

This program performs the sensitivity calculation for magnetic vector inversion. Command line syntax is:

\texttt{mvisen mvisen.inp [nThreads]}

For a sample input file type:

\texttt{mvisen -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, the program will choose 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.

4.2.1 Input files

The table below describes each rows. An example file can be copied from here.

<table>
<thead>
<tr>
<th>Line</th>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>mesh</td>
<td>3D Mesh file.</td>
</tr>
<tr>
<td>4</td>
<td>obs</td>
<td>Observations file</td>
</tr>
<tr>
<td>5</td>
<td>topography</td>
<td>Topography file</td>
</tr>
<tr>
<td>6</td>
<td>wvltx</td>
<td>Wavelet type: daubx</td>
</tr>
<tr>
<td>7</td>
<td>itol, eps</td>
<td>Wavelet parameters</td>
</tr>
</tbody>
</table>

The input parameters for the control file are:

- \texttt{mesh.msh}: Name of 3D mesh file.
- \texttt{obs.mag}: 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).
- \texttt{topo.dat}: Surface topography. If null is entered, the surface will be treated as being flat on top of the mesh.
- \texttt{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 (\texttt{daub1}, \texttt{daub2} and so on) and Symmlets with 4 to 6 vanishing moments (\texttt{symm4}, \texttt{symm5}, \texttt{symm6}). Note that \texttt{daub1} is the Haar wavelet and \texttt{daub2} is the Daubechies-4 wavelet. The Daubechies-4 wavelet is suitable for most inversions, while the others are provided for user experimentation. If \texttt{NONE} is entered, the program does not use wavelet compression.
- \texttt{itol, eps}: A positive integer value that specifies how the wavelet threshold level is determined. This line is ignored if no wavelet compression is being used, however a placeholder value must still be used in the input file.
itol=1: program calculates the relative threshold and \( \text{eps} \) 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 \( \text{eps} \) is the threshold to be used. If \text{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., \( \text{itol}=1, \text{eps}=0.05 \)).

\textbf{NOTE} The detailed explanation of threshold level and reconstruction error can be found in the \textit{wavelet section} of this manual.

\textbf{Note:} An example can be found here

### 4.2.2 Output files

The program \texttt{mvisen} outputs five files. They are:

1. \texttt{mviinv.mtx}: The sensitivity matrix file to be used in the inversion. This file contains the sensitivity matrix, generalized depth weighting function, mesh, and discretized surface topography. It is produced by the program and its name is not adjustable. It is very large and may be deleted once the work is completed.

2. \texttt{mvisen.log}: The log file produced by the program for the user's records on how the sensitivity matrix was formed including the full control file used.

3. \texttt{sensitivity\_x.txt}: This file is a model file that contains the average sensitivity for the \textit{East}ing component of the magnetization vector for each cell. This file can be used for depth of investigation analysis or for use in designing special model objective function weighting.

4. \texttt{sensitivity\_y.txt}: This file is a model file that contains the average sensitivity for the \textit{North}ing component of the magnetization vector for each cell. This file can be used for depth of investigation analysis or for use in designing special model objective function weighting.

5. \texttt{sensitivity\_z.txt}: This file is a model file that contains the average sensitivity for the \textit{vertical} component of the magnetization vector for each cell. This file can be used for depth of investigation analysis or for use in designing special model objective function weighting.

### 4.3 MVIINV

This program performs the 3D inversion of magnetic data to recover magnetic vectors as effective susceptibilities. The command line syntax for running this code is:

\texttt{mviinv mviinv.inp \{nThreads\}}

For a sample input file type:

\texttt{mviinv \text{-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.
4.3.1 Input files

Input files can be given any name. If there are spaces in the path or file name, you MUST use quotes around the entire path (including the filename).

The input file contains the following elements. An example file can be copied from here.

<table>
<thead>
<tr>
<th>Line</th>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>mode</td>
<td>1 (PST), 2 (ATP)</td>
</tr>
<tr>
<td>2</td>
<td>invMode</td>
<td>1 (Target misfit), 2 (Fix beta)</td>
</tr>
<tr>
<td>3</td>
<td>par, tol</td>
<td>(Mode 1) Chifac, tol</td>
</tr>
<tr>
<td>4</td>
<td>obs</td>
<td>Observations file</td>
</tr>
<tr>
<td>5</td>
<td>matrixFile</td>
<td>Sensitivity matrix file from MVISEN</td>
</tr>
<tr>
<td>6</td>
<td>init</td>
<td>Starting vector model file</td>
</tr>
<tr>
<td>7</td>
<td>ref</td>
<td>Reference Vector model file</td>
</tr>
<tr>
<td>8</td>
<td>act</td>
<td>Active model file</td>
</tr>
<tr>
<td>9</td>
<td>lowerBounds</td>
<td>Lower bound vector model file</td>
</tr>
<tr>
<td>10</td>
<td>upperBounds</td>
<td>Upper bound vector model file</td>
</tr>
<tr>
<td>11</td>
<td>scalings</td>
<td>(\alpha_x, \alpha_y, \alpha_z)</td>
</tr>
<tr>
<td>12</td>
<td>remGamma</td>
<td>Trade-off induced/remanence (PST)</td>
</tr>
<tr>
<td>13</td>
<td>SMOOTH_MOD</td>
<td>Reference model in gradient term</td>
</tr>
<tr>
<td>14</td>
<td>w1.dat</td>
<td>P weighting file</td>
</tr>
<tr>
<td>15</td>
<td>w2.dat</td>
<td>S weighting file</td>
</tr>
<tr>
<td>16</td>
<td>w3.dat</td>
<td>T weighting file</td>
</tr>
<tr>
<td>17</td>
<td>ps px py pz</td>
<td>Amplitude norm model file</td>
</tr>
<tr>
<td>18</td>
<td>ps px py pz</td>
<td>Theta angle norm model file</td>
</tr>
<tr>
<td>19</td>
<td>ps px py pz</td>
<td>Phi angle norm model file</td>
</tr>
</tbody>
</table>

The parameters within the control file are:

- **mode**: An integer specifying one of three choices on which solution the inversion will solve:

  1. **mode=1**: the program solves the vector problem in the **P,S,T** (Cartesian) space where **P** is the inducing field direction and **S** and **T** are its orthogonal components.

  2. **mode=2**: the program solves the vector problem in the **A,T,P** (Spherical) space where **A** is the amplitude (i.e., effective susceptibility), **T** is the theta angle, and **P** is the psi angle.
3. **mode=3**: the program solves the vector problem in the A,T,P (Spherical) and then uses the last four lines of the input file to solve the Lp/Lq problem for compactness/blockiness.

- **invMode**: An integer specifying one of three choices for determining the trade-off parameter.
  1. **invMode=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. **invMode=2**: the user inputs the trade-off parameter (par).
  3. **invMode=3**: The user gives the trade-off parameter (par) and the initial model from an A,T,P L2 inversion (mode=2) is used (and required) and the program will automatically go to the Lp/Lq solves. *This mode only runs the A.T.P formulation for Lp/Lq.*

- **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., \( \text{par}=1 \) is equivalent to \( \phi_d^* = N \) and \( \text{par}=0.5 \) 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, \( \text{par}=1 \) 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. **invMode=2**: par is the user-input value of trade off parameter. In this case, tolc is not used by the program.
  3. **invMode=3**: par is the user-input value of trade off parameter and tolc is the misfit tolerance in fractional percentage.

  **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 containing the sensitivities created by MVISEN.

- **init**: The initial magnetization vector model in P,S,T mode. Values can be defined as a value for uniform models (e.g. VALUE 0.001 0.001 0.001), or by a filename. There must be three values (P,S,T) if this option is used. Each component must be within the upper and lower bounds.

- **ref**: The reference magnetization vector model in P,S,T mode. Values can be defined as a value for uniform models (e.g. VALUE 0 0 0), or by a filename. There must be three values (P,S,T) if this option is used. Each component must be within the upper and lower bounds.

- **act**: The active model file defining which cells in the model are allowed to be solved.

- **lowerBounds**: The reference magnetization vector model in P,S,T mode. Values can be defined as a value for uniform models (e.g. VALUE -1 -1 -1), or by a filename. There must be three values (P,S,T) if this option is used. For example, a P value of -1 is a magnetization reverse to the inducing field with an amplitude of 1 SI.

- **upperBounds**: The reference magnetization vector model in P,S,T mode. Values can be defined as a value for uniform models (e.g. VALUE 1 1 1), or by a filename. There must be three values (P,S,T) if this option is used. For example, a P value of 1 is a magnetization in the inducing field direction with an amplitude of 1 SI.
• \( \alpha_s, \alpha_x, \alpha_y, \alpha_z \): Coefficients for each model component. \( \alpha_s \) is the smallest model component. \( \alpha_x \) is the coefficient for the derivative in the easting direction. \( \alpha_y \) is the coefficient for the derivative in the northing direction. The coefficient \( \alpha_z \) is for the derivative in the vertical direction.

If \texttt{null} is entered on this line, then the above four parameters take the following default values: \( \alpha_s = \alpha_x = \alpha_y = \alpha_z = 1 \). All alphas must be positive and they cannot be all equal to zero at the same time.

**NOTE:** The four coefficients in line 9 of the control file may be substituted for three corresponding length scales \( L_x, L_y \) and \( L_z \) and are in units of metres. They generally define smoothness of the recovered model in each direction. Larger ratios result in smoother models, smaller ratios result in blockier models. Internally, the length scales are converted back to \( \alpha \)-values such that:

\[
\alpha_s = \left( \frac{1}{L} \right)^2; \alpha_x = \left( \frac{L_x}{L} \right)^2; \alpha_y = \left( \frac{L_y}{L} \right)^2; \alpha_z = \left( \frac{L_z}{L} \right)^2
\]

where \( L = \max[L_x, L_y, L_z] \). When user-defined, it is preferable to have length scales exceed the corresponding cell dimensions.

• \texttt{remGamma}: This is a number that places (de-)emphasis on the remanent magnetization components (and extra scaling of \( S, T \) components versus \( P \)). If \texttt{null} is chosen, the trade-off between induced and remnant components are all 0.5. The higher the number, the stronger the inversion will try to recover an induced magnetization model.

• \texttt{SMOOTH_MOD}: This option was not available in previous versions of the code and can be used to define the reference model in and out of the derivative terms. The options are: \texttt{SMOOTH_MOD_DIF} (reference model is defined in the derivative terms) and \texttt{SMOOTH_MOD} (reference model is defined in only the smallest term). See equation (2.10) for details.

• \texttt{w1.dat}: Name of the weights file containing weighting matrices for the \( P \) component. If \texttt{null} is entered, default values of unity are used.

• \texttt{w2.dat}: Name of the weights file containing weighting matrices for the \( S \) component. If \texttt{null} is entered, default values of unity are used.

• \texttt{w3.dat}: Name of the weights file containing weighting matrices for the \( T \) component. If \texttt{null} is entered, default values of unity are used.

The \( L_p/L_q \) exponents used for sparsity applied on the model and its gradients. Sparsity can vary on a cell-by-cell basis through a column file (nC x 4) | VALUES [p, q, qy, qz] | \texttt{null} (Default 2 2 2 2)

• \texttt{P} \( qx \) \( qy \) \( Qz \): magnetization amplitude (A). The mode must be 2 or 3 and this line is not required if \texttt{mode}=1. \texttt{null} makes \( P = Q_x = Q_y = Q_z = 2 \). \( P \) works on the smallest model component and \( Q \)s are on the spatial components of the model objective function.

• \texttt{P} \( qx \) \( qy \) \( Qz \): theta angle (T: polar angle positive down). The \( L_p \) constant is ignored. The mode must be 2 or 3 and this line is not required if \texttt{mode}=1. \texttt{null} makes \( P = Q_x = Q_y = Q_z = 2 \). \( Q \)s are on the spatial components of the model objective function.

• \texttt{P} \( qx \) \( qy \) \( Qz \): phi angle (P: zenith angle). The \( L_p \) constant is ignored. The mode must be 2 or 3 and this line is not required if \texttt{mode}=1. \texttt{null} makes \( P = Q_x = Q_y = Q_z = 2 \). \( Q \)s are on the spatial components of the model objective function.

**NOTE:** This line is only incorporated for the amplitude. The smallest model component is turned off for the \( L_p \) with the two angles, theta and phi. The gradient effective zero is set to two and five degrees for theta and phi, respectively.
4.3.2 Output files

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

1. \texttt{mviinv.log}: The log file containing the minimum information for each iteration and summary of the inversion.

2. \texttt{mviinv.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. \texttt{mviinv\_xxx.amp}: Amplitude of the recovered model (i.e., effective susceptibility) for the “\texttt{xxx}” iteration in an \texttt{model file} format (e.g., “\texttt{mviinv\_004.amp}”).

4. \texttt{mviinv\_xxx.rem}: Remanent component of the recovered model for the “\texttt{xxx}” iteration in an \texttt{model file} format

5. \texttt{mviinv\_xxx.ind}: Induced component of the recovered model for the “\texttt{xxx}” iteration in an \texttt{model file} format.

6. \texttt{mviinv\_xxx.fld}: Recovered magnetization vector for the “\texttt{xxx}” iteration in an \texttt{model vector file} format.

7. \texttt{mviinv\_xxx.pre}: Predicted data files (without uncertainties) output for the “\texttt{xxx}” iteration.

All programs in the package can be executed within Windows or Linux environments. They can be run by typing the program name followed by a control file in the \texttt{command prompt} (Windows) or \texttt{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.

4.4 Execution on a single computer

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

\texttt{program arg\textsubscript{1} [arg\textsubscript{2} \cdots arg\textsubscript{i}]}

where:

- program: the name of the executable
- arg\textsubscript{i}: a command line argument, which can be the name of a 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 to the executable; which requires this control file. Different control file formats will be explained further in the document for each executable.

4.5 Execution on a local network or commodity cluster

The \texttt{MVI} program library’s main programs are currently not setup to use Message Pass Interface (MPI).
In this section, we present a simple synthetic example to illustrate forward modelling and inversion of magnetic vector models. For more in-depth training material, please visit our AtoZ example.

The example is made up of three magnetic blocks, 120 m in side length, placed 85 meters below a grid of observation points. The blocks are magnetized along different directions, indicating the effects of remanent magnetization are large.

The example, including the model and simulated data, can be downloaded here.

5.1 Susceptibility Inversion

To demonstrate the importance of accounting for remanence, we first assume all magnetization is induced and invert the data using the MAG3D inversion program. Below we see that the final magnetic susceptibility model clearly fails to recover the shape and location of the three magnetic blocks.
5.2 Magnetic Vector Inversion

In order to properly account for arbitrary orientations of magnetization, the data are re-inverted with the MVI algorithm using the input file shown below. Plotted are a smooth solution recovered using the Cartesian formulation and a sparse solution recovered using the Spherical formulation.

Comparing the results we note that:

- The smooth solution (MVI-Cartesian) imaged all three blocks at the correct locations and the magnetization directions at the center of each block are well recovered. The center block is much more difficult to identify.

- The sparse solution (MVI-Spherical) greatly simplified the magnetization model and highlighted the presence of the center block anomaly.
CHAPTER 6

References


