



A collection of programs for

Multi-modal **A**ppplied **G**eophysical **N**umerical modelling on **U**nstructured
Meshes

Peter G. Lelièvre

January 23, 2026

This document is a work-in-progress. Please contact me at plelievre@mta.ca if something needs updating, you need something explained further, or for any suggestions or requests.

Contents

1	MAGNUM preliminaries	4
1.1	About MAGNUM	4
1.2	About this document	4
2	File formats	5
2.1	Unstructured meshes	5
2.2	Rectilinear meshes	5
2.2.1	Rectilinear mesh files	5
2.2.2	Rectilinear model files	6
2.3	Model discretization specification files	6
2.4	Physical property files	7
2.4.1	Specifying the physical property model	8
2.4.2	Specifying depth/distance/sensitivity weighting for inversion	8
2.4.3	Inversion regularization parameters	10
2.4.4	Constraint options	10
2.5	Joint coupling measure files	10
2.5.1	Heating the coupling (parameters <code>rho</code> and <code>nsteps</code>)	11
2.5.2	Coupling options (parameter <code>coupling</code>)	11
2.5.3	Weighting the coupling	13
2.5.4	Defining clusters	13
2.5.5	Single property inversion	15
2.6	Regions files	15
2.7	Rock-units files	15
2.8	Optimization engine specification files	16
2.8.1	General optimization options	17
2.8.2	Local optimization options	17
2.8.3	Global optimization options	18
2.8.4	Genetic algorithm options	18
2.9	Unstructured block files	19
3	Forward modelling: FOGO	20
3.1	Program summary	20
3.2	Running the program	20
3.2.1	Command line usages:	20
3.2.2	Command line parameters	20
3.3	Data input file format	21
3.3.1	Units	22
3.3.2	Specifying the type of data response	22
3.3.3	Data observations	24
3.3.4	Specifying the modelling grid	25
3.3.5	Specifying the model	25
3.3.6	Data misfit and trade-off parameters	26
3.3.7	Sensitivity matrices and compression	26
3.3.8	Seismic traveltime modelling options	27
3.3.9	Muography traveltime modelling options	28
3.4	Output files	29
4	Mesh-based inversion: VIDI	30
4.1	Program summary	30
4.2	Running the program	31

4.2.1	Command line usages	31
4.2.2	Command line parameters	31
4.3	Outputs	31
4.3.1	Output files	31
4.3.2	Log file and terminal output	32
4.4	Input file format	33
4.4.1	revision	34
4.4.2	Mesh-related parameters	34
4.4.3	Physical property-related parameters	35
4.4.4	Data-related parameters	35
4.4.5	Regularization parameters	35
4.4.6	Instructions for rotating the smoothness directions	37
4.4.7	Constraint parameters	38
4.4.8	Joint inversion options	38
4.4.9	Optimization options	39
4.4.10	Tikhonov search options	39
4.4.11	Output options	39
4.5	The objective function	40
4.5.1	The data misfit terms	41
4.5.2	The model objective function	41
4.5.3	The joint coupling term	42
5	Surface geometry inversion: DYNO	43
5.1	Program summary	43
5.2	Running the program	43
5.2.1	Command line usages	43
5.2.2	Command line parameters	43
5.3	Outputs	44
5.3.1	Terminal output	44
5.3.2	Output files	44
5.4	Input file format	44
5.4.1	Parameters for all modes	46
5.4.2	Parameters for cart mode	47
5.4.3	Parameters for cart , casp and casd modes	47
5.4.4	Parameters for sphr , spsp and harm modes	47
5.4.5	Parameters for casp and spsp modes	47
5.4.6	Parameters for sphr , spsp and spsd modes	48
5.4.7	Parameters for casd mode	48
5.4.8	Parameters for harm mode	48
5.4.9	Defining constraints	48
6	Change Log	50

Chapter 1

MAGNUM preliminaries

1.1 About MAGNUM

MAGNUM is a collection of three programs (`FOGO`, `VIDI` and `DYN0`) for Multi-modal Applied Geophysical Numerical modelling on Unstructured Meshes. Despite the name, rectilinear meshes are also supported by this package. MAGNUM includes only forward or inverse modelling programs; for utility programs, see the `PODIUM` package.

1.2 About this document

[Section 2](#) provides reference material to explain file formats used by various MAGNUM programs. Documentation on the MAGNUM programs starts in [Section 3](#).

Chapter 2

File formats

This chapter provides information about many file formats used by the software in this documentation. Later chapters in this documentation link back to this chapter to help you organize your input files.

Avoid tab characters in all input files! Use spaces instead.

When specifying path/file names in an input file, place double quotes around them (e.g. `"../../data.txt"`). Single quotes should also work but this is machine dependent.

Often, input files specify the locations of other files. In these cases, you can use an absolute or relative path name. A relative path name should be relative to the location of the input file in which it is specified. For example, if file `main_input_file.inp` contains the line

```
meshinp "../mesh.inp"
```

then the file `mesh.inp` should reside in the directory above the one where `main_input_file.inp` lives.

2.1 Unstructured meshes

These file formats ...

- `.node`
- `.ele`
- `.neigh`
- `.poly`

... are explained in more detail on these webpages:

- Triangle: 2D tetrahedral meshing program
<http://www.cs.cmu.edu/~quake/triangle.html>
- TetGen: 3D tetrahedral meshing program
<http://wias-berlin.de/software/tetgen/1.5/doc/manual/manual006.html>

I allow an extension to those formats: attribute names can be included in the first line of the `.node` and `.ele` files as follows:

```
[standard information] "name1","name2",...,"nameN"
```

where N is the number of attributes. The names MUST be contained within double quotes and separated by commas. I cannot guarantee that other software that works with `.node` and `.ele` files will accept this format extension.

2.2 Rectilinear meshes

2.2.1 Rectilinear mesh files

UBC-GIF format mesh and model files are used for rectilinear meshes, which are explained further in various places here:

<http://gif.eos.ubc.ca>

The 3D mesh file format is as follows:

```
nE nN nV
EO NO VO
dE1 dE2 ... dEnE
dN1 dN2 ... dNnN
dV1 dV2 ... dVnV
```

where

- **nE**, **nN** and **nV** define the number of cells in the Easting, Northing and Vertical directions respectively
- point (EO,NO,VO) defines the coordinates of the top SW corner of the mesh
- the last three lines define the cell dimensions along each Cartesian direction.

On the cell dimension specification lines (the last three lines) you can indicate groups of equal cell dimensions using the syntax **n*d** which indicates **n** repetitions of the dimension **d**. For example, The following lines are equivalent:

```
100 100 100 50 25 25 25 25 25 50 100 100 100
3*100 50 5*25 50 3*100
```

2.2.2 Rectilinear model files

The standard UBC-GIF model file format is a single column of model values. I allow two extensions to that format. A file can contain multiple columns, each representing a different mesh cell attribute; each row contains all attributes for a particular cell. A file can contain an optional commented header line that specifies the attribute names:

```
# "name1", "name2", ..., "nameN"
```

where N is the number of attributes. The names MUST be contained within double quotes and separated by commas. I cannot guarantee that the UBC-GIF utilities will accept these format extensions.

2.3 Model discretization specification files

These files tell the forward modelling and inversion programs how to discretize the model.

Each line of the input file should be of the format

```
name value
```

where **name** is the name of some modelling parameter and **value** is the value for that parameter. The possible parameters and default values are listed below. If a parameter is not found in the input file then the default value is used for that parameter. Note that some of the parameters are for use in inversion and are ignored for forward modelling purposes.

Lines in the input file beginning with the **#** or **!** character are ignored and can be used as comments for your own reference.

Name	Default	Brief description
meshtype	"unstructured"	the type of mesh (the other option is "rectilinear")
meshfile	""	file containing mesh information
modelfile	""	file containing model information
neighfile	""	another file containing mesh information (unstructured meshes only)
split	0	how to convert from rectilinear to unstructured mesh
zdir	1	specifies the coordinate system
regionsfile	""	a list of regions that can be used with a surface model
nodeunitsfile	""	a list of node-related rockunits that can be used with a surface model
facetunitsfile	""	a list of facet-related rockunits that can be used with a surface model
regionunitsfile	""	a list of region-related rockunits; can be used with voxel or surface model

- The model may be a 2D or 3D rectilinear or unstructured (triangular or tetrahedral) mesh.
- The unstructured mesh can define a voxel mesh or tessellated surface.
- An unstructured voxel mesh should contain triangular cells in 2D and tetrahedral cells in 3D.

- An unstructured tessellated surface should contain line element facets in 2D and polygonal (e.g. triangular) facets in 3D.
- For a rectilinear mesh, the `meshfile` and `modelfile` should be UBC-GIF format files.
- For an unstructured mesh, the `meshfile` and `modelfile` should be `.node` and `.ele` files respectively and the optional `neighfile` a `.neigh` file (if `neighfile` is set to "null" then the neighbour information is calculated automatically, which may be slow for large problems).

`zdir`

- For 3D, set `zdir > 0` to specify $+z$ up, $+x$ East, $+y$ North;
set `zdir < 0` to specify $+z$ down, $+x$ North, $+y$ East.
For 2D, the $+x$ axis is always to the right and `zdir` only specifies the $+z$ direction. The y axis is along-strike.
- Output mesh/model `.node` files have $+z$ as specified.
- Output mesh/model `.vtu` files have $+z$ up, regardless of the value supplied for `zdir`.

2.4 Physical property files

These files tell the forward modelling and inversion programs how to convert *models* to *physical properties*.

Each line of the input file should be of the format

`name value`

where `name` is the name of some modelling parameter and `value` is the value for that parameter. The possible parameters and default values are listed below. If a parameter is not found in the input file then the default value is used for that parameter. Note that some of the parameters are for use in inversion and are ignored for forward modelling purposes.

Lines in the input file beginning with the `#` or `!` character are ignored and can be used as comments for your own reference.

The capitalized headings in the table below link to sections where the parameters are discussed in more detail.

Name	Default	Brief description
<code>proptype</code>	<code>""</code>	e.g. set to <code>"den"</code> , <code>"sus"</code> etc. for density, mag. susceptibility, etc.
SCALING PARAMETERS		
<code>mmul</code>	1.0	multiplicative scalar to convert model values to physical property values
<code>madd</code>	0.0	additive scalar to convert model values to physical property values
<code>mtrend</code>	0.0	background physical property depth trend
<code>uselog</code>	<code>t</code>	if true then log(conductivity) is used; only applies to conductivity models
<code>inputscaled</code>	<code>t</code>	specify if input model quantities are scaled (true) or actual (false) physical properties
INVERSION WEIGHTING PARAMETERS		
<code>wmode</code>	<code>"none"</code>	defines what type of weighting is used in an inversion
<code>wbeta</code>	1.0	distance/sensitivity weighting strength
<code>wnorm</code>	2.0	distance/sensitivity weighting norm
<code>wpower</code>	0.0	depth/distance weighting power
<code>wzero</code>	0.0	depth/distance weighting z_0/r_0
<code>obsfile</code>	<code>""</code>	file containing observation locations for use with distance weighting
<code>zdir</code>	<code>""</code>	defines coordinate system for <code>obsfile</code> and <code>wzero</code> (if used)
<code>datatype</code>	<code>""</code>	specifies the data type (e.g. <code>"gz"</code>) from which information is taken from for sensitivity weighting
INVERSION REGULARIZATION OPTIONS		
<code>initfile</code>	<code>"null"</code>	file containing an initial model
<code>initindex</code>	0	attribute index for the initial model file
<code>initvalue</code>	<code>*</code>	initial model value
<code>reffile</code>	<code>"null"</code>	file containing a reference model
<code>refindex</code>	0	attribute index for the reference model file
<code>refvalue</code>	<code>*</code>	reference model value
<code>wsfile</code>	<code>"null"</code>	file containing smallness weights
<code>wsindex</code>	0	attribute index for the <code>wsfile</code> smallness weights file
<code>alphas</code>	0.0	multiplier on the smallness regularization
<code>alphab</code>	1.0	multiplier on both the smallness and smoothness regularization terms
INVERSION CONSTRAINT OPTIONS		

Name	Default	Brief description
<code>boundsfile</code>	<code>"null"</code>	file containing model bounds
<code>lowerindex</code>	0	attribute index to use for the lower bound in a bounds file
<code>upperindex</code>	0	attribute index to use for the upper bound in a bounds file
<code>lowervalue</code>	*	lower bound value for the entire mesh
<code>uppervalue</code>	-10^6	upper bound value for the entire mesh

* Defaults for these parameters depend on the physical property type. See more information below.

2.4.1 Specifying the physical property model

`proptype`

- The following are the most commonly used properties:
 - `den` density; required for vertical gravity and gravity gradiometry data
 - `slo` slowness; required for seismic first arrival travel times
 - `sus` magnetic susceptibility; used for inverting magnetic data with the `"sus"` model type (see Section 3.3.2); if using the `"amp"` data type then this is the effective magnetic susceptibility (total magnetization amplitude divided by Earth's field strength)
- The following properties are used for inverting magnetic data for with the Cartesian `"pst"` magnetization vector inversion (MVI) model type:
 - `mvp` total magnetization vector p component (generally specified parallel to direction of Earth's field)
 - `mvs` total magnetization vector s component (generally specified perpendicular to direction of Earth's field)
 - `mvt` total magnetization vector t component (generally specified perpendicular to direction of Earth's field)
- The following properties are used for inverting magnetic data with the spherical `"atp"` MVI model type:
 - `sus` effective magnetic susceptibility
 - `inc` total magnetization vector inclination (phi angle)
 - `dec` total magnetization vector declination (theta angle)

`mmul`, `madd`, `mtrend`

- If m is the model value in a particular cell then the physical property value, p , used for that cell is calculated as

$$p = \text{mmul} * m + \text{madd} + \text{mtrend} * (z - z_0)$$

where z is the depth of the cell and z_0 is the depth of the top of the modelling mesh.

- Rearranging the equation above gives: $m = (p - \text{madd} - \text{mtrend} * (z - z_0)) / \text{mmul}$

which can help you set bounds, initial values, reference values, and joint coupling cluster coordinates, which must all be set in terms of the scaled physical property m .

- Typically, one has some background physical property value and is interested in the anomalous property relative to that value: set `madd` to the background value.
- If you want to rescale the physical property values then `mmul` can be used.

2.4.2 Specifying depth/distance/sensitivity weighting for inversion

`wmode`

- Depth/distance/sensitivity weighting model.
- Set to `"none"`, `"depth"`, `"distance"` or `"sensitivity"`.

`wbeta`, `wnorm`, `wpower`, `wzero`

- The depth weighting for the j^{th} cell is like

$$w_j = (d_j + h_0)^{-\text{wpower}}$$

where d_j is the *depth* (+ down) of the cell centroid and h_0 is, for example, the average survey *height* (+ up). That is the description given by [Li & Oldenburg \(1998\)](#) and I find it a bit complicated given that one quantity is defined positive-down and another positive-up. Thinking of it another way, the depth weighting should be

$$w_j = |z_j - z_0|^{-\text{wpower}}$$

where z_j is the z coordinate of the cell centroid and z_0 is, for example, the average z coordinate of the survey observation locations. Looking at the depth weighting in this latter way, you should see that subtracting z_0 calculates the required distance $|z_j - z_0|$ between the cell centroid and the average survey elevation.

- For depth weighting, **wzero** should define the average survey elevation in the input coordinate system. For example, if the topography surface is flat and lies at $z = 0$, and the survey data are all above the topography surface (i.e. airborne or ground data, no downhole data), then:
 - if the input coordinate system has $+z$ up then the **wzero** value specified in the file should be positive
 - if the input coordinate system has $+z$ down then the **wzero** value specified in the file should be negative.
- The distance weighting for the j^{th} cell is like

$$w_j = \left(\sum_{i=1}^N |r_i + \text{wzero}|^{(-\text{wpower} * \text{wnorm})} \right)^{(\text{wbeta} / \text{wnorm})}$$

where the sum is over all data observations, and r_i is the distance between the cell centroid and the i^{th} observation location. See [Li & Oldenburg \(2000b\)](#) for more information, although I do things slightly differently (I do not integrate over the volume of the cell - testing has shown this to be of minimal importance).

- For distance weighting, **wzero** should be some small value, such as half the smallest cell dimension.
- The sensitivity weighting for the j^{th} cell is like

$$w_j = \left(\sum_{i=1}^N \left| \frac{G_{ij}}{v_j} \right|^{\text{wnorm}} \right)^{(\text{wbeta} / \text{wnorm})} = v_j^{-\text{wbeta}} \left(\sum_{i=1}^N |G_{ij}|^{\text{wnorm}} \right)^{(\text{wbeta} / \text{wnorm})}$$

where the sum is over all data observations (all elements of the j^{th} column of the sensitivity matrix), G_{ij} is an element in the sensitivity matrix and v_j is the cell volume. See [Li & Oldenburg \(2000b\)](#) for more information, although I do things slightly differently (I normalize by cell volume - testing has shown this to be extremely important when cell volumes are not constant across the mesh).

- When weights are applied to mesh faces, the above equations for depth and distance weighting hold but cell centroids are replaced by face centroids. For sensitivity weighting, if the j^{th} face is between cells a and b then the weight used for the cell is:

$$w_j = \left(\frac{1}{2} \sum_{i=1}^N \left| \frac{G_{ia}}{v_a} \right|^{\text{wnorm}} + \frac{1}{2} \sum_{i=1}^N \left| \frac{G_{ib}}{v_b} \right|^{\text{wnorm}} \right)^{(\text{wbeta} / \text{wnorm})}$$

obsfile

- The observation locations from this **.node** file are used if **wmode** specifies distance weighting.
- **obsfile** should typically specify the data file used in the inversion but I'm providing you the flexibility here to do whatever crazy things you like.

datatype

- The sensitivity matrix for this data type is used if **wmode** specifies sensitivity weighting.
- **datatype** should typically be consistent with **proptype**, e.g. gravity data combined with density, but I'm providing you the flexibility here to do whatever crazy things you like.

zdir

- The **zdir** parameter only relates to (defines the coordinate system of) the **obsfile** and **wzero** parameters. These are only relevant if distance or depth weighting is used.
- For 3D, set **zdir** > 0 to specify $+z$ up, $+x$ East, $+y$ North;
set **zdir** < 0 to specify $+z$ down, $+x$ North, $+y$ East.
For 2D, the $+x$ axis is always to the right and **zdir** only specifies the $+z$ direction. The y axis is along-strike.

2.4.3 Inversion regularization parameters

initfile, initindex, initvalue

- The **initindex** column in the **initfile** (.ele or UBC-GIF file) is used as the initial model.
- If **initfile** is specified as "null" or **initindex** ≤ 0 then the initial model value is **initvalue** for the entire mesh.
- If the initial model specified does not honour all bound constraints then the model is projected such that it honours the bounds.
- If not specified in the physical property input file, the default for **initvalue** is 10^{-8} for conductivity and 0.0 for all other physical property models.

reffile, reffindex, refvalue

- The **reffindex** column in the **reffile** (.ele or UBC-GIF file) is used as the reference model.
- If **reffile** is specified as "null" then the reference model value is **refvalue** for the entire mesh.
- If not specified in the physical property input file, the default for **refvalue** is 10^{-8} for conductivity and 0.0 for all other physical property models.

alphas, wsfile, wsindex

- The **alphas** value and the weights in any **wsfile** are combined/compounded (one does not override the other).
- The **wsindex** column in the **wsfile** (.ele or UBC-GIF file) is used for the smallness weights.
- If **wsfile** is specified as "null" then the smallness weights are 1.0 for the entire mesh.

2.4.4 Constraint options

boundsfile, lowerindex, upperindex, lowervalue, uppervalue

- The **lowerindex** and **upperindex** columns in the **boundsfile** (.ele or UBC-GIF file) are used for the lower and upper bounds.
- If **boundsfile** is specified as "null" then the lower and upper bounds are **lowervalue** and **uppervalue** for the entire mesh.
- If a **boundsfile** is specified (any file name other than "null") and **lowerindex** or **upperindex** is non-positive then the specified values of **lowervalue** and **uppervalue**, respectively, are used for the entire mesh. In this way, you can have, for example, a heterogeneous upper bound specified in a **boundsfile** but the lower bound specified as some constant value for the entire mesh (or vice versa).
- If not specified in the physical property input file, the default for **lowervalue** is 10^{-8} for conductivity and 0.0 for all other physical property models.

2.5 Joint coupling measure files

These files tell the inversion program VIDI (formerly VINV) how to couple the multiple physical property models in a joint inversion. They are also required if you want to add some additional regularization, based on a joint coupling measure, to a single physical property inversion.

Each line of the input file should be of the format

name value

where **name** is the name of some modelling parameter and **value** is the value for that parameter. The possible parameters and default values are listed below. If a parameter is not found in the input file then the default value is used for that parameter. Note that some of the parameters are for use in inversion and are ignored for forward modelling purposes.

Lines in the input file beginning with the # or ! character are ignored and can be used as comments for your own reference.

Name	Default	Brief description
coupling	"null"	specifies the type of coupling
proptype1	"null"	specifies a property type for the first model in the coupling
proptype2	"null"	specifies a property type for the second model in the coupling
proptype3	"null"	specifies a property type for the third model (only used for FCM clustering)

Name	Default	Brief description
rho	0.0	final multiplier value for the joint measure
nsteps	1	number of lambda steps over which to heat the rho value
wjvalue	1.0	constant coupling weight value (applied across entire mesh)
wjfile	"null"	optional file containing cell-centred coupling weights
wjindex	1	attribute index for the wjfile for the coupling weights
OPTIONS FOR LINEAR COUPLING		
lina1	1.0	see explanation below
linb1	0.0	see explanation below
lina2	1.0	see explanation below
linb2	0.0	see explanation below
OPTIONS FOR CORRELATION COUPLING		
issqr	"t"	set to false ("f") if you want to specify a positive or negative correlation
pn	0	the sign specifies a positive or negative correlation (only used if issqr is false)
OPTIONS FOR FUZZY C-MEANS COUPLING		
fcmf	2.0	an exponential power used in the fuzzy c-means (FCM) joint measure
fcmd	"c"	specifies the distance measure to use for the FCM joint measure
slopes	""	linear model slope parameters
intercepts	""	linear model intercept parameters
OPTIONS FOR FUZZY C-MEANS OR GAUSSIAN PDF COUPLING		
wjconstant	"f"	if true then the weights in the wjfile are treated as constants
wj10	"t"	determines how the weights in the wjfile are interpreted
nclusters	0	number of clusters
centres1	""	cluster centres for first physical property
centres2	""	cluster centres for second physical property
centres3	""	cluster centres for third physical property (only used for FCM clustering)
spreads1	""	cluster spreads for first physical property
spreads2	""	cluster spreads for second physical property
rotations	""	cluster rotations in degrees
variances1	""	cluster variances for first physical property
variances2	""	cluster variances for second physical property
covariances	""	cluster covariances for the two physical properties
OPTIONS FOR SUMMATIVE GRADIENT COUPLING		
epsilon	1.0E-6	small value to help avoid division by zero
OPTIONS FOR JOINT TOTAL VARIATION (JTV) COUPLING		
epsilon	1.0E-6	small value to help avoid division by zero
power	1.0	exponent in the JTV coupling

2.5.1 Heating the coupling (parameters rho and nsteps)

A joint inversion proceeds in several stages, heating the coupling multiplier values in the objective function (see ρ_k in [Section 4.5.3](#)) from 0.0 at stage 0 up to the final value **rho** at stage **nsteps**.

2.5.2 Coupling options (parameter coupling)

eq, equal

The equal coupling option specifies that the two models should be equal:

$$\mathbf{m}_1 = \mathbf{m}_2 \quad (2.1)$$

eqgrad, equalgrad

The equal gradient coupling option specifies that the spatial gradients of the two models should be equal:

$$\mathbf{G}\mathbf{m}_1 = \mathbf{G}\mathbf{m}_2 \quad (2.2)$$

where \mathbf{G} is the spatial gradient operator.

lin, linear

The linear coupling option specifies that there is a known linear relationship between the two models:

$$a_1 \mathbf{m}_1 + b_1 = a_2 \mathbf{m}_2 + b_2 \quad (2.3)$$

lingrad, lineargrad

The linear gradient coupling option specifies that there is a known linear relationship between the spatial gradients of the two models:

$$a_1 \mathbf{Gm}_1 + b_1 = a_2 \mathbf{Gm}_2 + b_2 \quad (2.4)$$

corr, correlation

The correlation coupling option specifies that there is some linear relationship between the two models but the linear parameters of that relationship are unknown. See [Lelièvre et al. \(2012\)](#) for the mathematics. In this case, if you don't know anything about the relationship then you will set parameter `issqr` to "t" (true). If you know that the linear relationship should have a positive or negative slope then set `issqr` to "f" (false) and use parameter `pn` to specify a positive or negative correlation.

corrgrad

The gradient correlation coupling option specifies that there is some linear relationship between the spatial gradients of the two models but the linear parameters of that relationship are unknown.

cross, crossgrad

The cross-gradient structural coupling measure compares the spatial gradients of the two models. See [Gallardo & Meju \(2004\)](#) and [Fregoso & Gallardo \(2009\)](#) for more information. At any specific location in the modelling mesh, a zero cross-gradient measure can indicate that the two models have gradients in the same direction, either parallel or antiparallel, or it can indicate that one or both models has zero gradient. As such, this measure suffers from the issue of multiple local minima and it should be used with care to avoid local minima entrapment. This measure is currently only implemented in 2D.

fcm, fuzzy

The fuzzy c-means (FCM) measure is used to specify different clusters in a physical property cross-plot. See [Lelièvre et al. \(2012\)](#), [Carter-McAuslan et al. \(2015\)](#) and [Sun & Li \(2017\)](#) for the mathematics and usage suggestions. The FCM measure also suffers from the issue of multiple local minima and should be used with care to avoid local minima entrapment.

There are three different options for the distance measure used in the FCM coupling. Set parameter `fcmd` as follows:

- "c" for circular clusters (Euclidean distance); cluster centres must be specified (see [Section 2.5.4](#) below)
- "e" for elliptical clusters (Mahalanobis distance); cluster centres, spreads and rotations must be specified, or centres, variances and covariances (see [Section 2.5.4](#) below)
- "l" for a linear regression relationship; the linear regression parameters must be specified (see [Section 2.5.4](#) below).

gaus, gauss, gaussian

This coupling measure is formed from a combination of Gaussian functions. This is a similar measure to the fuzzy c-means option but is not as "fuzzy", meaning it suffers even more from the issue of multiple local minima and should be used with even more care to avoid local minima entrapment.

sumgrad

The summative gradient measure is effectively a normalized version of the equal gradient measure:

$$\frac{\mathbf{Gm}_1}{|\mathbf{Gm}_1|^2 + \text{epsilon}^2} = \frac{\mathbf{Gm}_2}{|\mathbf{Gm}_2|^2 + \text{epsilon}^2} \quad (2.5)$$

jtv, tvar, totvar

The joint total variation minimizes the following constraint vector:

$$\left((\mathbf{Gm}_1)^2 + (\mathbf{Gm}_2)^2 + \text{epsilon}^2 \right)^{\text{power}/2} \quad (2.6)$$

2.5.3 Weighting the coupling

The **wjvalue** is always applied, in addition to any other weights specified through the **wjfile**. This can be useful if, for example, the values in the coupling constraint vector are very small (as can happen with the cross-gradient measure) but you'd rather work with a scaled version of those. This scaling weight is applied inside the joint coupling calculation, before the **rho** value is applied (inside the ψ_k function in [Section 4.5.3](#)).

If a **wjfile** is not specified, or equal to "null", then the joint coupling measure will be applied across the entire modelling volume with equal weight throughout.

If the mesh is unstructured then the **wjfile** can be a **.ele** file or a column-based data file. If the mesh is rectilinear then the **wjfile** must be a column-based data file. The number of cells in the **.ele** file, or the number of rows in the column-based data file, must equal the number of cells in the inversion mesh.

For the **equal**, **lin** and **cross** coupling options, there is only a single weight required for each cell. The **wjindex** parameter specifies an attribute index for the **wjfile** and the coupling weights are taken from that attribute column. The weights are applied to the coupling measurement for each cell. For example, the **equal** measure becomes

$$\Psi_{equal} = \sum_i w_i (m_{2,i} - m_{1,i}) \quad (2.7)$$

where the sum is over every cell in the mesh.

For the **corr** coupling option, there is also only a single weight required for each cell but the weights are used differently. The weights are used to specify which region of the mesh should be included in the coupling. Cells with $w_i = 0$ are ignored. Any non-zero weighting value w_i can be used to indicate that the i^{th} cell should be included in the coupling. If you want different implicit linear relationships in different regions of your model then you will need different coupling files and associated weighting files for each region.

For the **fcm** coupling option there can be a single weight specified for each cell or multiple weights for each cell.

- If **wjconstant** is "t" (true) then the supplied weights are treated as constant membership weights and parameter **wj10** is not used. The **wjfile** and **wjindex** parameters should specify the weights w_{ij} for each cell and each cluster. Weight w_{ij} is the membership of the i^{th} cell for the j^{th} cluster. N_c attributes are required in the **wjfile**, where N_c is the number of clusters defined in the [physical property file\(s\)](#). The **wjindex** parameter specifies the attribute column to start taking the attributes from: attributes are taken from columns **wjindex** to **wjindex**+ N_c -1. Each attribute column contains the information for a different cluster; the order of the attributes in the **wjfile** should be consistent with the order of the clusters defined in the [physical property file\(s\)](#).
- If **wjconstant** is "f" (false, the default) then the supplied weights are used to specify which region of the mesh should be included in the coupling. The values in the **wjindex** file are not actually treated as coupling weights but simply determine which cells are attached to which clusters. The actual membership weights are determined automatically as part of the joint inversion process. Parameter **wj10** is treated as follows.
 - If **wj10** is "t" (true, the default) then the **wjfile** and **wjindex** parameters should be specified as above for **wjconstant** equal to "t" (true). However, cells with $w_{ij} = 0$ are ignored. Any non-zero value w_{ij} can be used to indicate that the i^{th} cell should be coupled with the j^{th} cluster.
 - If **w10** is "f" (false) then only a single attribute column is required in the **wjfile**. The **wjindex** parameter should specify the attribute column to use. The single attribute column should contain integer indices on $[1, N_c]$ that specify which cells should correspond to which cluster.

For the **gauss** coupling option, the measure is a Gaussian mixture,

$$\Psi_{gauss} = \sum_i \sum_j w_{ij} N(m_{1,i}, m_{2,i}, \dots | \mu_j, \sigma_j), \quad (2.8)$$

and the **wjfile** and **wjindex** parameters should be specified as described above for the **fcm** measure with **wjconstant** equal to "t" (true).

2.5.4 Defining clusters

centres1, **centres2**, **centres3**, **spreads1**, **spreads2**, **rotations**, **variances1**, **variances2**, **covariances**

These are string buffers specifying the cluster information. For example, if on a density-vs-susceptibility plot there are three clusters at (0.0,0.0), (1.0,0.3) and (2.0,0.5) then you will have the following lines in the coupling measure input file:

```

proptype1 den
proptype2 sus
nclusters 3
centres1 "0.0 1.0 2.0"
centres2 "0.0 0.3 0.5"

```

The cluster centres, spreads and variances/covariances should be specified in terms of the scaled physical property: see the physical property input file parameters `mmul`, `madd` and `mtrend` explained in [Section 2.4.1](#).

Below, two options are explained for specifying elliptical clusters rotated with respect to the two physical property axes. Option 1 is always used if the `rotations` parameter is specified, in which case the `variances1`, `variances2` and `covariances` information is ignored.

Option 1

You can specify the spreads and rotations (`spreads1`, `spreads2`, `rotations`) of the clusters. A positive rotation moves the first physical property axis towards the second. That is, counter-clockwise if the first and second properties are plotted on the x and y axes respectively, with x right and y up the page. The “spreads” are standard deviations for the two axes of the rotated ellipses. A matrix is calculated as follows:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{bmatrix} \quad (2.9a)$$

$$a_{11} = \frac{\cos^2 \theta}{\sigma_1^2} + \frac{\sin^2 \theta}{\sigma_2^2} \quad (2.9b)$$

$$a_{22} = \frac{\sin^2 \theta}{\sigma_1^2} + \frac{\cos^2 \theta}{\sigma_2^2} \quad (2.9c)$$

$$a_{12} = \frac{\sin 2\theta}{2\sigma_1^2} - \frac{\sin 2\theta}{2\sigma_2^2} \quad (2.9d)$$

where σ_1 , σ_2 and θ are values pulled from the `spreads1`, `spreads2` and `rotations` lists respectively. That matrix is used to calculate the Mahalanobis distance:

$$\begin{aligned} & (\mathbf{p} - \boldsymbol{\mu})^T \mathbf{A} (\mathbf{p} - \boldsymbol{\mu}) \\ &= \begin{bmatrix} p_1 - \mu_1 & p_2 - \mu_2 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{bmatrix} \begin{bmatrix} p_1 - \mu_1 \\ p_2 - \mu_2 \end{bmatrix} \\ &= a_{11} (p_1 - \mu_1)^2 + 2a_{12} (p_1 - \mu_1) (p_2 - \mu_2) + a_{22} (p_2 - \mu_2)^2 \end{aligned} \quad (2.10)$$

where \mathbf{p} is a vector holding the two physical property values in a mesh cell and $\boldsymbol{\mu}$ is a vector holding the means for the two physical properties. Those means are pulled from the `centres1` and `centres2` lists. The Mahalanobis distance appears in a Gaussian distribution f as follows:

$$f(\mathbf{p}) = \exp \left(-\frac{1}{2} (\mathbf{p} - \boldsymbol{\mu})^T \mathbf{C}^{-1} (\mathbf{p} - \boldsymbol{\mu}) \right). \quad (2.11)$$

See https://en.wikipedia.org/wiki/Gaussian_function#Two-dimensional_Gaussian_function and [Sun & Li \(2017\)](#) for more information on the mathematics for the `gauss` and `fcm` coupling options respectively.

Option 2

If you specify the variances and covariances (`variances1`, `variances2`, `covariances`) of the clusters then a covariance matrix is generated for each cluster:

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} \\ c_{12} & c_{22} \end{bmatrix} \quad (2.12)$$

where c_{11} , c_{22} and c_{12} are values pulled from the `variances1`, `variances2` and `covariances` lists respectively. The covariance matrix is inverted and used to calculate the Mahalanobis distance:

$$(\mathbf{p} - \boldsymbol{\mu})^T \mathbf{C}^{-1} (\mathbf{p} - \boldsymbol{\mu}). \quad (2.13)$$

See https://en.wikipedia.org/wiki/Mahalanobis_distance and https://en.wikipedia.org/wiki/Multivariate_normal_distribution#Density_function for more information on the mathematics.

2.5.5 Single property inversion

If you want to apply some additional regularization to a single physical property inversion, for example to apply some clustering to the recovered single physical property, then use `proptype1` to specify the single property and leave `proptype2` set to `null` (the default if absent).

2.6 Regions files

These files specify different regions within a wireframe model. The format is like at the end of a `.poly` file:

```
nreg ndim nat nvol
1 x1 [y1] z1 [attribute] [volume]
2 x2 [y2] z2 [attribute] [volume]
...
n xn [yn] zn [attribute] [volume]
```

where

- `nreg=n` is the number of regions (the number of lines that follow in the file)
- `ndim` is the number of dimensions (2 or 3)
- `nat` is the number of attributes (0 or 1)
- `nvol` is the number of volumes (0 or 1)
- the `(xi,[yi],zi)` coordinates specify points within each region.

Refer to the [TetGen](#) or [Triangle](#) documentation for further information on the `.poly` file format.

Any line beginning with `#` is a comment line and is ignored. Fully commented lines may not appear within the list of regions but comments may appear at the end of any data line.

The attribute and volume columns are only required if `nat=1` and `nvol=1` respectively. The volumes are used for specifying minimum volume constraints when meshing. The file format only allows one attribute. If you want more than one attribute it would be better to use the attributes to specify rock-unit ID values that can then get linked to the attribute values in a rock-units file: see program `rockunits2ele`.

2.7 Rock-units files

These files specify different attributes (e.g. physical property values) for different rock units. The format is as follows:

```
nunits nat
ID1 a11 a12 ...
ID2 a21 a22 ...
...
name1
name2
...
```

where

- `nunits` is the number of units
- `nat` is the number of attributes
- each row after the first specifies the unit ID and the attribute values for that unit:
 - `IDi` is the i^{th} unit ID (a unique integer)
 - `aij` is the j^{th} attribute for the i^{th} unit (e.g. a real value)
- the final several lines contain the attribute names:
 - `namej` is the name for the j^{th} attribute.

Any unique integer values can be used for the unit ID's. However, some programs will run faster if they are sequential from 1.

The names specified for the rock unit names may need to be set to exactly those required for a specific purpose. For example, [Section 3.3.5](#) lists possible names for physical properties needed when forward modelling; [Sections 5.4, 5.4.4 and 5.4.9](#) list possible names for physical properties and bounds when inverting for a surface-based model.

2.8 Optimization engine specification files

These files tell the inversion programs VIDI and DYN0 how to solve the problems (what optimization options to use).

Each line of the input file should be of the format

name value

where **name** is the name of some modelling parameter and **value** is the value for that parameter. The possible parameters and default values are listed below. If a parameter is not found in the input file then the default value is used for that parameter. Note that some of the parameters are for use in inversion and are ignored for forward modelling purposes.

Lines in the input file beginning with the # or ! character are ignored and can be used as comments for your own reference.

Name	Default	Brief description
engine	"local"	set to "local", "pso", "ga" or "mcmc" (the latter three are still under development)
LOCAL OPTIMIZATION OPTIONS		
gaussnewton	"f"	if "t" (true) then the approximate Hessian is used
usebounds	"f"	set to "t" (true) to perform a bound-constrained inversion
doquad	"f"	set to "t" (true) to perform a quadratic line search as well as bisection
maxstepsi	-1	max. # of model perturbations in first minimization (burn-in), set ≤ 0 to use maxsteps0
maxsteps0	2	max. # of model perturbations when far from misfit target
maxsteps1	-1	as for maxsteps0 but used when close to misfit target, set ≤ 0 to use maxsteps0
maxstepso	-1	max. # of model perturbations in last minimization (burn-out), set < 0 to use maxsteps1
maxstepsj	0	added to maxstepsi/0/1/o when in the joint inversion stages
omega1	2.0	target-normalized misfit that determines when maxsteps1 takes effect
cgmaxit	2000	maximum iterations for the CG algorithm when solving for the search direction
cgtol	1.0E-3	tolerance for the CG algorithm when solving for the search direction
lstol	1.0	tolerance on relative objective function value for the model perturbation line search
GLOBAL OPTIMIZATION OPTIONS		
multiobj	"t"	single- or multi-objective function inversion
discrete	"f"	continuous or discrete physical properties?
usex0	"t"	if "t" (true) then the initial model is used to initialize the global solver
maxit	0	maximum iterations of global optimization engine; set ≤ 0 for no maximum
nind	0	number of individual solutions, e.g. number of particles for PSO or population size for GA
find	2.0	size factor: if nind=0 then nind is set to $round(find \times npar)$
npar	0	number of parameters (length of each solution vector)
nobj	0	number of objective/goal functions
ncon	0	number of constraints
eps	1.0E-3	convergence tolerance
targ	0.0	function value target; stop when objective function falls below this
timelimit	0	run time limit in minutes; set ≤ 0 for no time limit
iverb	0	verbose printing to screen every iverb iterations; set ≤ 0 for no printing
iwrite	0	if > 0 then the output files are written every iwrite iterations
fiditer	0	file used for writing the objective function and solution every iverb iterations
fidbest	0	file used for writing the best objective function and related solution so far
writebest	f	if "t" (true) then the output files are written whenever an improved solution is found
GENETIC ALGORITHM OPTIONS		
separate	"t"	how to perform the crossover and mutation
fmutat	0.2	fraction of children created by mutation v.s. crossover; only used if separate is "t" (true)
fox	" "	a "fox" output file from a previous optimization used to initialize the current optimization
GA SELECTION OPTIONS		
selection	"tour"	the type of selection used: "tour" for tournament, "roul" for roulette wheel
ntourn	2	tournament size; only used if selection is "tour"
tourndup	"f"	set to "t" (true) to allow duplicates in the tournament population

Name	Default	Brief description
<code>coupling</code>	"all"	rules for tournament coupling: "all", "self" or "couples"
GA CROSSOVER OPTIONS		
<code>crossint</code>	"double"	the type of crossover for integer problems: "single", "double" or "uniform"
<code>crossreal</code>	"sbx"	the type of crossover for real problems: "blx" or "sbx"
<code>alpha</code>	0.5	BLX-alpha value for real crossover; see Deb et al. (2002)
<code>pcross</code>	0.9	SBX real crossover probability on [0.0,1.0]; see Deb et al. (2002)
<code>ncross</code>	20.0	n value for SBX real crossover; see Deb et al. (2002)
GA MUTATION OPTIONS		
<code>mutation</code>	"poly"	the type of mutation used: "uniform", "normal", "poly", "probe" or "litho"
<code>pmutat</code>	0.1	mutation probability on [0.0,1.0]
<code>sigma</code>	0.5	standard deviation for normal (Gaussian) mutation relative to bounds
<code>shrink</code>	0.5	shrinking factor for normal (Gaussian) mutation
<code>nmutat</code>	20.0	n value for polynomial real mutation; see Deb et al. (2002)
GA ELITISM OPTIONS		
<code>elitism</code>	"all"	the type of elitism used: "none", "some" or "all"
<code>nelite</code>	1	number of elite individuals to maintain between generations; single obj. problem only
<code>fpareto</code>	0.35	maximum fraction of population pulled from the first non-dominated front each generation
<code>chicrowd</code>	-1.0	if >0 then crowding distance is set HUGE for solution with misfit closest to <code>chicrowd</code>
<code>cutcrowd</code>	-1.0	if >0 then crowding distances are decreased for solutions with misfit > <code>cutcrowd</code>
GA CONVERGENCE OPTIONS		
<code>nstallobj</code>	50	number of generations across which to check for stall on objective space
<code>nstallpar</code>	50	number of generations across which to check for stall on parameter space
<code>stalltolobj</code>	1.0E-6	tolerance for stall check in objective space
<code>stalltolpar</code>	1.0E-6	tolerance for stall check in parameter space

2.8.1 General optimization options

engine

- Option `local` runs a Newton-style descent-based local optimization.
- Option `pso` runs a particle swarm global optimization strategy.
- Option `ga` runs a genetic algorithm global optimization strategy.

multiobj

- Only used with global optimization strategies, although multi-objective optimization is not yet ready for the PSO engine.
- If "t" (true) then the different terms in the objective function (e.g. misfit, model regularization and joint coupling) are kept separate and many solutions on the Pareto front are found.
- If "f" (false) then the different terms in the objective function are summed into a single objective and a single best solution is found.
- If there is only one term in the objective function then single-objective optimization is always performed.

2.8.2 Local optimization options

gaussnewton

- This parameter is only applicable to magnetics inversions with the `amp` data type or `aia` model type.
- Thorough testing has yet to be performed but preliminary testing suggests that the higher order terms in the Hessian calculation are important and therefore this parameter should be set to "f" (false).

lstol

- If Φ is the objective function, \mathbf{m} is the current model and $\delta\mathbf{m}$ is the current search direction (model perturbation) then the line search for the model perturbation scaling factor α stops once

$$\frac{\Phi(\mathbf{m} + \alpha\delta\mathbf{m})}{\Phi(\mathbf{m})} < \text{lstol}. \quad (2.14)$$

- If `lstol` is set to one then α is accepted for any reduction in the objective function.

2.8.3 Global optimization options

discrete

- If "f" (false) then the problem parameters are treated as continuous variables (between bounds).
- If "t" (true) then the problem parameters are treated as integers.

nobj

- This should always be equal to the number of objectives for the multi-objective problem, regardless of the value of **multiobj**.
- If **multiobj** is "f" (false) , meaning it is a single objective problem, then the **nobj** objectives are summed into the single aggregate objective that is minimized.
- If **discrete** is "t" (true) then the number of objectives is equal to the number of datasets.
- If **discrete** is "f" (false) then the number of objectives is equal to the number of datasets plus the number of physical properties plus the number of joint coupling pairs.

iwrite, writebest

- **iwrite** is currently only used for MCMC, PSO and GA.
- **writebest** is currently only used for PSO and GA.

2.8.4 Genetic algorithm options

separate, fmutat

- If **separate** is "t" (true, the default) then the crossover and mutation operators occur separately: $(1.0 - \text{fmutat}) \times \text{nind}$ children are generated from crossover, and $\text{fmutat} \times \text{nind}$ different children are generated from mutation, and those children are combined into the population of children.
- If **separate** is "f" (false) then **nind** children are generated from crossover and then those children are mutated.

coupling

- This parameter is only used if **selection** is "tour".
- If **coupling** is "all" then all couples (father-mother pairs) are allowed.
- If **coupling** is "self" then no self duplication (father = mother) is allowed.
- If **coupling** is "couples" then no duplicated couples are allowed.

sigma, shrink

- The standard deviation for normal (Gaussian) mutation is determined by the following equation:

$$\text{std} = \text{sigma} \times (1.0 - \text{shrink} \times (i - 1)/\text{maxit}) * (b_u - b_l)$$

where i is the current generation count, and b_l and b_u are the lower and upper bounds for a particular parameter respectively.

elitism, nelite, fpareto

- If **elitism** is "none" then no parents pass into the next generation.
- For single objective problems: if **elitism** is "some" then the best **nelite** parents pass into the next generation; if **elitism** is "all" then all parents and children are ranked together and the best pass into the next generation.
- For multi-objective problems, if **elitism** is "some" or "all" then **fpareto** is used. If **fpareto** = 1.0 then the new population for the next generation is filled with individuals from the first non-dominated front, then the second, third, etc., until the new population is completely filled. If **fpareto** < 1.0 then only that fraction of the individuals are taken from the first non-dominated front and the portion taken from later fronts follows a geometric reduction.

nstallobj, nstallpar, stalltolobj, stalltolpar

- Two of the convergence criteria are stalls in objective space and parameter space.
- These convergence tests are only performed for single objective problems.
- If the relative average change of the best single objective function value across **nstallobj** generations is below **stalltolobj** then the optimization stops.

- If the relative average change of the best solution (a 2-norm of the model parameters) across **nstallpar** generations is below **stalltolpar** then the optimization stops.
- Set **nstall*** \geq **maxit** or **nstall*** \leq 1 or **stalltol*** = 0.0 to avoid these convergence tests.

2.9 Unstructured block files

These files specify a number of blocks in a simple block model, destined for use in an unstructured mesh. The format is as follows:

```
n nbb ndim nat nvol
xc1 [yc1] zt1/zc1 dx1 [dy1] dz1 [phi1] theta1 [psi1] [a1] [v1]
xc2 [yc2] zt1/zc2 dx2 [dy2] dz2 [phi2] theta2 [psi2] [a2] [v2]
...
xcn [ycn] ztn/zcn dxn [dyn] dzn [phin] thetan [psin] [an] [vn]
```

where

- **n** is the number of blocks (the number of lines that follow in the file)
- **nbb** is the number of boundary blocks (usually 1 for simple models)
- **ndim** is the number of dimensions (2 or 3)
- **nat** is the number of attributes (0 or 1)
- **nvol** is the number of volumes (0 or 1)
- (**xc1**, **[yc1]**, **zt1**) specifies the LATERAL CENTRE OF THE TOP of any boundary blocks (*y* coordinates are not used in 2D)
- (**xc1**, **[yc1]**, **zci**) specifies the CENTROID of any non-boundary blocks
- the **dx**, **dy**, **dz** quantities define the dimensions of the block
- **phi**, **theta** and **psi** are the strike, dip and tilt angles respectively, in degrees, defined as in [Li & Oldenburg \(2000a\)](#) and [Lelièvre & Oldenburg \(2009\)](#) (only dip, **theta**, is required for 2D problems)
- **ai** is the attribute value for the i^{th} block
- **vi** is a volume for the i^{th} block (used as a maximum cell volume constraint for meshing programs [Triangle](#) and [TetGen](#)).

Alternatively, the block specification lines can be in this format:

```
A x1 x2 [y1 y2] z1 z2 [phi] theta [psi] [attribute] [volume]
```

where that “A” character must exist at the start of the line.

All boundary blocks should be specified first, before any non-boundary blocks. Hence, for the first **nbb** blocks, the *z* value supplied is assumed to be the lateral centre of the top of the boundary blocks; for the other blocks, the *z* value supplied is assumed to be that of the centroid of the non-boundary blocks.

The coordinates in the blocks file are assumed to have +*x* East, +*y* North, +*z* up for 3D and +*x* right, +*z* up for 2D.

Chapter 3

Forward modelling of geophysical data: FOGO (formerly FWD)

3.1 Program summary

FOGO is a program for **FO**rward modelling of **GeO**physical data. It provides support for the following data types:

- scalar gravity data (vertical component other single Cartesian component)
- gradiometry (tensor) gravity data
- total magnetic field data
- first-arrival seismic traveltimes

To calculate magnetic data with remanence you may have to perform two calculations, e.g. one for the induced field and another for the remanent component. Program `sum_node` can be used to combine the two data responses.

The model can be build on a rectilinear or unstructured mesh, or can involve unstructured wireframe surfaces representing contacts between rock units.

This program will run in parallel. Remember to set the `OMP_NUM_THREADS` environment variable. On a Mac you use the following command in the terminal window before running this program:

```
> export OMP_NUM_THREADS=value
```

Potential field modelling uses the methods of [Okabe \(1979\)](#). Seismic modelling uses the methods of [Lelièvre et al. \(2011\)](#).

3.2 Running the program

3.2.1 Command line usages:

```
fogo meshinp propinp datainp
fogo meshinp propinp datainp outroot
fogo meshinp propinp datainp outroot dosen
fogo meshinp propinp datainp outroot dosen ai
```

3.2.2 Command line parameters

`meshinp`

- A model discretization specification file.
- See the documentation on [model discretization specification files](#) for more information on the file format.

`propinp`

- A physical property specification file.
- See the documentation on [physical property specification files](#) for more information on the file format.

`datainp`

- The name of the geophysical data input file (the format is described below).

`outputroot`

- The names of the output files have this root (the output files are described below).
- If `outputroot` is absent then it is set to the root of the input file, that is, the input file name with extension and any path information stripped off: for example, if `inputfile` is set to `mydirectory/myfile.ext` then the output files will be named `myfile*` and placed in the current working directory.

`dosen`

- If set to `"t"` (true) then sensitivity information is calculated.
- The default is `"f"` (false).

`ai`

- Only used for a mesh-based model.
- Specifies the index of the attribute to use for the model.
- If `ai ≤ 0` then this input is ignored.
- If `ai > 0` then this input overrides all other specifications of what model to use (the attribute names in the model file header are ignored).

3.3 Data input file format

Each line of the input file should be of the format

`name value`

where `name` is the name of some modelling parameter and `value` is the value for that parameter. The possible parameters and default values are listed below. If a parameter is not found in the input file then the default value is used for that parameter. Note that some of the parameters are for use in inversion and are ignored for forward modelling purposes.

Lines in the input file beginning with the `#` or `!` character are ignored and can be used as comments for your own reference.

When specifying path/file names in the input file, place double quotes around them (e.g. `"../../data.txt"`). Single quotes should also work but this is machine dependent.

Name	Default	Brief description
PARAMETERS COMMON TO ANY DATATYPE		
<code>datatype</code>	<code>" "</code>	this must be the first parameter specified in the file
<code>datafile</code>	<code>" "</code>	file containing the data response (for inversion only)
<code>gamma</code>	1.0	multiplier on the data misfit term (for inversion only)
<code>lambdainit</code>	0.0	initial trade-off parameter value (for inversion only)
<code>chifact</code>	1.0	normalized target misfit (for inversion only)
<code>chitol</code>	0.05	relative tolerance on the target misfit (for inversion only)
<code>measure</code>	<code>"ell2"</code>	specifies the measure for the misfit (for inversion only)
<code>ekblomp</code>	2.0	the p-value for the Ekblom measure (for inversion only)
<code>ekblome</code>	1.0E-9	the epsilon value for the Ekblom measure (for inversion only)
<code>huberc</code>	<code>>> 0</code>	the c-value value for the Huber measure (for inversion only)
PARAMETERS SPECIFIC TO GRAVITY OR MAGNETICS DATA		
<code>obsfile</code>	<code>" "</code>	file containing the observation locations
<code>approx</code>	<code>"f"</code>	perform approximate modelling or not
<code>move</code>	<code>"f"</code>	allows you to copy the data to the x or z coordinate
<code>mtxroot</code>	<code>" "</code>	defines the file(s) containing the sensitivity matrix/matrices
<code>compmeth</code>	<code>"none"</code>	compression method; set to <code>"none"</code> , <code>"noco"</code> or <code>"wave"</code>
<code>compdir</code>	<code>"row"</code>	compression direction; set to <code>"row"</code> or <code>"col"</code>
<code>wavelet</code>	<code>"null"</code>	type of wavelet compression: set to <code>"daub1"</code> – <code>"daub6"</code> , <code>"symm4"</code> – <code>"symm6"</code> , or <code>"null"</code>
<code>tol</code>	0.0	relative wavelet threshold
<code>level</code>	0.0	value added to the observed data before inverting (for inversion only)
<code>autolevel</code>	<code>"f"</code>	if <code>"t"</code> (true) then an attempt is made to automatically level the data (for inversion only)
PARAMETERS SPECIFIC TO GRAVITY DATA		
<code>comp</code>	<code>"z"</code>	specifies which gravity component to use; set to <code>"x"</code> , <code>"y"</code> or <code>"z"</code>)
PARAMETERS SPECIFIC TO GRAVITY GRADIOMETRY DATA		
<code>comps</code>	<code>tttttt</code>	specifies which tensor components to use

Name	Default	Brief description
PARAMETERS SPECIFIC TO MAGNETICS DATA		
magdata	""	magnetics data type; set to "tmi", "xyz" or "amp"
magmodel	""	magnetics model type; set to "sus", "pst" or "atp"
igeo	0.0	geomagnetic field inclination in degrees
dgeo	0.0	geomagnetic field declination in degrees
sgeo	1.0	geomagnetic field strength in nT
idir	0.0	measurement inclination in degrees
ddir	0.0	measurement declination in degrees
PARAMETERS SPECIFIC TO FIRST ARRIVAL TRAVELTIME DATA		
sourcesfile	""	.node file specifying the source locations
receiversfile	""	.node file specifying the receiver locations
combosfile	"null"	.ele file specifying the source/receiver combinations, or set to "all" or "match"
recip	"f"	set to "t" (true) to perform reciprocal modelling
radius	0.0	the initialization radius in the fast marching
thresh	0.0	a threshold on the sensitivity values
tracemode	"none"	specifies the type of tracing to perform (if any)
sloray	0.0	homogeneous slowness value to remove when calculating traveltimes along ray paths
senfullflag	"f"	set to "t" (true) to use a full sensitivity matrix instead of sparse
writettimes	"t"	if "t" (true) then the traveltimes are written to an output file
writetypes	"f"	if "t" (true) then the traversal types are written to an output file
PARAMETERS SPECIFIC TO MUOGRAPHY DATA		
method	"sph"	"sph" pr "cart" for spherical or Cartesian formulations respectively
obsfile	" "	muography observation locations (see below for specific file format)
receiversfile	" "	a .node file specifying the receiver location(s)
nsub	-1	Used differently depending on the method (see below for more information)
mtxroot	""	defines the file(s) containing the sensitivity matrix/matrices
level	0.0	value added to the observed data before inverting (for inversion only)
autolevel	"f"	if "t" (true) then an attempt is made to automatically level the data (for inversion only)
alphac	0.0	weighting for additional inversion parameter for solving for the level

3.3.1 Units

If all spatial quantities are in m (metres) then:

- For gravity data, if density is in g/cm^3 (grams per cubic centimetre) then the data is in $mGal$ (milligal).
- For gravity gradiometry data, if density is in g/cm^3 (grams per cubic centimetre) then the data is in E (eotvos).
- For magnetic data, if susceptibility is S.I. (unitless), or magnetization is expressed as an effective susceptibility, then the data is in nT (nanotesla).
- For seismic data, if slowness is in s/m (seconds per metre) then the data is in s (seconds).
- For muography data, if density is in g/cm^3 (grams per cubic centimetre) then the data is also in g/cm^3 .

3.3.2 Specifying the type of data response

datatype

- Set to "fat" for first-arrival traveltime data.
- Set to "gz" for gravity data (vertical component or other single Cartesian component).
- Set to "gg" for gravity gradiometry data.
See more below under parameter datafile for specifications on the data file format for this case.
- Set to "mag_tmi_sus" for total field magnetic intensity data with susceptibility model.
- Set to "mag_tmi_pst" for total field magnetic intensity data with Cartesian vector magnetization model.
- Set to "mag_tmi_aid" for total field magnetic intensity data with spherical vector magnetization model.
- Set to "mag_xyz_sus" for multi-component magnetics data with susceptibility model.
- Set to "mag_xyz_pst" for multi-component magnetics data with Cartesian vector magnetization model.

- Set to "mag_xyz_aid" for multi-component magnetics data with spherical vector magnetization model.
- Set to "mag_amp_sus" for magnetic amplitude data with effective susceptibility model.
- Set to "muog" for muography data.

igeo, dgeo, sgeo, idir, ddir

- Inclination, declination and strength of the geomagnetic inducing field and measurement direction.
- Enter the angles in degrees and strength in nT.
- For total field data set the measurement direction equal to the inducing field:
 $\text{idir} = \text{igeo}$
 $\text{ddir} = \text{dgeo}$.
- dgeo and ddir are ignored for 2D problems.
- For 2D problems, any magnetization component in the strike direction (normal to the plane of the vertical section) will not contribute to the data measurements, assuming that the 2D model is constant to infinity in the strike direction. Any magnetic data component measured in the strike direction will also be zero. Hence, **for 2D problems, igeo and idir should be specified as inclinations projected onto the 2D section.** This can be achieved as follows:

$$\vec{v}_{proj} = \vec{v}_{geo} - (\vec{v}_{geo} \cdot \hat{n}) \hat{n}$$

where \vec{v}_{geo} is the geomagnetic field vector, \vec{v}_{proj} is the projected vector and \hat{n} is a unit vector normal to the 2D section plane. To calculate \vec{v}_{geo} you can use:

$$\begin{aligned}\vec{v}_{geo,x} &= \text{sgeo} \cos(\text{igeo}) \cos(\text{dgeo}) \\ \vec{v}_{geo,y} &= \text{sgeo} \cos(\text{igeo}) \sin(\text{dgeo}) \\ \vec{v}_{geo,z} &= \text{sgeo} \sin(\text{igeo})\end{aligned}$$

which is in a +z-down system. For vertical 2D section planes (horizontal normal vector), the projected inclination, θ_{proj} , can be calculated using:

$$\begin{aligned}r &= \sqrt{\vec{v}_{proj,x}^2 + \vec{v}_{proj,y}^2} \\ z &= \vec{v}_{proj,z} = \vec{v}_{geo,z} \\ \theta_{proj} &= \arctan(z/r).\end{aligned}$$

Furthermore, if you know the declination angle of the 2D section, ϕ_{sect} , then the projected inclination, θ_{proj} , can be calculated using:

$$\begin{aligned}r &= \text{sgeo} \cos(\text{igeo}) \cos(\Delta\phi) \\ z &= \text{sgeo} \sin(\text{igeo}) \\ \theta_{proj} &= \arctan(z/r) = \arctan\left(\frac{\sin(\text{igeo})}{\cos(\text{igeo}) \cos(\Delta\phi)}\right) = \arctan\left(\frac{\tan(\text{igeo})}{\cos(\Delta\phi)}\right)\end{aligned}$$

where $\Delta\phi$ is the angle made between the geomagnetic field direction and the 2D section, e.g.

$$\Delta\phi = \text{dgeo} - \phi_{sect}$$

but be careful when making such a calculation if the two angles straddle the 0/360 degree fence.

comps

- If the i^{th} character of the value string is "t" then the i^{th} tensor component is calculated in the forward modelling or is used in the inversion.
- If inverting, all tensor components are assumed to exist in the data file, even if all are not being inverted.
- See more below under parameter datafile for specifications on the data file format for gravity gradiometry data.

approx

- If "t" (true) then the triangular or tetrahedral cells in a mesh are approximated as dipoles (circles or spheres).

move

- If this parameter is set to "x" or "z" then the data are copied to the x or z coordinate column of the output `.vtu` file for viewing purposes.
- This **only** applies to the `.vtu` output file.
- The copying is only performed if the x or z observation values are identical, e.g. for horizontal synthetic airborne data or vertical down-hole data.

3.3.3 Data observations

obsfile

- This file is used for gravity or magnetics data only. For traveltime data, use `combosfile`.
- A `.node` file defining the data observations.
- For synthetic modelling, you can use program `make_obs` to generate this file.
- When forward modelling gravity or magnetics data, the observation locations are read from this file.
- When inverse modelling, this file is ignored and the observation locations and data responses read from the `datafile`.

combosfile

- This file is used for traveltime data only. For gravity or magnetics data, use `obsfile`.
- This parameter can specify a `.ele` file with two columns of indices that define the source/receiver pairs.
- You can use program `make_obs` to generate this file.
- Specify "all" if you want to use all source/receiver combinations.
- Specify "match" if you want to match the i^{th} source with the i^{th} receiver.
- When forward modelling seismic traveltime data, the source/receiver pairs are read from this file.
- When inverse modelling, the `combosfile` is ignored and the source/receiver pairs and traveltimes are read from the `datafile`.

datafile

- A `.node` or `.ele` file (depending on the data type) containing the data response.
- When forward modelling, this file is ignored.
- When inverse modelling, observation locations (source/receiver pairs for traveltime data) are read from this file.
- For tensor data:
 - The first 6 columns (node attributes) in the data file are the components in this order:
 $xx - xy - xz - yy - yz - zz$
 - If inverting, the uncertainties should be in columns 7 through 12 in that same order.
 - **The coordinate system for the tensor data components is always a right-handed +z down system (+x north, +y east).** Hence, if you have your data in a right-handed +z up system (+x east, +y north) then you will need to reorder your data columns as follows:
 $yy - yx - yz - xx - xz - zz$
 and you will also need to change the sign of the xz and yz measurements, but please see the final item below.
 - If, on the other hand, you have your data in a left-handed +z down system (+x east, +y north) then you will still need to reorder your data columns as above but you do not need to change the sign of any measurements.
 - For complete clarity, the components should be in this order (E=east, N=north, D=down):
 $NN - NE - ND - EE - ED - DD$
 - Note that the NE component equals the EN component, so does not appear in the list of components specified above.
 - If you are at all unclear about the coordinate system and need to convert, especially any uncertainty about having to change the sign on any measurements, then I highly suggest you invert all components separately before continuing with an inversion of multiple components together. When inverting components separately, compare the results, make sure they are all consistent, particularly the sign of the densities in the results, and make sure all inversions converge as expected. If there are any inconsistencies then fix as necessary before inverting components together.

3.3.4 Specifying the modelling grid

zdir

- For 3D, set **zdir** > 0 to specify +z up, +x East, +y North;
set **zdir** < 0 to specify +z down, +x North, +y East.
For 2D, the +x axis is always to the right and **zdir** only specifies the +z direction. The y axis is along-strike.
- The **zdir** parameter in the **meshinp** file (see [Section 2.3](#)) is applied to the data file. Hence, the code assumes that the coordinate system for the mesh and data files are consistent.
- The **zdir** parameter only affects the observation locations: parameters **datafile**, **obsfile**, **sourcesfile**, **receiversfile**. The coordinate system for tensor gravity components and vector magnetometry are always +z down.

3.3.5 Specifying the model

modelfile, **facetunitsfile**, **regionunitsfile**, **regionsfile**

- These are specified in the **meshinp** file.
- The required attribute name is either **Density**, **MagSus**, **Slowness** or **Conductivity** depending on the data type. The case is important. This can be present in the **modelfile** or **regionunitsfile** (explained further below).
- Remember that the property values are scaled using the **mmul** and **madd** parameters (specified in the **propinp** file) to obtain the values used in the modelling.
- For a voxel model:
 - Unless **ai** is specified, the required attribute name should be present in either the **modelfile** or **regionunitsfile** but not both. If it is found in both then an error is thrown and the program quits.
 - Specifying **ai** overrides the required attribute name and the following three bulleted items are irrelevant.
 - Placing the required attribute in the **modelfile** allows you to have different physical property values in every cell.
 - Placing the required attribute in the **regionunitsfile** is an alternate approach if you want to have different physical property values in several homogeneous regions of a model. In this case, the first attribute in the **modelfile** is assumed to contain rock-unit IDs that are linked to physical properties via the **regionunitsfile**. If no attribute columns exist in the **modelfile** then rock-unit IDs of 1 are used.
 - If the required attribute name is not found in the **modelfile** or **regionunitsfile** then the first attribute in the **modelfile** is used instead, regardless of any attribute names in the file. If there are no attributes in the **modelfile** then: a zero-valued model is used if **dosen** is "t" (true), or a unit-valued model is used if **dosen** is "f" (false).
- For an unstructured surface model:
 - The required physical property attribute must be present in the **regionunitsfile**.
 - For traveltimes data (straight or curved rays), the surface model is meshed before performing the fast marching solution. The slowness in every cell of the mesh is determined by taking the region attribute values in the **regionsfile**, which are assumed to be region unit IDs, and matching those to the information in the **regionunitsfile**. The coordinates in the **regionsfile** are used by the meshing program to specify different regions in the surface model. The 3D problem is not yet supported.
 - For gravity or magnetics data, an intermediate mesh is not required and the data are calculated directly from the geometry of the surfaces in the wireframe model. To do so, the program must know which regions are on each side of every surface facet:
 - * The first attribute in the **modelfile** should specify facet IDs. If the **modelfile** contains no attributes then default facet IDs of 1 are used.
 - * Attributes **Region1** and **Region2** must be present in the **facetunitsfile**. Those attributes specify region unit IDs on each side of the surface facets. For each facet, the facet ID is taken from the **modelfile** and matched to an ID in the **facetunitsfile**, which provides the two region unit IDs on either side of the facet. Physical property values on either side of the facet are obtained from the **regionunitsfile** file. The difference (second minus first) is used to calculate the gravity or magnetic response. The ordering of the nodes in the facets is important ...
 - * In 2D, if the nodes define an inward normal, using a right-hand-rule that crosses the vector along the line element facet (edge) and the y-axis, then the 1st region should be inside and the 2nd region outside. For example, if you have a single blob defined by a polygonal outline, a clockwise node ordering of the polygon in a +z down

coordinate system will define inward normals, in which case the 1st region would be the blob and the second the background.

- * In 3D, if the nodes define an inward normal, using a right-hand-rule around the nodes as ordered, then the 1st region should be outside and the 2nd region inside. Another way to think of this is that a vector pointing from the 1st region into the 2nd region (across some face) should point in the same direction (i.e. positive dot product) as the normal vector defined by the node ordering.
- * Program `surface_normals` can be used to help order the nodes in a wireframe surface model to aid with forward modelling.

3.3.6 Data misfit and trade-off parameters

`measure`

- set to "ell2" for the standard, sum-of-squares, ℓ_2 norm
- set to "ekblom" for the Ekblohm measure
- set to "huber" for the Huber measure

`gamma`

- This parameter is a user-defined scalar that multiplies the data misfit term during an inversion.
- The value does not change during the inversion.

See

`lambdainit`

- In an inversion, the trade-off parameter(s) multiply the data misfit term(s).
- This parameter specifies the initial value of all trade-off parameters in an inversion.
- If this parameter is greater than zero then it takes precedence over the `lambdainit` parameter in the main input file for a VIDI inversion (see [Section 4.4](#)).

3.3.7 Sensitivity matrices and compression

`mtxroot`

- For linear forward problems: when forward modelling, the sensitivity matrix is usually calculated and written to file(s); when inverting, the sensitivity matrix is usually read from file(s).
- If `mtxroot="null"` or `=""` then: the sensitivity matrix is not written to file(s) in the forward modelling; the sensitivity matrix is calculated in an inversion, instead of being read from file(s).
- The files named `mtxroot_*.mtx` hold the sensitivity matrices (there are multiple files for tensor data).
- For scalar gravity data, and the magnetics problem with a susceptibility model and a single data component, there is only a single sensitivity matrix and the file is called `mtxroot.mtx`.
- For tensor gravity data, there are 6 matrices and the files are called `mtxroot_xx.mtx`, `mtxroot_xy.mtx`, etc. When forward modelling with `dosen` set to "t" (true), all of these sensitivity matrices are calculated and written to the different files.
- For the "amp" and "atp" magnetics model types with a single data component, there are 3 matrices and the files are called `mtxroot_x.mtx`, `mtxroot_y.mtx` and `mtxroot_z.mtx`. These should be in a +z down coordinate system. When forward modelling with `dosen` set to "t" (true), there is currently no way to calculate all of these sensitivity matrices at once. Instead, you have to run FOGO three times with the "sus" model type, and each time specify a different Earth's field direction for the three different Cartesian directions:
 - x-direction = northing: `dgeo = 0`, `igeo = 0`
 - y-direction = easting: `dgeo = 90`, `igeo = 0`
 - z-direction = down: `dgeo = 0`, `igeo = 90`.

However, it is probably easier to just run an inversion and use the `mtxroot` parameter in the inversion input file: see [Section 4.4.11](#).

- For the "pst" magnetics model type, there are 3 matrices and the files are called `mtxroot.p.mtx`, `mtxroot.s.mtx` and `mtxroot.t.mtx`. When forward modelling with `dosen` set to "t" (true), there is currently no way to calculate all of these sensitivity matrices at once. Instead, you have to run FOGO three times with the "sus" model type, and each time specify a different Earth's field direction for the three different components. The p-component should be in the direction of the Earth's field and the s- and t-components perpendicular to that, and all three orthogonal. A simple way to achieve this is to set the s-component direction to have the same declination as the Earth's field but with inclination rotated 90 degrees, then the t-component is horizontal and has declination rotated 90 degrees away from that of the Earth's field. For example:

- p-direction: `dgeo = 14.9, igeo = 64.8`
- s-direction: `dgeo = 14.9, igeo = -25.2`
- t-direction: `dgeo = 0, igeo = 104.9`.

However, it is probably easier to just run an inversion and use the `mtxroot` parameter in the inversion input file: see [Section 4.4.11](#).

compmeth

- Set to "none" for no sensitivity compression.
- Set to "noco" for compression by reduced precision.
- Set to "wave" for wavelet compression, which uses parameters `wavelet` and `tol`.

compsdir

- Set to "row" for sensitivity compression along rows (the most sensible choice for rectilinear meshes).
- Set to "col" for sensitivity compression along columns (the only sensible choice for unstructured meshes).
- **Too much compression of a sensitivity matrix can lead to spurious artifacts in models recovered from inversion. Hence, whenever sensitivity compression is used it is important to analyze the forward modelling errors associated with the compression and compare the magnitude of those errors against the uncertainties assigned to the observed data.**

wavelet

- If set to "null" then the default is used, which is "daub2".

tol

- For wavelet compression, wavelet coefficients with absolute value less than $|w_{max}| * tol$ are set to zero, where w_{max} is the coefficient with the largest absolute value in the row or column being compressed.

3.3.8 Seismic traveltime modelling options

recip

- If set to true ("t") then the sources and receivers are switched.
- The fast marching method will run faster the fewer sources you use
- (the number of receivers has a much smaller impact on the solution time).
- The data should be identical (within numerical error) due to the reciprocity theorem.

radius

- If `radius > 0` then traveltimes at any nodes within that radius are calculated as slowness-times-distance using the slowness in the cell containing the source. This assumes homogeneity in the near-source region and helps reduce the modelling error.
- If $|radius| > 0$ then any rays being back-traced from receiver to source are traced directly to the source once they get within `abs(radius)` distance from the source. This is required because small numerical errors in the ray tracing means that rays may not actually hit the source exactly.
- When forward modelling, it is best to set `radius > 0` and as large as possible as allowed for my your model such that the assumption of homogeneity still holds. However, this assumes homogeneity around the source so you may have to be careful when inverting with `radius > 0`.
- This parameter is not used when `tracemode` is `straight`.

tracemode

- If set to **"none"** then no ray-tracing is performed.
- If set to **"straight"** then straight rays are traced directly between source and receiver (make sure **dosen** is **"t"** (true) in this case).
- If set to **"source"** then rays are back-traced through the traveltimes field and at sonic points it takes the direction that points the closest towards the source.
- If set to **"straightest"** then rays are back-traced through the traveltimes field and at sonic points it takes the direction that is most similar to the previous ray direction.
- If set to **"exhaustive"** then rays are back-traced through the traveltimes field and at sonic points all possible ray paths are tested and only the shortest path accepted (this option has the potential to be very time consuming).
- If tracing is performed then the traveltimes at the receivers are calculated as the traveltime integral along the trace path.
- If tracing is not performed then traveltimes are interpolated at receiver locations. The data will be about the same.
- The ray tracing in 3D is somewhat problematic. In 2D it is absolutely fine.
- If all you care about are the nodal traveltimes across the grid then you don't need to worry about this parameter (set to **"null"** or remove from input file).

writetypes

- The travel types are integer values that specify what kind of waves (traversal, head, diffraction) happened just before hitting a node:
 - 0 corresponds to near-source initialization,
 - 1 traversal along a tetrahedral edge element or triangular face (i.e. a 1D element),
 - 2 traversal across a tetrahedral face or triangular cell (i.e. a 2D element),
 - 3 traversal through a tetrahedral cell (i.e. a 3D element - only used in 3D).

sloray

- If rays are traced then this value is removed from the slowness model when calculating the traveltimes along the ray path. The ray paths themselves are still determined using the original slowness model.
- This can be helpful if you want to extract an anomalous component of the data.

3.3.9 Muography traveltime modelling options

The formats of the input files have been taken directly from the formats of the information given to me when I worked on muography data for the first time, with Barnoud et al. We can adjust these formats as desired in the future.

method

- The spherical formulation (**sph**) should be used for any production work.
- The **cart** option was used during research to provide a check of the **sph** option, but the **sph** option is superior.

obsfile

- This is a column-based file with each line of the format

$$x \ y \ z \ \phi \ \theta$$
 where the first three columns specify the Cartesian coordinates of the corresponding receiver, and ϕ and θ define the azimuthal and altitude angles at the centre of the bin (solid angle).
- The **datafile** should have two additional columns:

$$x \ y \ z \ \phi \ \theta \ d \ \sigma$$
 where d is the data value and σ is an assigned uncertainty.
- ϕ is the azimuthal angle in degrees from 0 towards north and positive east of north, i.e. our standard definition for declination.
- θ is the elevation in degrees from 0 horizontally to 90 vertically, i.e. opposite sign to our standard definition for inclination.

receiversfile

- This `.node` file should contain three coordinate columns for the receiver locations, plus two attribute columns specifying $\Delta\phi$ and $\Delta\theta$, the bin solid angle width and height respectively.
- There will be duplicated coordinate information in the `obsfile` and `receiversfile`. It is important that this information matches: each set of receiver coordinates in `obsfile` must match a set of receiver coordinates in `receiversfile`.

`nsub`

- If the method is "sph" then this specifies the number of integration sub-bins across a single data bin (solid angle).
- If the method is "cart" then this specifies the number of times to subdivide each mesh cell edge.
- If a negative value is provided and the method is "sph" then a default value of 128 is used for `nsub`.
- If a negative value is provided and the method is "cart" then a default value of 3 is used for `nsub`.

3.4 Output files

The output files are named `outroot*` with various suffixes and extensions, as listed below. The output file directory is the directory from which the program is being run, unless specified differently by adding path information to `outroot`.

`*.node` OR `*.ele`

- Contains the calculated data as the first attribute, named `VerticalGravity`, `GravityXComponent`, `GravityYComponent`, `GravityTensor`, `TotalFieldMag` or `TravelTime` depending on the data type.
- A `.ele` file is used for traveltime data and a `.node` file for all others.
- For traveltime data, the two index columns in the `.ele` file specify the source/receiver pairs for each traveltime.

`*.vtu`

- As above but for viewing the data response in ParaView.

`*_sen.vtu`

- Holds the model and another cell attribute equal to the ℓ_2 norm along each sensitivity matrix column (each cell has a value corresponding to a specific matrix column).
- Only written if parameter `dosen` is set to true ("t").

For traveltime data, the following additional files are written:

`*_traces.*`

- Hold seismic traces for each source-receiver pair.
- Only written if parameter `tracemode` is not "none".

`*_ttimes.vtu`

- Only written if either parameter `writettimes` or `writettypes` is "t" (true).
- If `writettimes` is "t" (true) then this file holds the traveltimes at all grid nodes in the modelling grid.
- If `writettypes` is "t" (true) then this file holds the traversal types at all grid nodes in the modelling grid:
 - 0 means the node was initialized with a traveltime
 - 1 means a traversal along a tetrahedral edge element or triangular face
 - 2 means a traversal across a tetrahedral face or through a triangular cell
 - 3 means a traversal through a tetrahedral cell (only used in 3D)

Chapter 4

Mesh-based physical property and lithological inversion: VIDI (formerly VINV)

4.1 Program summary

VIDI (**V**oxel**I**zed **D**iscretization) is a program for mesh-based inversion that determines the physical property values or lithologies inside the mesh cells. This is a flexible, modular and highly functional program:

- The discretization is voxellized (many space-filling cells) on a 2D or 3D rectilinear or unstructured (triangular or tetrahedral) mesh. See [Section 4.4.2](#) for related input file parameters.
- Two different inversion approaches are available that use very different optimization algorithms:
 - the inversion can treat the model values as physical properties on continuous, but possibly bounded, ranges
 - a lithological inversion can be designed such that only specific physical property values are allowed for each of the known or assumed rock types.
- The supported geophysical data types are:
 - vertical component (scalar) gravity data
 - gradiometry (tensor) gravity data
 - total magnetic field data
 - magnetic amplitude data
 - first-arrival seismic traveltimes.
- Single or multiple data-types can be inverted (independent or joint inversion). See [Section 4.4.8](#) for related input file parameters. Joint inversion can include:
 - multiple geophysical data types responsive to a single physical property
 - a single geophysical data type responsive to multiple physical properties
 - multiple geophysical data types responsive to multiple physical properties.

There are various possible joint coupling strategies for any pair of physical property models:

- explicit linear relationship (a specific mathematical relationship between two physical properties)
- implicit linear relationship (based on the concept of correlation from statistics)
- structure-based coupling using the cross-gradient measure
- petrophysical coupling using fuzzy c-means to specify clusters
- petrophysical coupling using a combination of Gaussian functions.
- There are many options for regularization. See [Section 4.4.5](#) for related input file parameters. Regularization includes the traditional smallness and smoothness/roughness measures with general norms. The smoothness axes can be rotated to any arbitrary orientations. See [Section 4.5.2](#) for a mathematical description of the model objective function.
- For magnetic inversion there are several possible forms:
 - inversion of total field data for a susceptibility model (a scalar quantity in each mesh cell)
 - inversion of magnetic amplitude data for an effective susceptibility model
 - inversion of total field data for a magnetization model (a vector quantity in each mesh cell)

For the latter there are further options for the format of the magnetization model:

- Cartesian components in arbitrary directions (typically one is in the direction of the Earth's field)
- spherical/polar coordinates, i.e. magnetization amplitude plus orientation angle(s).

See [Section 4.4.4](#) for related input file parameters.

- There are many options for controlling the minimization. See sections [4.4.9](#) and [4.4.10](#) for related input file parameters.

This program will run in parallel. Remember to set the `OMP_NUM_THREADS` environment variable. On a Mac you use the following command in the terminal window before running this program:

```
> export OMP_NUM_THREADS=value
```

4.2 Running the program

4.2.1 Command line usages

```
vidi inputfile
vidi inputfile outputroot
vidi inputfile outputroot sensibility
vidi inputfile outputroot sensibility restart
vidi inputfile outputroot sensibility restart seed
```

4.2.2 Command line parameters

inputfile

- The name of the input file (the format is described below).
- See the note at the start of [Chapter 2](#) regarding relative file paths.

outputroot

- The names of the output files have this root (the output files are described below).
- If `outputroot` is absent then it is set to the root of the input file, that is, the input file name with extension and any path information stripped off: for example, if `inputfile` is set to `mydirectory/myfile.ext` then the output files will be named `myfile*` and placed in the current working directory.

sensibility

- Set to "t" (true) to perform sensibility checks on the input files without actually running the inversion.
- The inversion will stop after calculating the misfit and model objective measures for the initial model. This can be helpful if you simply want to calculate that information for a particular model.

restart

- Set to "t" (true) to restart an inversion.
- This simply appends the output to any existing `.log` and `.aux` output files. It does not automatically determine which trade-off value to start from or which initial model to use: you must specify that information manually in the input files.

seed

- Set to some integer value to use as the seed for random number generation.
- This is only important if global optimization is used, not for local optimization.

4.3 Outputs

4.3.1 Output files

The output files are named `outroot*` with various suffixes and extensions, as listed below. The output file directory is the directory from which the program is being run, unless specified differently by adding path information to `outroot`.

There are many possible output files. For each dataset there will be predicted data files written in `.node` or `.ele` format, and in `.vtu` format. Those file names will contain the geophysical data type name, e.g. `"vertical_gravity"` or `"total_field_mag"`, and the data residuals calculated as predicted data minus observed data, divided by the data uncertainties.

If there are multiple physical property models or multiple model components involved, e.g. for a magnetization inversion or a joint inversion, the output files named `outputroot_models.*` contain those models; for a single dataset inversion with a single physical property model, those files are named similarly but with `"models"` replaced by the physical property name(s), e.g. `"density"` or `"magsus"`. The contents of the output files should be easy to assess by these naming strategies.

For joint inversions, the output `outputroot_models.*` file contains the physical property models and also the joint coupling constraint vector information. If there is only a single joint coupling measure then attribute `"UnscaledConstraintVector"` contains the constraint vector values before any weighting, and `"ScaledConstraintVector"` is multiplied by the `wjvalue` but **not** any weights specified in the `wjfile`. If there is more than one joint coupling measure then attribute `"CombinedConstraintVector"` contains a sum of the constraint vectors weighted by the corresponding `wjvalue` and `rho` values but **not** any weights specified in the `wjfile`.

For a magnetization inversion, the model names are as follows:

- **MagSus** is the effective magnetic susceptibility (vector magnitude divided by Earth's field strength)
- **MagP**, **MagS** and **MagT** are the vector components with the p-component in the direction of the Earth's field and the other two perpendicular.
- **MagST** is the RMS of the s- and t-components.
- **MagX**, **MagY** and **MagZ** are the vector components in a +z-down coordinate system (+x is North and +y is East).
- **MagQ** is an approximate Koenigsberger ratio calculated as the RMS of the s- and t-components divided by the absolute value of the p-component.
- **MagInc** and **MagDec** are the inclination and declination angles: inclination is zero at horizontal and positive below horizontal; declination is zero to the North and positive East of North.

For inversion on rectilinear meshes, the files containing the recovered models are named `*.txt` and contain multiple columns of values with a first header line indicating the model attributes in each column. These files can not be read directly by the UBC-GIF model visualization program `MeshTools3d`. You will need to extract a single column of the file into another file: you can use program `xyz2xyz` for this purpose.

4.3.2 Log file and terminal output

All information displayed in the terminal/command window during execution is written to the output log file. Below is a brief description of some of the items displayed:

- **totit** - the total iteration number increments each time the lambda trade-off parameter(s) change(s)
- **stage** - the stage of a joint inversion (the coupling weights are increased each stage)
- **lambit** - like **totit** but it resets to zero when **stage** increments
- **objfun** - the total objective function value
- **lambda1** (and **lambda2** if a joint inversion) - the trade-off parameter value(s)
- **omega1** (and **omega2** if a joint inversion) - omega is equal to the misfit divided by the target misfit
- **norm1** (and **norm2** if a joint inversion) - the model objective function value(s)
- **gtg** - the gradient norm
- **slope** - the slope of the objective function in the direction of the model perturbation
- **alpha** - the length of model perturbation taken
- **nact** - the number of active bound constraints (the number of model parameters that have hit their bounds)
- **ncgi** - the number of iterations taken in the conjugate gradient algorithm when calculating the model perturbation direction
- **rcg** - the residual for the conjugate gradient solution for the model perturbation.

Additional items are displayed for joint inversions:

- **alphaj** - the **alpha** value used

- `rho ...` - the current value(s) of the `rho` parameter(s) being heated
- `jnorm` - the sum of the joint coupling measure(s) after all multipliers (`alphaj`, `rho` and joint coupling weights) have been applied
- `jnormi ...` - the individual joint coupling measure(s) with joint coupling weights applied but without other multipliers applied (`alphaj` and `rho`).

4.4 Input file format

Each line of the input file should be of the format

```
name value
```

where `name` is the name of some inversion parameter and `value` is the value for that parameter.

Lines in the input file beginning with the `#` or `!` character are ignored and can be used as comments for your own reference.

When specifying path/file names in the input file, place double quotes around them (e.g. `"../mesh.txt"`). Single quotes should also work but this is machine dependent.

The possible parameters and default values are listed in the table below. The capitalized headings in the table link to sections where the parameters are discussed in more detail.

Name	Default	Brief description
<code>revision</code>	0	a code revision number
MESH INFORMATION		
<code>meshinp</code>	""	a model parameterization specification file
PHYSICAL PROPERTY INFORMATION		
<code>propinp</code>	""	input file for a particular physical property
DATA-RELATED INFORMATION		
<code>datainp</code>	""	input file for a particular geophysical data set
<code>normalized</code>	"t"	if "t" (true) then the data misfit term(s) are normalized by the number of data
<code>normalizedc</code>	"t"	if "t" (true) then the data misfit term(s) are normalized by the <code>chifact</code> value(s)
REGULARIZATION		
<code>normalizem</code>	"f"	if "t" (true) then the regularization terms are normalized by the mesh volume
<code>rotate</code>	"f"	set to "t" (true) to rotate the smoothness axes
<code>maskfile</code>	""	file that masks the regularization across the mesh
<code>maskindex</code>	""	attribute index for the <code>maskfile</code>
<code>wmfile</code>	""	file containing smoothness weights and axis rotation information
<code>wmindex</code>	0	attribute index for the <code>wmfile</code> smoothness weights file
<code>wxindex</code>	0	attribute index for the <code>wmfile</code> for cell-centred smoothness weights in x-direction
<code>wyindex</code>	0	attribute index for the <code>wmfile</code> for cell-centred smoothness weights in y-direction
<code>wzindex</code>	0	attribute index for the <code>wmfile</code> for cell-centred smoothness weights in z-direction
<code>wnindex</code>	0	an alias for <code>wxindex</code> ("n" for north)
<code>weindex</code>	0	an alias for <code>wyindex</code> ("e" for east)
<code>wvindex</code>	0	an alias for <code>wzindex</code> ("v" for vertical)
<code>strikeindex</code>	0	attribute index for the <code>wmfile</code> for the strike rotation angle
<code>dipindex</code>	0	attribute index for the <code>wmfile</code> for the dip rotation angle
<code>tiltindex</code>	0	attribute index for the <code>wmfile</code> for the tilt rotation angle
<code>strikevalue</code>	0.0	strike rotation angle for the entire mesh
<code>dipvalue</code>	90.0	dip rotation angle for the entire mesh
<code>tiltvalue</code>	0.0	tilt rotation angle for the entire mesh
<code>gradtol</code>	0.0	tolerance on min. vertex/dihedral angle when generating the gradient operators
<code>alpham</code>	1.0	across-face smoothness regularization multiplier (see parameter <code>rotate</code> for usage!)
<code>alphax</code>	1.0	smoothness regularization multiplier for x-direction (see parameter <code>rotate</code> for usage!)
<code>alphay</code>	1.0	smoothness regularization multiplier for y-direction (see parameter <code>rotate</code> for usage!)
<code>alphaz</code>	1.0	smoothness regularization multiplier for z-direction (see parameter <code>rotate</code> for usage!)
<code>alphan</code>	1.0	smoothness regularization multiplier for north/south direction (an alias for <code>alphax</code>)
<code>alphae</code>	1.0	smoothness regularization multiplier for east/west direction (an alias for <code>alphay</code>)
<code>alphav</code>	1.0	smoothness regularization multiplier for vertical direction (an alias for <code>alphaz</code>)
<code>measure0</code>	"ell2"	specifies the type of measure for the smallness regularization term
<code>measure1</code>	"ell2"	specifies the type of measure for the smoothness regularization term

Name	Default	Brief description
ekblomp	2.0	the p-value for the Ekblom measure or total-variation measure
ekblome	1.0E-9	the epsilon value for the Ekblom measure or or total-variation measure
smooth_mod_diff	"f"	specifies whether or not the reference model appears in the smoothness terms
smooth_wts_inside	"f"	specifies where the smoothness weights should appear in the smoothness terms
CONSTRAINTS		
lowerunit	0	lower bound for rock-unit IDs when discrete is "t" (true)
upperunit	0	upper bound for rock-unit IDs when discrete is "t" (true)
JOINT INVERSION		
alphaj	0.0	multiplier on the sum of joint measures
jointinp	""	input file for a particular joint coupling
stageinit	0	the joint inversion will start at this stage
searchr	"t"	set to false ("f") to avoid ratio search for lambda
jchitol	0.05	relative tolerance on the joint pareto misfit
OPTIMIZATION		
optiminp	""	input file for optimization engine options
doparinv	"t"	determines where parallelization occurs
TIKHONOV SEARCH		
lambdainit	0.0	initial lambda trade-off parameter value
minlambdasteps	4	minimum number of steps in lambda adjustment (e.g. bisection) search
maxlambdasteps	48	maximum number of steps in lambda adjustment (e.g. bisection) search
lambdafactmin	1.01	minimum multiplication factor when adjusting lambda
lambdafactmax	2.0	maximum multiplication factor when adjusting lambda
lambdamult	1.0	increasing this factor will lead to larger adjustments when close to the target
ratiomult	1.0	increasing this factor will lead to larger adjustments when close to the target
tikinterp	"f"	set to "t" (true) to attempt interpolation for tradeoff parameter
linearize	"f"	if true then linearization is used (affects how the Jacobian is adjusted)
lambdatol	0.01	convergence tolerance on relative lambda change when linearizing
modeltol	0.01	convergence tolerance on relative model change when linearizing
misfitstallsteps	2	number of steps across which to check for a stall in misfit values when linearizing
misfitstalltol	0.002	stall tolerance on relative misfit change when linearizing
maxlinearizesteps	4	minimum number of linearization iterations allowed
maxlinearizesteps	12	maximum number of linearization iterations allowed
maxratiosteps	48	maximum number of ratio steps allowed (for joint inversion)
OUTPUTS		
writeinter	1	set > 0 to write intermediate output files (see more below)
totitsuffix	"f"	set to "t" (true) to adjust suffix of intermediate output files to indicate iteration numbers
mtxroot	""	set to some non-empty file root to write sensitivity matrix file(s) (linear problems only)

4.4.1 revision

The **revision** parameter is used to assess whether you are re-running an inversion that was originally run on an older version of the program. If the specified revision number is less than the built-in version number of the program then a warning message may be issued and the user will be prompted to continue or quit. The idea is that you will specify the program revision number at the time that you set up the inversion. Then, if at a future date some changes have been made to the program defaults, you will be informed of the changes. Otherwise, your work may not be reproducible unless you change your older input files.

For users in my research group, you can simply run **svn update** and use the quoted revision number. For public users, your SVN revision numbers are different because you are using a different repository. Therefore you will have to refer to the [change log](#). If in doubt, you can always put a huge revision number into your files, e.g. 1000000.

4.4.2 Mesh-related parameters

The mesh can be a 2D or 3D rectilinear or unstructured (triangular or tetrahedral) mesh.

meshinp

- A model discretization specification file.
- See the documentation on [model discretization specification files](#) for more information on the file format.

4.4.3 Physical property-related parameters

`propinp`

- Each physical property has its own input file (`propinp`).
- Specify the `propinp` parameter as many times as required.
- See the documentation on [physical property files](#) for more information on the file format.

4.4.4 Data-related parameters

`datainp`

- Each dataset has its own input file (`datainp`).
- Specify the `datainp` parameter as many times as required.
- See the documentation for program `FOGO` for the data-specific parameters.

`normalized`

- If `"t"` (true, the default) then the data misfit term(s) are normalized by the number of data.
- This normalization occurs in the objective function and is most helpful in a joint inversion when trying to balance multiple geophysical data sets.

`normalizec`

- If `"t"` (true, the default) then the data misfit term(s) are normalized by the `chifact` value(s).
- If any of those values are zero, which can occur during testing or when using a global optimization, then this normalization does not occur.
- This normalization occurs in the objective function and is most helpful in a joint inversion when trying to balance multiple geophysical data sets.

4.4.5 Regularization parameters

Some regularization parameters are specified in the individual physical property definition files instead of the inversion input file. See the documentation on [physical property files](#) for more information.

Some regularization parameters specify directions: pay careful attention to the Cartesian coordinate system definitions in [Section 4.4.6](#).

`rotate`

- For rectilinear meshes, if `rotate` is `"f"` (false, the default) then there are three smoothness operators that act along each Cartesian axis direction; those operators calculate model differences across cell faces. If `rotate` is `"t"` (true) then those smoothness operators can be rotated to some arbitrary orthogonal orientations, in which case the operators no longer calculate differences across cell faces but rather look at the spatial gradients in small packages of cells (see [Lelièvre & Oldenburg, 2009](#)). In either case, parameters `alphax`, `alphay` and `alphaz` are used to weight the three different smoothness terms. Refer to the equations for the model objective function in [Section 4.5.2](#).
- For unstructured meshes, if `rotate` is `"f"` (false, the default) then there is a single smoothness operator that calculates differences across cell faces and parameter `alpham` is the associated weight. If `rotate` is `"t"` (true) then there are three smoothness operators that look at the spatial gradients in small packages of cells (see [Lelièvre & Farquharson, 2013](#)) and parameters `alphax`, `alphay` and `alphaz` are used to weight the three different smoothness terms. **Hence, with unstructured meshes, you must specify different weighting parameters depending on the value specified for parameter `rotate`.** Refer to the equations for the model objective function in [Section 4.5.2](#).
- See the notes below for parameter `gradtol` if using rotated smoothness axes on unstructured meshes.
- [Section 4.4.6](#) provides further instructions on how to perform an inversion with the smoothness axes rotated.
- If you wish to use the total variation measure for the smoothness term, you must set `rotate` to `"t"` (true) if you are using an unstructured mesh. However, the `alphax/y/z` parameters are ignored (refer to the notes below for parameter `measure1`).

`maskfile`, `maskindex`

- The **maskfile** specifies the mask for the regularization across the mesh. You will want to mask the regularization if, for example, you have a rectilinear mesh with cells above the topography surface, for which program **toposplit** can be used to generate the required **maskfile**. In that situation, if you do not mask the regularization then smoothness values will be calculated across the earth-air interface.
- The **maskfile** should be a **.ele** format file or a simple text file with one or more columns of values.
- Attribute index **maskindex** of file **maskfile** is used as the mask. Cells with values ≤ 0 are not included in the regularization. For example, no smoothness regularization occurs across a mesh face if either adjacent cell has value ≤ 0 .

wmfile, wmindex

- The **wmfile** specifies the smoothness weights. This should be a **.ele** format file or a simple text file with one or more columns of values.
- If **rotate** is "f" (false, the default) then, for either rectilinear or unstructured meshes, the number of required weights is equal to the number of internal faces in the mesh (faces other than those lying on the outer boundary of the mesh). The **wmfile** should list the smoothness weights for the easting, northing and then vertical directions:

```

wx1
wx2
...
wxL
wy1
wy2
...
wyM
wz1
wz2
...
wzN

```

where L , M and N denote the number of internal mesh faces in each Cartesian direction in the mesh. For an unstructured mesh, the ordering of the weights should be as specified in the **.faces** file generated by TetGen. For a rectilinear mesh, the ordering of the weights should be as in the UBC-GIF format for smoothness weighting files. Program **make_face_weights** writes a file in the correct format, with the correct ordering, and may be of help.

- For legacy considerations, if **rotate** is "f" (false) and you are using a rectilinear mesh then the **wmfile** may contain smallness weights w_s before the smoothness weights:

```

ws1
ws2
...
wsK
wx1
...
wzN

```

where K denotes the number of mesh cells. That is the standard UBC-GIF format for smoothness weighting files. However, any smallness weights present in the **wmfile** are ignored by program **VIDI** when generating the smoothness operators. Instead, the smallness weights should be specified in the physical property input files using the **wsfile** parameter. This behaviour allows you to specify the same file for both the **wsfile** and the **wmfile**, in which case all the weights in the file will be used by **VIDI**.

- If **rotate** is "t" (true) then, for either rectilinear or unstructured meshes, the number of required weights is equal to the number of cells in the mesh. In this case, the **wmfile** can be built as you would any other model on your mesh. For an unstructured mesh, parameter **wmindex** specifies which attribute in the **.ele** file, or which data column in the column-based data file, to use for the weights. For a rectilinear mesh, if **rotate** is "t" (true) then the file should be in the standard UBC-GIF model format, with one value for each cell in the mesh, and each value on a new line.
- If **wmfile** is specified as "null" then the smoothness weights are set to 1.0 for the entire mesh.

wxindex, wyindex, wzindex, strikeindex, dipindex, tiltindex, strikevalue, dipvalue, tiltvalue

- These parameters are only used if **rotate** is "t" (true).
- Parameters **wxindex**, **wyindex** and **wzindex** specify the columns in the **wmfile** is used for the cell-centred smoothness weights for the three Cartesian directions.

- The `strikeindex`, `dipindex` and `tiltindex` columns in the `wmfile` are used to specify the rotation of the smoothness axes.
- If `wmfile` is specified as "null" then `strikevalue`, `dipvalue` and `tiltvalue` specify the rotations for the entire mesh.
- The rotation parameters are only used if `rotate` is set to "t" (true).

`gradtol`

- When using rotated smoothness axes on unstructured meshes, if the minimum vertex (2D) or dihedral (3D) angle defined by the cell centroids in a particular package of neighbouring cells is less than this tolerance then an approximate gradient calculation is used for that package of cells. The alternative would be to remove that package of cells from the gradient calculation completely. However, if many such packages of cells were removed from the calculation and `alphas = 0` then some cells may be completely absent from the regularization, leading to an ill-posed problem. Hence, the code uses the approximation option as necessary to avoid ill-posed problems.
- If `gradtol` is set too low, e.g. using the default value of 0, poor quality meshes may lead to division by small values in the gradient calculations, possibly leading to artifacts in recovered models. If `gradtol` is set too high, the approximate gradient calculation may lead to inversion results that do not follow the intended character, e.g. orientations in incorrect directions. Hence, when using rotated smoothness axes on unstructured meshes, users should attempt to use meshes with as good quality as is feasible (higher quality often means more cells) such that the value of this parameter can be set low.
- You will need to experiment with the `gradtol` parameter. If the mesh is good quality then it can be set fairly low. Maybe even 0. Try an inversion with a low value, e.g. 0, and if there are clearly spurious artifacts then try it higher. If you go too high then you won't be using true gradient operators and the result may not recover the orientations that you prescribed. It is a trade-off.

`measure0`, `measure1`

- set to "ell2" for the standard, sum-of-squares, L2 norm
- set to "ekblom" for the Ekblom measure; parameters `ekblomp` and `ekblome` then come into effect
- set to "totvar" for the total variation measure; parameters `ekblomp` and `ekblome` then come into effect; can only be used for the `measure1` and then only if directional smoothness operators are being used; parameters `alphax/y/z` are ignored when the total variation measure is used.

4.4.6 Instructions for rotating the smoothness directions

If you want to emphasize pipe- or rod-like features, you will want to rotate the smoothness axes so that they measure the smoothness along the major and minor axes of the desired features. Set parameter `rotate` to "t" (true). If using an unstructured mesh, pay attention to parameter `gradtol`.

If you want to specify a homogeneous orientation across the entire inversion mesh then set parameter `wmfile` to "null" and use parameters `strikevalue`, `dipvalue` and `tiltvalue` to specify your desired orientation. If you want to specify heterogeneous orientations then construct an appropriate file for use with parameter `wmfile` and use parameters `strikeindex`, `dipindex` and `tiltindex` to specify your desired orientations. When you rotate the smoothness axes you end up with gradient operators that calculate quantities at cell centres instead of at the faces between cells. This means that your `wmfile` specifies orientations and weights for every cell in the model. You can therefore build that weighting information like you would any other model on your mesh.

If you want to specify homogeneous smoothness weights across the entire inversion mesh then use parameters `alphax`, `alphay` and `alphaz` to specify the weights for the three rotated smoothness directions. If you want to specify heterogeneous smoothness weights then use parameters `wxindex`, `wyindex` and `wzindex`.

For 2D, the only angle used is the dip. The coordinate system definition used inside the code has $+x$ right and $+z$ down. A dip of zero is along the x -axis, to the right. A positive dip is a clockwise rotation from $+x$ towards $+z$. Hence, a dip of 90 degrees is downwards.

For 3D, the strike is the clockwise angle from north towards east: a zero strike is north, a 90 degree strike is east. If you are standing and looking in the strike direction then the dip is an angle to your right below horizontal: a dip of 0 is horizontal, a dip of 90 degrees is straight down. The coordinate system definition used inside the code has $+x$ north, $+y$ east and $+z$ down. The rotations define a new set of rotated axes with the rotated x -axis along the strike direction and the z -axis in the down-dip direction. **The parameters `alphax`, `alphay` and `alphaz` are weights along the rotated smoothness axes. If the axes are not rotated, then `alphax` controls the smoothing in the north-south direction, `alphay` in the east-west direction, and `alphaz` vertically.**

A positive tilt is a rotation around the new y' -axis that moves x' towards z' . If the dip is 90 degrees then a small positive tilt rotation moves the x' axis beneath the ground surface. i.e. it is a plunge below horizontal.

Refer to [Section 4.5.2](#) for a description of the rotated model objective function. Here I define w_x to be a particular cell's weight for the x-smoothness direction, with similar definitions for w_y and w_z . If you want to emphasize pipe-like features, you rotate the smoothness axes so that the z-smoothness axis is along the major axis of the pipe and you set

$$\begin{aligned} w_x &= w_y \\ w_z &\gg w_x \end{aligned}$$

which specifies higher smoothness along the pipe major axis, the result being an elongation of features along the major axis.

If you want to emphasize plate-like (or disk-like) features, you rotate the smoothness axes so that the z-smoothness axis is perpendicular to the plate and you set

$$\begin{aligned} w_x &= w_y \\ w_z &\ll w_x \end{aligned}$$

which specifies lower smoothness perpendicular to the plate, the result being elongation parallel to the plate and sharp jumps perpendicular to it.

You may have to experiment with the ratio w_z/w_x to obtain appropriate results. I generally start with a ratio of 10 for pipe-like features and 0.1 for plate-like features.

If you want to emphasize features that do not contain symmetry around a major or minor axes then you will generally have

$$w_x \neq w_y \neq w_z.$$

4.4.7 Constraint parameters

Some constraint parameters are specified in the individual physical property definition files instead of the inversion input file. See the documentation on [physical property files](#) for more information.

4.4.8 Joint inversion options

alpha_j

- The multiplier on the sum of joint measures is typically only set to 1.0 or 0.0.
- Use 1.0 to include the coupling specified in the **jointinp** files.
- Use 0.0 to exclude any coupling. This can be helpful before a joint inversion to assess relative weights for each model norm term before running a coupled joint inversion with **alpha_j** set to 1.0: see parameter **alphab** in [Section 2.4](#) and the model objective function [Section 4.5.2](#).

jointinp

- Each joint coupling measure you wish to include has its own input file (**jointinp**).
- Specify the **jointinp** parameter as many times as you have different couplings (pairs of coupled models).
- See the documentation on [joint coupling measure files](#) for more information on the file format.

stageinit

- A joint inversion proceeds in several stages, heating the coupling multiplier values in the objective function (see ρ_k in [Section 4.5.3](#)) from 0.0 at stage 0 up to the final value **rho** at stage **nsteps** (both specified in the **jointinp** files). You may wish to skip stage 0, or you may need to use the **stageinit** parameter to restart a joint inversion.

A joint inversion proceeds in several stages, heating the **rho** values in the objective function (see [Section 4.5.3](#)) from 0.0 at stage 0 up to the specified final value the **jointinp** file. This heating occurs over **nsteps** stages (also specified in the **jointinp** file).

•

searchr

- If set to "f" (false) to avoid ratio search for lambda then the final misfits may not be as close to their targets as requested.

The suggested standard procedure for running joint inversions is:

1. Run the independent inversions.
2. Run a joint inversion with no coupling (set `alphaj` to 0.0) and compare to the results from the previous step. Pay particular attention to the model objective function values `norm1` and `norm2` specified in the output `.log` files. If the results are significantly different then the model objective function terms may need to be differentially weighted in the joint inversion: see parameter `alphab` in [Section 2.4](#) and the model objective function [Section 4.5.2](#).
3. Re-run the joint inversion with no coupling, as necessary to assess appropriate relative weights for each model norm term.
4. Run a joint inversion with coupling, using the result from the previous step to initialize. This means setting `stageinit` to 1, setting appropriate values for `lambdainit` in the data input files (see [Section 3.3](#)) and specifying the initial models in the [physical property input files](#).

4.4.9 Optimization options

`optiminp`

- See the documentation on [optimization engine specification files](#) for more information on the file format.

`doparinv`

- If `doparinv` is "t" (true, the default) then the GA or PMOGA algorithm (if used) calculates the objective function for each candidate solution in parallel. This means that many forward modelling solutions occur in parallel. If `doparinv` is "f" (false) then the parallelization will only occur inside the forward modelling solution (if the forward modelling of a given type data is also parallelized), and only one forward modelling solution occurs at any one time.

`discrete`

- This parameter is specified in the `optiminp` file.
- If "f" (false) then the physical properties in each cell are continuous variables (between bounds). This is always the approach used when `engine` is set to "local".
- If "t" (true) then the model values are integers representing rock-unit IDs that are linked to physical properties via the `regionunitsfile` specified in the `meshinp` file (see [Sections 2.7](#) and [3.3.5](#) for more information). This option is only used when `engine` specifies a global optimization, i.e. set to "pso", or "ga". Bounds, specified using `lowerunit` and `upperunit`, should be used to limit the model values to the allowed range of rock-unit IDs and all rock-unit IDs must represent the set of all integers between and including those bounds.

4.4.10 Tikhonov search options

These parameters are only used if performing a single-objective optimization, i.e. `multiobj` is "f" (false) in the `optiminp` file. Currently, they are not used unless `engine` is `local`.

`lambdainit`

- When there are multiple datasets, all lambda values are set to `lambdainit` by default.
- However, any lambda values specified in the data input files take precedence: see parameter `lambdainit` in [Section 3.3](#).

`tikinterp`

- If "t" (true) then the inversion will attempt an interpolation for the tradeoff parameter once straddling the target misfit value. This will only occur once, after which the usual heating search will take over.
- If "t" (true) then the other Tikhonov search options will be ignored until after the single interpolation has been performed.
- Currently, only a linear interpolation in log-log space is supported.
- If there is more than one dataset then this input parameter is ignored.

4.4.11 Output options

`writeinter`

- Set ≥ 1 to write intermediate output files at each lambda value; ≥ 2 for each Gauss-Newton or GPRN iteration.
- For very fast inversions, your machine's runtime library I/O buffer may not flush between the different intermediate inversion iterations. This means that if an output file is open by another process, e.g. in a text editor, the file may not be written correctly each iteration. However, measures have been put in place such that the final output file will be written

correctly: the approach I have taken is to pause execution by one second, allowing the I/O buffer to catch up - this should work on all but the most archaic computers (think vacuum tubes) but if you want to be safe, don't have the output files open while running an inversion.

totitsuffix

- If `writeinter` ≥ 1 then the names of the files are `*_totit####.*` where `####` is an integer value indicating the iteration number in the lambda (trade-off parameter) search.
- If `writeinter` ≥ 2 then the names of the files are `*_totit####_gnit####.*` where the additional suffix indicates the Gauss-Newton or GPRN iteration.
- Use the log file to cross-reference with stage or lambda iteration numbers.

mtxroot

- The names of the output sensitivity matrix file(s) are `mtxroot*.mtx`.
- These are overwritten if they exist already.
- This is helpful if you plan to run multiple inversions with the same sensitivity matrix. You could use program [FOGO](#) to calculate the sensitivity matrix. However, by using this parameter you can have that happen automatically on the first inversion you run, then alter the [physical property input files](#) to specify that matrix file for later inversions.

4.5 The objective function

The objective function for an inversion with a single data type and a single causative physical property model is

$$\Phi = \lambda\gamma\Phi_d + \Phi_m \quad (4.1)$$

The objective function for an inversion with a single data type and multiple physical property models (for example, inversion of total field magnetic data for three components of the vector magnetization) is

$$\Phi = \lambda\gamma\Phi_d + \Phi_{m1} + \Phi_{m2} + \dots \quad (4.2)$$

The objective function for a joint inversion with two data types and a single physical property model (for example, inversion of vertical gravity and gravity gradiometry data for density) is

$$\Phi = \lambda_1\gamma_1\Phi_{d1} + \lambda_2\gamma_2\Phi_{d2} + \Phi_m \quad (4.3)$$

In this situation, the inversion attempts to fit both data sets to their assigned targets by altering the two λ (lambda) trade-off parameters.

The objective function for a joint inversion with two data types and two physical property models (for example, inversion of vertical gravity data for density, and total field magnetic data for magnetic susceptibility) is

$$\Phi = \lambda_1\gamma_1\Phi_{d1} + \lambda_2\gamma_2\Phi_{d2} + \Phi_{m1} + \Phi_{m2} + \Psi \quad (4.4)$$

In this situation, the inversion attempts to fit both data sets to their assigned targets by altering the two λ (lambda) trade-off parameters. Function Ψ is the joint coupling term.

The objective function for a joint inversion with more than two data types is

$$\Phi = \lambda(\gamma_1\Phi_{d1} + \gamma_2\Phi_{d2}) + \Phi_{m1} + \dots \quad (4.5)$$

In this situation, the inversion does not attempt to fit both data sets to their assigned targets.

The γ (gamma) values are user-defined constants specified in the data input files: see section 3.3. Note the λ (lambda) trade-off parameters multiply the data misfit terms, Φ_{d1} and Φ_{d2} , not the model objective terms as is sometimes seen in literature. The initial values of the trade-off parameters can be specified in the inversion and data input files: see sections 3.3 and 4.4.

4.5.1 The data misfit terms

Each misfit term is of the form

$$\Phi_d = \frac{1}{NC} \mathbf{e}^T \rho \left(\mathbf{W}_d (\mathbf{d}_{pred} - \mathbf{d}_{obs}) \right)$$

where N is the number of data, C is the target **chifact** value, \mathbf{e} is a vector of ones, \mathbf{W}_d is a diagonal matrix holding the inverses of the assigned uncertainties, \mathbf{d}_{pred} is the predicted data (the forward modelled response of some candidate model), \mathbf{d}_{obs} is the observed data, and the function ρ defines a particular measure: $\rho(\dots)$ is a vector. See below for more information on the possible options for the function ρ . When an ℓ_2 -norm is used, the result is the classic χ -squared data misfit:

$$\Phi_d = \frac{1}{N} \sum_{i=1}^N \frac{(d_{i,pred} - d_{i,obs})^2}{\sigma_i^2}$$

where σ_i is the uncertainty assigned to the i^{th} data measurement. The normalization by the number of data and chifact value can be removed using the **normalized** and **normalizec** parameters in the inversion input file: see section 4.4.

4.5.2 The model objective function

The model objective function is

$$\Phi_m = \Phi_s + \Phi_f + \Phi_x + \Phi_y + \Phi_z + \Phi_t$$

Each term is of the form

$$\Phi_- = \alpha \mathbf{v}^T \mathbf{W} \rho(\boldsymbol{\omega})$$

where \mathbf{v} holds integration volumes, \mathbf{W} is a diagonal matrix of weights, $\boldsymbol{\omega}$ is some function of the model \mathbf{m} , and the function ρ defines a particular measure: $\rho(\boldsymbol{\omega})$ is a vector. The options for the $\rho(\boldsymbol{\omega})$ function are the ℓ_2 -norm

$$\rho(\boldsymbol{\omega}) = \boldsymbol{\omega}^2$$

and the Ekblohm measure

$$\rho(\boldsymbol{\omega}) = (\boldsymbol{\omega}^2 + \epsilon^2)^{p/2}$$

which reduces to the ℓ_2 -norm for $p = 2$ and $\epsilon = 0$.

The smallness term is

$$\begin{aligned} \Phi_s &= \alpha_b \alpha_s \mathbf{v}_c^T \mathbf{W}_s \rho(\boldsymbol{\omega}_s) \\ \boldsymbol{\omega}_s &= \mathbf{m} - \mathbf{m}_{ref} \end{aligned}$$

where

- \mathbf{m}_{ref} is a reference model (set to zero if absent),
- \mathbf{v}_c holds cell volumes,
- \mathbf{W}_s is a diagonal matrix containing any user-defined smallness weights combined with any depth, distance or sensitivity-based weights.

The remaining terms are all smoothness terms. **Only one may be used!** The first is currently only used for unstructured grids:

$$\begin{aligned} \Phi_f &= \alpha_b \alpha_m \mathbf{v}_f^T \mathbf{W}_f \rho(\boldsymbol{\omega}_f) \\ \boldsymbol{\omega}_f &= \mathbf{D}_f \mathbf{m} \end{aligned}$$

where

- \mathbf{D}_f is a general difference matrix across cell faces,
- \mathbf{v}_f holds the appropriate integration volumes,
- \mathbf{W}_f is a diagonal matrix containing any user-defined smoothness weights combined with any depth, distance or sensitivity-based weights.

The next three smoothness terms are always used for rectilinear grids and are only used for unstructured grids when rotating the smoothness directions:

$$\begin{aligned} \Phi_x &= \alpha_b \alpha_m \alpha_x \mathbf{v}_x^T \mathbf{W}_x \rho(\boldsymbol{\omega}_x) \\ \boldsymbol{\omega}_x &= \mathbf{G}_x \mathbf{m} \end{aligned}$$

where

- \mathbf{G}_x is a gradient operator for the x direction,
- similar definitions exist for the y and z terms.

Note the additional α_m multiplier! When the gradient directions are rotated, the volume vectors \mathbf{v}_x , \mathbf{v}_y and \mathbf{v}_z are identical and equal to \mathbf{v}_c .

The final smoothness term is a total variation term:

$$\begin{aligned}\Phi_t &= \alpha_b \alpha_m \mathbf{v}_c^T \mathbf{W}_c (\omega_t^2 + \epsilon^2)^{p/2} \\ \omega_t &= \mathbf{Q}_x \left((\mathbf{G}_x \mathbf{m})^2 \right) + \mathbf{Q}_y \left((\mathbf{G}_y \mathbf{m})^2 \right) + \mathbf{Q}_z \left((\mathbf{G}_z \mathbf{m})^2 \right)\end{aligned}$$

- \mathbf{W}_c is a diagonal matrix containing any user-defined smoothness weights combined with any depth, distance or sensitivity-based weights,
- the ρ function is always the Eklblom measure for this term,
- the exponential powers are applied element-by-element,
- \mathbf{Q}_x interpolates the x -direction squared gradient values at cell centres.

For rectilinear grids, the gradient operators calculate gradients on cell faces and interpolation is required to calculate values at cell centres. For unstructured grids, the gradient operators calculate gradients at cell centres and the interpolation operators are identity matrices.

For the smoothness terms, if `smooth_mod_diff` is "f" (false, the default) then the operators are applied to the model, as indicated in the equations above. If `smooth_mod_diff` is set to "t" (true) then the operators are applied to $\mathbf{m} - \mathbf{m}_{ref}$ (replace \mathbf{m} with $\mathbf{m} - \mathbf{m}_{ref}$ in the equations above). The appropriate method will depend on the characteristics of the reference model: please see [Williams \(2008\)](#) for guidance.

For the smoothness terms, if `smooth_wts_inside` is "f" (false, the default) then the smoothness weights are applied outside the derivatives (to the left of the gradient operators), as indicated in the equations above. If `smooth_wts_inside` is set to "t" (true) then the operators are applied inside the derivatives (to the immediate left of the model \mathbf{m} , and therefore to the right of the gradient operators, in the equations above). This latter use is consistent with the UBC-GIF inversion codes **but is not recommended**, especially if non- ℓ_2 norms are used.

4.5.3 The joint coupling term

The joint coupling term is a weighted sum of any couplings specified in the input files:

$$\Psi = \alpha_j \sum_k \rho_k \psi_k \quad (4.6)$$

where the sum is over all coupling measures specified, ψ_k .

Scalar quantities used in the equations above correspond to inversion input parameters as indicated below:

symbol	parameter(s)
α_b	alphab
α_s	alphas
α_m	alpham
α_x	alphax
α_y	alphay
α_z	alphaz
α_j	alphaj
\mathbf{W}_s	wsfile, wsindex
\mathbf{W}_f	wmfile, wmindex
\mathbf{W}_x	wmfile, wxindex
\mathbf{W}_y	wmfile, wyindex
\mathbf{W}_z	wmfile, wzindex
ϵ	ekblome
p	ekblomp

Chapter 5

Surface geometry inversion: DYN0 (formerly WINV)

5.1 Program summary

DYN0 (**DY**namic **NO**de inversion) is a surface geometry inversion program that determines the positions of surfaces in a geological model that represent the contacts between rock units. The discretization is a wireframe surface of tessellated triangles. Single or multiple data-types can be inverted.

The surface(s) can be parameterized with Cartesian coordinate nodes, polar/spherical coordinate nodes, or using circular/spherical harmonics. In Cartesian mode, the (x,y,z) positions of the surface nodes are solved for directly. In spherical mode, the surface is solved for in spherical coordinates (angles fixed, radius determined). In harmonics mode, the real and imaginary values of the spherical (3D) or circular (2D) harmonics are solved for and the Cartesian positions of the surface nodes are calculated indirectly. In spherical and harmonics mode, the centroid(s) remain fixed. The harmonics mode only works for a single closed 2D outline.

A spline can be fit through knot points in 2D Cartesian or polar coordinates. 2D or 3D Cartesian or polar/spherical control points on a coarse wireframe can be refined via surface subdivision.

Gravity (vertical and gradiometry), magnetics and straight-ray seismic data are currently supported. Curved-ray seismics and DC/IP are not supported yet but are planned.

This program will run in parallel. Remember to set the OMP_NUM_THREADS environment variable. On a Mac you use the following command in the terminal window before running this program:

```
> export OMP_NUM_THREADS=value
```

5.2 Running the program

5.2.1 Command line usages

```
dyno inputfile
dyno inputfile outputroot
dyno inputfile outputroot stage
dyno inputfile outputroot stage seed
```

5.2.2 Command line parameters

inputfile

- The name of the input file (the format is described below).
- See the note at the start of [Chapter 2](#) regarding relative file paths.

outputroot

- The names of the output files have this root (the output files are described below).
- If **outputroot** is absent then it is set to the root of the input file, that is, the input file name with extension and any path information stripped off: for example, if **inputfile** is set to **mydirectory/myfile.ext** then the output files will be named **myfile*** and placed in the current working directory.

stage

- Set to 1 to perform the minimization, e.g. using the genetic algorithm solver.

- Set to 2 to perform the burn-in of the Metropolis-Hastings MCMC algorithm (before statistics calculation).
- Set to 3 to perform the statistics calculation (using the Metropolis-Hastings MCMC algorithm).

The general procedure is to first run stage 1 to find a good starting model for the later stages. Stage 2 is an MCMC sampling used as the burn-in. No statistics are calculated in stage 2. The assumption is that after the burn-in stage 2, the Markov Chain has reached equilibrium such that the statistics can then be calculated, which happens in stage 3.

seed

- Set to some integer value to use as the seed for random number generation (important for global optimizers).

5.3 Outputs

5.3.1 Terminal output

The output printed to the terminal/command window remains rudimentary and can be altered as users require. Please contact the developers with any suggestions.

Currently, the terminal outputs for a single-objective global search with GA (Genetic Algorithm optimization engine) are:

```
Generation ParamSpread ObjBestEver ObjBestPop VioBestEver VioBestPop NFea NInf
```

where

- **Generation** is the current GA iteration number (generation number)
- **ParamSpread** is the parameter spread
- **ObjBestEver** is the best objective function value (see more below) every obtained by any solution at any iteration (any individual in the GA population at any generation)
- **ObjBestPop** is the best objective function value for the solutions at the current iteration (the individuals in the GA population at the current generation)
- **VioBestEver** is the best constrain violation measure every obtained by any solution at any iteration (any individual in the GA population at any generation)
- **VioBestPop** is the best constrain violation measure for the solutions at the current iteration (the individuals in the GA population at the current generation)
- **NFea** is the number of feasible solutions (which honour the constraints) in the current GA population.
- **NInf** is the number of infeasible solutions (which do not honour the constraints) in the current GA population.

5.3.2 Output files

The output files are named **outroot*** with various suffixes and extensions, as listed below. The output file directory is the directory from which the program is being run, unless specified differently by adding path information to **outroot**.

Output files named ***.best.*** are for the model that fits the data best. When running an MCMC search, additional output files named ***.mean.*** will be written that hold the mean model; the ***.mean.vtu** file will contain node attributes indicating the standard deviation of the node locations. The predicted data files are also written.

5.4 Input file format

Each line of the input file should be of the format

```
name [index] value
```

where **name** is the name of some modelling parameter and **value** is the value for that parameter. Some parameters require that an index is specified to link the parameter to a specific dataset and associated physical property (currently, it is assumed that each dataset is associated with a different physical property). The possible parameters and default values are listed below. Those that require the index specifier are indicated with "[]".

Lines in the input file beginning with the # or ! character are ignored and can be used as comments for your own reference.

When specifying path/file names in the input file, place double quotes around them (e.g. "../../../../mesh.txt"). Single quotes should also work but this is machine dependent.

Parameters `revision`, `propinp` and `datainp` are explained further in [Section 4.4](#).

Name	Default	Brief description
<code>revision</code>	0	a code revision number (see Section 4.4 for more information)
<code>meshinp</code>	"null"	a model parameterization specification file
<code>propinp</code>	""	input file for a particular physical property (see Section 4.4 for more information)
<code>datainp</code>	""	input file for a particular geophysical data set (see Section 4.4 for more information)
<code>optiminp</code>	""	input file for optimization engine options
<code>doparinv</code>	"t"	determines where parallelization occurs
<code>parMPI</code>	"f"	when parallelization occurs, determine if use MPI
<code>surffile</code>	""	file describing the surface parameterization and any related inversion parameters
<code>minangle</code>	200.0	minimum facet angle
<code>colldet</code>	"f"	set to "t" (true) to perform collision detection
<code>collall</code>	"t"	if "t" (true, the default) then all facet collisions are counted; otherwise stops counting after the first
<code>collfile</code>	"null"	a <code>.ele</code> file specifying possible groups of facets than could collide (intersect)

`meshinp`

- See the documentation on [model discretization specification files](#) for more information on the file format.
- The `meshfile` and `modelfile` should be `.node` and `.ele` format files respectively defining the initial wireframe surface model.
- The `nodeunitsfile` should include the following attributes:
 - `Dynamic` defines whether or not (values 1 or 0) the node can move.
 - `XLower`, `XUpper`, `YLower`, etc. define the upper and lower Cartesian node coordinate bounds. These can be absolute or relative to initial positions (see parameters `relative` and `relativez` - default "t" (true)). These bounds are only used in `cart`, `casp` and `casd` mode. They should be specified in the input coordinate system (+z up or down, whichever you are using).
 - `Centroid` specifies centroid indices used to move the nodes in `sphr` and `spsp` mode.
 - `RadiusLower`, `RadiusUpper` and `RadiusScaling` define the bounds and scaling for the radius used in `sphr`, `spsp` and `harm` mode. These can be absolute or relative to initial positions (see parameter `relative` - default "f" (false)).
- The `facetunitsfile` should include the attributes `Region1` and `Region2` if you are including gravity or magnetics data. See program [FOGO](#) for more information.
- The `regionunitsfile` should include the physical property attributes (e.g. `Density`, `MagSus`, `Slowness`, `Conductivity`) required for the data types included.

`optiminp`

- See the documentation on [optimization engine specification files](#) for more information on the file format.
- You must use a global optimization engine, e.g. PSO or GA.

`doparinv`

- If `doparinv` is "t" (true, the default) then the GA or PMOGA algorithm (if used) calculates the objective function for each candidate solution in parallel. This means that many forward modelling solutions occur in parallel. If `doparinv` is "f" (false) then the parallelization will only occur inside the forward modelling solution (if the forward modelling of a given type data is also parallelized), and only one forward modelling solution occurs at any one time.

`parMPI`

- When `doparinv` is "t" (true) the GA or PMOGA parallelization is achieved by OpenMP (Open Multi-processing) by default. If `parMPI` is also "t" (true) then this task is achieved by MPI (Message Passing Interface) rather than by OpenMP. `parMPI` is "f" (false, the default).

`surffile`

- As for the input file, each line of the `surffile` should be of the format "name value".

Name	Default	Brief description
OPTIONS FOR ALL MODES		

Name	Default	Brief description
surftype	""	defines the type of wireframe surface parameterization
ndim	0	the number of dimensions
ztopo	∞	upper bound on z coordinates for subdivided models
coordinatescaling	1.0	scaling applied to the Cartesian node/centroid coordinates
OPTIONS FOR MODES CART, CASP & CASD		
strike	0.0	rotation angle applied to obtain the model for forward modelling
dip	90.0	rotation angle applied to obtain the model for forward modelling
tilt	0.0	rotation angle applied to obtain the model for forward modelling
relative	t	determines whether coordinate bounds are relative to initial node positions
relativez	t	determines whether z-coordinate bounds are relative to initial node z-position
OPTIONS FOR MODES SPHR, SPSP & HARM		
centroidsfile	""	regions format file containing the centroids for each surface
centroidunitsfile	""	rock-units format file that specifies radius bounds and scaling
relative	f	determines whether radius bounds are relative to the initial node positions
OPTIONS FOR MODES CASP & SPSP		
npk	4	the number of nodes per knot
OPTIONS FOR MODE CASD		
nsub	4	the number of subdivisions to perform
stat	t	whether or not the control nodes remain stationary during subdivision smoothing
OPTIONS FOR MODE HARM		
radiusinit	1.0	the initial value of the radius (the initial model is a circle/sphere)
amplitudeboundslower	0.0	lower bound for the 1st and higher harmonic coefficients
amplitudeboundsupper	1.0E6	upper bound for the 1st and higher harmonic coefficients
amplitudescaling	1.0	scaling applied to the amplitudes
maxcoeff	0	the maximum harmonic coefficient to keep

5.4.1 Parameters for all modes

surftype

The possible options are:

- **cart** to parameterize the 2D or 3D surface(s) in Cartesian coordinates. The parameters are x, y, z for each vertex.
- **sphr** to parameterize the 2D or 3D surface(s) in spherical coordinates. Nodes move towards or away from their centroid control points. The parameters are r (radius away from centroids) for each vertex. The angles are determined from the initial model and kept constant during the inversion.
- **harm** to parameterize the 2D or 3D surface(s) in spherical coordinates but stored via harmonic coefficients. The parameters are the centroid(s) and harmonic coefficients of radius versus angle. This mode should generally be avoided, except for research purposes. It is a method taken from medial imaging, where the bodies of interest can be easily fit by a spherical harmonics representation with some small number of harmonics. For geological models, however, a large number of harmonic coefficients is required to accurately describe the model shapes, and this method is then not particularly useful.
- **casp** to parameterize the 2D surface(s) with knots in Cartesian coordinates and additional interpolated nodes around a spline outline. The parameters are as for the **cart** mode.
- **spsp** to parameterize the 2D surface(s) with knots in spherical coordinates and additional interpolated nodes around a spline outline. The parameters are as for the **sphr** mode.
- **casd** to parameterize the 2D or 3D surface(s) with the control surface in Cartesian coordinates which gets refined via surface subdivision. The parameters are as for the **cart** mode.
- **spsd** to parameterize the 2D or 3D surface(s) with the control surface in spherical coordinates which gets refined via surface subdivision. The parameters are as for the **sphr** mode.

ztopo

- this parameter is only used when subdividing a model
- it can be used to ensure that no nodes in a subdivided model are above the topography surface
- this parameter does NOT affect non-subdivided models (or the coarse control surface in a subdivision scenario): you should always use appropriate bounds if not subdividing!

- this parameter acts through projection rather than the constraints: any active model nodes above this elevation are projected onto this elevation

coordinatescaling

- scaling applied to the Cartesian node/centroid coordinates when taking a random step in the MCMC search
- In MCMC mode the Metropolis–Hastings algorithm is used. In that algorithm, random perturbations to the model are considered. The resulting perturbed candidate models are always accepted if they lower the objective function, and are accepted with non-zero probability if they raise the objective function. If the model perturbations are too large (changes to the model parameters are too large) then it takes too long to converge because the rejection rate is too large: the model perturbations may not be accepted for a long time, during which the chain is stationary and you aren't doing any sensible sampling of the parameter space. If you make too small changes to the model parameters then it also takes too long to converge because it takes a long time to adequately sample all regions of the parameter space. There is a fine balance here. There is some automatic scaling that happens in MCMC but if you can get a handle on the appropriate value of the “coordinatescaling” option then this can help.

5.4.2 Parameters for cart mode

Also see the following subsection for parameters shared with other modes.

writepar

- Set to **"t"** (true) to write files “par2tot.txt” and “par2dim.txt” containing information on how to convert parameter indices to node and dimension indices.
- Set to **"f"** (false, the default) to not write those files.

5.4.3 Parameters for cart, casp and casd modes

strike, dip, tilt

- these rotation angles are applied to the model before calculating the data response
- these are helpful if you are trying to control the movement of the nodes in an object that does not have major axes aligned with the Cartesian axes

relative, relativez

- Parameter **relative** is applied to all coordinates.
- Parameter **relativez** is only applied to the z-coordinate.
- Parameter **relativez** overrides parameter **relative**. Hence, if you want all coordinates absolute then both of these parameters must be specified as **"f"** (false) in the input file.

5.4.4 Parameters for sphr, spsp and harm modes

centroidsfile

- each node can be moved towards or away from a different centroid
- for **harm** mode, only a single centroid is allowed

centroidunitsfile

- this rock-units format file must contain the attributes **RadiusLower**, **RadiusUpper** and **RadiusScaling**
- the scaling is applied to the radius value(s) when taking a random step in the MCMC search

5.4.5 Parameters for casp and spsp modes

npk

- for example, npk=4 results in 3 interpolated nodes between pairs of knots (the nodes in the input model)

5.4.6 Parameters for sphr, spsp and spsd modes

The idea for these modes is that several nodes may share a single centroid that controls their movement, so that centroid information (e.g. radius bounds) needs to be associated with the centroids, not the nodes. Each centroid is defined in the `centroidsfile` (a `.node` format file). Then, for each centroid there may be identical radius bounds, so there is an additional link required from the `centroidsfile` to a `centroidunitsfile`. The `nodesfile` will have an attribute that links each node to a centroid defined in the `centroidsfile`. The `centroidsfile` will have an attribute that links each centroid to a “unit” in the `centroidunitsfile`. The `centroidunitsfile` defines the radius bounds.

5.4.7 Parameters for casd mode

`stat`

- if “t” (true, the default) then a Dyn-Levin-Gregory interpolation is used and the subdivided interpolated surface will pass through the control nodes (the nodes in the input model)
- if “f” (false) then a Cubic B-Spline interpolation is used and the subdivided interpolated surface does not pass through the control nodes

5.4.8 Parameters for harm mode

`amplitudescaling`

- scaling applied to the amplitudes when taking a random step in the MCMC search

`maxcoeff`

- the 0th harmonic coefficient is treated as the “radius” so is controlled by parameter `radiusinit` and the bounds and scaling on the radius (see parameter `centroidunitsfile`), although the radius scaling is current hardwired to 1.0

5.4.9 Defining constraints

Bound constraints are specified using various parameters mentioned above, e.g. `XLower`, `XUpper`. Additional constraints can be included using either built-in options or by writing your own subroutines.

Built-in constraints

Angle constraints: The second built-in constraint available is to avoid small angles between facets. 2D models with any facets joining at a node and making an angle less than or equal to `minangle`, in degrees, are considered infeasible. This constraint is only applied if `minangle` is on (0,180), meaning greater than zero and less than 180. This constraint is currently only supported for 2D problems.

Intersection constraints: The third built-in constraint available is associated with collision detection. You can turn on the collision detection using parameter `colldet`. When collision detection is used, `DYNO` counts the pairs of facets in the model that intersect. Collision detection may take considerable time and should be avoided if possible through smart use of bound constraints on the individual surface geometry parameters. However, when collisions are possible, use parameter `collfile` to help limit the number of facet-facet intersection tests: each line of the `collfile` (an `.ele` format file) should specify two indices that are integer facet ID’s linking to those in the `.ele` file that specifies the surface-based model (parameter `modelfile` specified in the `meshinp` input file). The `collfile` should specify pairs of facet ID’s that COULD possibly intersect and SHOULD be checked during the inversion; any pairs not specified will not be checked. For example, if the `modelfile` contains facets with ID’s (first attribute) ranging from 1 to 4 then you might have a `collfile` that looks like this:

```
6 2 0 # <number of lines below=6> <number of indices per line=2> <number of attributes=0>
1 1 1
2 1 2
3 1 3 # <index> <facet ID> <facet ID>
4 2 2
5 2 3
6 3 3
```

which indicates that no intersections are possible with any facet with ID equal to 4 but all other intersections are possible, including between pairs of facets with equal ID values (other than for ID’s equal to 4). If the `collfile` is not provided then

collision detection will check every possible pair of facets for intersection, which may take considerable time. To help design the `collfile`, the program `check_dyno.bounds` can be used to determine the facet group pairs that could intersect.

To use these built-in constraints, you must make sure that you set the `ncon` parameter in the [optimization engine specification file](#) to a consistent value on $[0, 3]$. If parameter `conobjs` is "t" (true) then `ncon` should be incremented by 1; if parameter `minangle` is on (0, 180) then `ncon` should be incremented by 1; if parameter `colldet` is "t" (true) then `ncon` should be incremented by 1.

Writing your own constraint subroutines

Please contact me for further instructions.

The objective function

Currently, the objective function minimized in a surface-geometry inversion only includes the data misfit term. If `chifact` equals zero, the `ObjBestPop` value in the terminal output indicates the misfit value. In that case, you probably want to allow the inversion to continue minimizing the misfit term until the maximum number of iterations are reached, or the stall convergence tests (if used) are triggered.

If `chifact` is non-zero, the `ObjBestPop` value in the terminal output indicates the following quantity:

$$\text{ObjBestPop} = \frac{|\Phi_d - \text{chifact}|}{\text{chifact}}$$

where Φ_d is defined in [Section 4.5.1](#) as the standard sum-of-squares misfit divided by the number of data. Hence, `ObjBestPop` is now the relative misfit value (relative to the target value). In this case, you probably want to stop the inversion once the `ObjBestPop` value gets below some tolerance, equivalent to parameter `chitol` for a VIDI inversion, however for DYN0 you would use the `targ` parameter instead, which is specified in the optimization input file `optiminp`. For example, set `targ` to 0.05 if you want the final misfit within 5% of the desired target.

Writing your own regularization subroutines

Please contact me for further instructions.

Chapter 6

Change Log

The modifications indicated in the table below are only those that have changed the functionality of one or more programs mentioned in this document. Only major bug fixes are indicated, i.e. those that would significantly change the results of a program.

Date* yyyy/mm/dd	Revision*	Description of change
2018/01/08	3260	Programs <code>fwd</code> , <code>vinv</code> and <code>winv</code> are now named <code>fogo</code> , <code>vidi</code> and <code>dyno</code> .
2016/03/02	2276	Changes to program <code>vinv</code> to move the weights outside the regularization measures by default.
2016/07/18	2399	New inversion option <code>smooth_wts_inside</code> for program <code>vinv</code> .
"	"	The smoothness weighting file for rectilinear meshes in program <code>vinv</code>
"	"	can now be the UBC-GIF format, with or without smoothness weights.
"	"	Extended program <code>fwd</code> to calculate sensitivities for first arrival traveltime data.
"	"	Added <code>ai</code> command line parameter to program <code>fwd</code> .
"	"	Added functionality for multi-component mag (new formulation xyz).

* The dates correspond to those when the public user repository was updated. The revision number is that for the main SVN repository accessed by users in my research group. Public users access a different repository so have different revision numbers. This is currently only important when running the inversion programs (see parameter `revision`).

Bibliography

- A. Carter-McAuslan, et al. (2015). ‘A study of fuzzy c-means coupling for joint inversion, using seismic tomography and gravity data test scenarios’. *Geophysics* **80**(1):W1–W15, doi:10.1190/geo2014-0056.1.
- K. Deb, et al. (2002). ‘A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II’. *IEEE Transactions on Evolutionary Computation* **6**(2):182–197, doi:10.1109/4235.996017.
- E. Fregoso & L. A. Gallardo (2009). ‘Cross-gradients joint 3D inversion with applications to gravity and magnetic data’. *Geophysics* **74**(4):L31–L42, doi:10.1190/1.3119263.
- L. A. Gallardo & M. A. Meju (2004). ‘Joint two-dimensional DC resistivity and seismic travel time inversion with cross-gradients constraints’. *Journal of Geophysical Research: Space Physics* **109**(B3):B03311, doi:10.1029/2003JB002716.
- P. G. Lelièvre & C. G. Farquharson (2013). ‘Gradient and smoothness regularization operators for geophysical inversion on unstructured meshes’. *Geophysical Journal International* **195**(1):330–341, doi:10.1093/gji/ggt255.
- P. G. Lelièvre, et al. (2011). ‘Computing first-arrival seismic traveltimes on unstructured 3D tetrahedral grids using the Fast Marching Method’. *Geophysical Journal International* **184**(2):885–896.
- P. G. Lelièvre, et al. (2012). ‘Joint inversion of seismic traveltimes and gravity data on unstructured grids with application to mineral exploration’. *Geophysics* **77**(1):K1–K15, doi:10.1190/geo2011-0154.1.
- P. G. Lelièvre & D. W. Oldenburg (2009). ‘A comprehensive study of including structural orientation information in geophysical inversions’. *Geophysical Journal International* **178**(2):623–637, doi:10.1111/j.1365-246X.2009.04188.x.
- Y. Li & D. W. Oldenburg (1998). ‘3-D inversion of gravity data’. *Geophysics* **63**(1):109–119.
- Y. Li & D. W. Oldenburg (2000a). ‘Incorporating geological dip information into geophysical inversions’. *Geophysics* **65**(1):148–157, doi:10.1190/1.1444705.
- Y. Li & D. W. Oldenburg (2000b). ‘Joint inversion of surface and three-component borehole magnetic data’. *Geophysics* **65**(2):540–552, doi:10.1190/1.1444749.
- M. Okabe (1979). ‘Analytic expressions for gravity anomalies due to homogeneous polyhedral bodies and translations into magnetic anomalies’. *Geophysics* **44**(4):730–741.
- J. Sun & Y. Li (2017). ‘Joint inversion of multiple geophysical and petrophysical data using generalized fuzzy clustering algorithms’. *Geophysical Journal International* **208**:1201–1216.
- N. C. Williams (2008). *Geologically-constrained UBC-GIF gravity and magnetic inversions with examples from the Agnew-Wiluna greenstone belt, Western Australia*. Ph.D. thesis, Dep. of Earth and Ocean Sciences, The University of British Columbia, Vancouver, British Columbia, Canada.