InvertTrishear

v. 3

by David Oakley © 2021

License and Disclaimer

In addition, to the terms of the GPL license, it is asked that use of this software in any academic or other publications be acknowledged and that appropriate papers be cited.

Questions about any aspect of this software or its licensing should be addressed to David Oakley at david.o.oakley@uis.no.

Table of Contents

1. **Introduction**

InvertTrishear is a program for fitting trishear fault propagation fold models to data. It works for models involving a single fault only. Several different types of data, fault geometry, and inversion methods can be used (see section 2). The program has been developed over several years during and following my Ph.D. work at Penn State. The program uses the trishear velocity field equations of Zehnder and Allmendinger (2000) and the rate of change of dip equation of Oakley and Fisher (2015). As of version 2, the program uses a rapid, quasi-analytic solution method for the s = 1 case, which is described in Appendix C of my Ph.D. dissertation (Oakley, 2017). The method of inverse modeling, or restoring trishear folds in order to test possible models, comes from Allmendinger (1998). The various data inversion methods that the program is capable of using come from different sources, which are cited in the relevant sections of this manual. Some understanding of trishear fold kinematics is advised when using this program, so that it is not treated as a black box. The papers cited in this paragraph will provide a good starting point for the interested user.

Several conventions used by the program should be noted in order to avoid difficulties:

Trishear Velocity Field: The velocity field used is that of Zehnder and Allmendinger (2000), which is not the only possible trishear velocity field. Users of Move should note that the trishear velocity field used in that software (from Hardy and Ford, 1997) is slightly different from the Zehnder and Allmendinger (2000) solution. Differences are typically small, but noticeable.

Fault Parallel Flow: Outside of the trishear zone, the program assumes fault parallel flow. Fold axes bisect bends in the fault, and slip is conserved across fault bends. This will match the fault-bend folding theory of Suppe (1983) only when stratigraphic horizons and bedding dips are parallel to the lower fault segment.

Dip Directions: The program uses the convention that bedding dips down to the right in cross-section are positive and those down to the left are negative. This holds for dip data and for regional dips used to define the expected restored state.

Fault Dips: The fault ramp angle is defined using an angle from 0° to 180°, measured up from the horizontal x-axis. Thus faults that dip down to the left fall in the range 0° to 90°, and those that dip down to the right fall in the range 90° to 180°.

If you have questions or comments about this software, or if you discover any bugs in it, contact me (David Oakley) at david.o.oakley@uis.no.

2. Model Setup

Running InvertTrishear requires specifying the data to be used, the fault model, the parameter space, the data inversion algorithm, and a variety of options. This can either be done directly when running InvertTrishear (in which case the options chosen will not be saved) or by using Setup_InvertTrishear, which creates a text file that InvertTrishear can read. Using Setup_InvertTrishear is recommended in most cases, as it creates a record of what options were used and allows for reuse of the options file. The questions asked by Setup_InvertTrishear are nearly identical to those asked by InvertTrishear if inputting options manually, and any differences are noted below.

The program runs in a console or command prompt window and has no graphical interface. To setup a run, you will be asked a series of questions or prompted to enter values or file names. All model runs require creating data files and a parameters file, in addition to specifying options for the program. These are simply text files.

Note: When specifying the names of text files, the file extension (such as .txt) must be included in the name.

2.1 Options file name (Setup_InvertTrishear only)

When running Setup_InvertTrishear, the first prompt is:

Enter name of a file to save the options to:

This is the name of a text file in which the options specified during setup will be saved. This file will be created in the same folder that Setup_InvertTrishear is in.

2.2 Data

<u>2.2.1 Number of Data Types</u>

The next prompt is:

Enter number of different data types:

For this prompt, enter a number between 1 and 5, specifying how many different types of data will be used to constrain your model.

<u>2.2.2 Data Types</u>

After entering the number of data types, you will be prompted to specify what types of data they are:

Data Are (list all that apply):
(1) beds
(2) dips
(3) points on a fault
(4) marine terraces
(5) points on restored state beds

Enter any or all of the numbers 1 through 5, separated by commas. The total number of data types specified must match the number of data types entered for the previous prompt.

2.2.3 Data Files

For each data type chosen, you will be prompted to enter a data file name:

'Input bedding file name (with extension)'

'Input dip file name (with extension)'

'Input fault file name (with extension)'

'Input terrace file name (with extension)'

'Input restored bedding file name (with extension)'

As the prompts suggest, you must include a file extension with the file name. The data files are all text files, listing the coordinates of the data points in the 2-dimensional Cartesian coordinate system of the cross section, along with any other necessary information, such as dip. Data must be projected on a cross-section and converted to a 2-dimensional coordinate system prior to using this program.

It is best to leave a blank line at the end of each data file. Failure to do this, may result in the last line not being read. Adding too many blank lines, however, may cause the program to crash.

2.2.3.1 Beds

Bed data consists of points along a marker bed or contact. They may be the digitized trace of a bed, as from a seismic interpretation, or they may be contact points mapped on the surface. The data file is a two column text file and may contain multiple beds. At the beginning of each bed, put the word “bed” in the first column and a name to identify that bed in the second column. Below that, list the coordinates of all points in the bed, with x coordinates in the first column and y coordinates in the second. Repeat for all additional beds.

Growth strata can also be included in the model. For growth strata, replace the word “bed” in the data file with the word “growth”. For each growth bed, you will need to include a parameter to determine the amount of slip necessary to restore the bed. The program will reject any models in which the slip to restore the growth strata is greater than that to restore the pregrowth strata.

If this data type is chosen, you will see the prompt:

Do the beds include growth strata? (0/1)

Enter 0 for no or 1 for yes. If yes (1), this prompt will be followed by:

How is slip on growth strata parameterized?

(1) Slip is total for each growth layer.

(2) Slip is additional since previous layer.

(In order in data file from top to bottom. Put youngest on top if using this option).

Each growth bed requires an additional parameter in the parameters file, specifying at what point in the slip history of the fault the growth layer is added. If option (1) is chosen this parameter is the total amount of slip after the growth layer is deposited. If option (2) is chosen, then the parameter is the amount of slip on the fault since the deposition of the last growth bed (or since the start of faulting for the oldest growth bed). If option (2) is used, then the growth beds in the data file must be organized by age with the youngest on top.

2.2.3.4 Dips

Dip data represent the apparent dip of a bed within the plane of the cross section. Strike and dip measurements must be projected to the cross section in another program (or by hand). The dip data file contains three columns: x-coordinates, y-coordinates and dips. No headings are needed. The convention used by the program is that dips are positive if dipping down to the right and negative if dipping down to the left.

2.2.3.5 Points on a Fault

These data consist of points known (from field, well, or seismic data) to lie on the fault. Any number of points may be used. The data file is simply a two-column text file, listing x and y coordinates. No headings are necessary, since there can only be one fault in a given model.

2.2.3.6 Marine Terraces

These are points on a marine terrace, that has been uplifted by the fault. An additional parameter is needed for each terrace, in order to determine the amount of slip necessary to restore it. The data file is like that for beds, except that the word “bed” is replaced by the word “terrace”. The first point of each terrace is assumed to represent the inner edge of the terrace. The use of this feature is presented in Oakley et al. (2018).

2.2.3.7 Point on Restored State Beds

These are points that should be on the beds in the restored state. Thus they don’t move but are tested against the restored bed geometry. This might be used, for example, for far-field points that are not expected to be affected by folding but that constrain the restored bed geometry. Data files are formatted in the same way as for beds. To make use of data of this type, you must also have regular, deformed-state “beds” data, and each restored state bed must have the same name as one of the deformed state beds, to which it will be compared. If the name of a restored state bed does not match one of the deformed state beds, then it will be ignored.

2.3 Fault Tip

After entering all data files, the next prompt is to choose which fault tip position to solve for.

Tip position is:
(1) Initial
(2) Final

The initial tip position is the position of the fault tip before propagation begins, and the final tip position is the position of the fault tip after fault propagation (and slip). The model can search for either one, but the parameter space should be chosen accordingly. If one is known, the other can be calculated based on the fault geometry. Note that some later options may only be available for one of these choices.

After choosing the fault tip position to fit for, you will receive a prompt asking if you want to put constraints on the fault tip that isn't being fit for as a model parameter:

Do you want to place constraints on the final/initial tip position too? (0/1)

where final/initial will be whichever of those two you didn't choose in the above prompt. Enter 0 for no or 1 for yes. **Warning:** This option only works for fault type options 1, 6, 7, or 8 below (you will see a warning message about this if you choose 1), so if you choose 1, make sure to choose one of those types when you get to the fault type prompt.

If you have chosen (1), you will be offered two options on how to constrain the non-parameter fault tip:

Do you want to:
(1) Constrain the tip position within a box?
(2) Constrain the tip position relative to a line?

If you choose (1) you will be prompted to

Enter tip position limits as: xmin,xmax,ymin,ymax.

while if you choose (2) you will be prompted to

Enter line as: line dip (in degrees, positive down to left), y-intercept.

In either case, enter the required values to define the box or the line. In the case of the line, you will also see the prompt:

Must the tip be (1) above / left of the line or (2) below / right of the line?

or

Must the tip be (1) above / right of the line or (2) below / left of the line?

depending on which way the line dips. In either case, enter 1 or 2 to define which side of the line the non-parameter fault tip must be on. Also note that if the line is vertical the options will only be "(1) left or (2) right," and if the line is horizontal, they will be "(1) above or (2) below."

2.4 Fault Types

The program is capable of modeling several different types of faults. You will be prompted to choose a fault type.

Choose Fault Type

(1) Straight Fault Ramp
(2) Ramp from Horizontal Detachment
(3) Fault with a bend in it
(4) Listric Fault (fault parallel flow)
(5) Parallel Fault Propagation Fold
(6) Multi-bend fault
(7) Listric Fault (circular or elliptic, approximate)
(8) Spline Fault (Requires tip position is final)

Note: The multi-bend fault (type 6) is a more general geometry that includes options 1, 2, and 3 as special cases. The lower numbered options were created first, however, and are retained for backwards compatibility.

2.4.1 Straight Fault Ramp

This is the simplest fault type: just a straight ramp extending infinitely far down. It will produce a monocline, with a trishear zone in the forelimb. Model parameters are fault tip position (x, y), total slip, ramp angle, ϕ, P/S, and the concentration factor s.

2.4.2 Ramp from Horizontal Detachment

This is a simple step fault-propagation fold, in which the fault ramp steps up from a horizontal detachment. This will produce a fold with a straight backlimb, a flat crest, and a curved, trishear fold forelimb. In addition to the parameters for the straight fault ramp, this model adds detachment depth as an additional parameter.

After choosing this model, you will see the prompt:

Must detachment depth be at initial fault tip depth? (0/1)

Enter either 0 (no) or 1 (yes). If no, the initial position of the fault tip may be anywhere above the detachment. If yes, the fault tip must propagate up from an initial position at the detachment. If you answer yes, you should not include detachment depth in the parameters file, since the program knows it will be the same as the initial y tip position.

2.4.3 Fault with a Bend in it

This fault model has two straight-segments. Unlike the horizontal detachment model, neither segment is required to be horizontal. Currently, the program only allows models in which the upper segment is steeper than the lower one. Models in which the lower segment is steeper will be rejected. This model adds two additional parameters: the elevation at which the bend occurs and the ramp angle of the lower segment.

Two prompts will appear after choosing this model:

Must depth of fault bend be at initial fault tip depth? (0/1)

This is analogous to the detachment depth at initial fault tip prompt. 0 is no and 1 is yes. It may be useful if, for example, you wish to have a fault propagating up from a low-angle, but not horizontal detachment. If choosing yes, you do not have to include bend elevation as a parameter. Note that this option is only available if you are solving for the initial fault tip position.

Can initial tipy be below fault bend? (0/1)

This prompt appears only if choosing no to the above prompt or if the fault tip position to solve for is final. For this prompt, choosing no (0) indicates that the initial position of the fault tip must be above the fault bend. Choosing yes (1) allows the fault tip to start above or below the fault bend.

2.4.4 Listric Fault (fault parallel flow)

Warning: This type has not been tested very well, and I think there are problems with it. It is recommended to use option 7 instead if you want to model listric faults. It can do essentially the same things as this fault type (with the caveat that the circular portion is approximated by straight segments rather than truly circular), and it has been better tested.

This model creates a circular listric fault, with a trishear zone at its tip. The fault shallows into a horizontal detachment at depth. At its upper end, it becomes straight after reaching a specified maximum ramp angle. This model has been less well tested than the previous three, so use with caution. Model parameters are (in the order that they appear in the parameters file): fault tip position (x, y), total slip, ramp angle, ϕ, P/S, the concentration factor s, the radius of curvature of the circular part of the fault, and the detachment depth.

This model assumes fault parallel flow along both the circular and straight parts of the fault. Linear velocity is constant for all locations outside of the trishear zone, so angular velocity is not constant. This is different from models that use either inclined shear (Cardozo and Brandenburg, 2014) or rigid rotation of a basement block (Erslev, 1986; Seeber and Sorlien, 2000) and should not be used where those models are thought to be more appropriate.

If you have chosen to solve for the initial position of the fault tip, one additional prompt will appear after choosing this model:

Must detachment depth be at initial fault tip depth? (0/1)

This is analogous to the same prompt for the ramp from a horizontal detachment fault model. 0 is no and 1 is yes. If no, the initial position of the fault tip may be anywhere above the detachment. If yes, the fault tip must propagate up from an initial position at the detachment. If you answer yes, you should not include detachment depth in the parameters file, since the program knows it will be the same as the initial y tip position. Note that this prompt will not appear if you are solving for the final tip position.

2.4.5 Parallel Fault Propagation Fold

This is the parallel fault propagation fold model of Suppe and Medwedeff (1990), which is a kink-band model and does not use trishear. The fault in this case consists of two linear segments, but we do not require the lower segment to be horizontal. In this program, we assume no layer parallel shear (S_p = 0 and S_b = 0). In addition, we assume that the pre-folding dip of the beds is 0. Thus θ_1 amd θ_2 are assumed to be equal to the ramp angles, and γ and γ_1 are assumed to be measured from the horizontal. In cases where those assumptions do not apply, this model should not be used.

Model parameters are (in the order that they appear in the parameters file): fault tip position (x, y), total slip, ramp angle (upper segment), ramp angle (lower segment), and the bend depth (only if not at the initial fault tip position).

One additional prompt will appear after choosing this model:

Must depth of fault bend be at initial fault tip depth? (0/1)
Choosing 0 may not work properly.

0 is no and 1 is yes. As with other fault models, choosing yes means that the fault tip is required to start at the bend in the fault and to propagate up from there. As the prompt notes, however, there may be problems if you choose no, as little work has been done on accommodating this option. The model of Suppe and Medwedeff (1990) seems to assume that fault propagation begins at the bend, so choosing that option is recommended for now.

Note: This model is purely for reverse faults. It will not work properly if you assign a normal sense of slip.

2.4.6 Multi-Bend Fault
Warnings:

(1) This model does not deal properly with intersections of backlimb syncline axes. If using only two fault segments or using inclined shear (in which all axes are parallel), this will not be a problem, but care should be taken if using more than two fault segments with fault parallel flow or fault-bend folding. In these cases, it is recommended to check that any intersections occur above the area of interest.

(2) It is possible to make very complex models with a large number of fault segments and changes in trishear parameters. If doing so, note that even efficient data inversion methods may fail to fully explore such high-dimensional parameter spaces.

This is a trishear fault-propagation fold with an arbitrary number of bends in the fault. This model can duplicate the functionality of the “straight fault ramp” (one segment), “ramp from horizontal detachment” (two segments, with the ramp angle for the second one required in the parameters file to be flat), and “fault with a bend in it” (two segments). It can also be used for models with more than two fault segments. In addition to fault bends, it can allow phi and P/S to change during fold growth.

Model parameters are (in the order that they appear in the parameters file): fault tip position (x, y), total slip, all ramp angles in order from the highest segment to the lowest, all fault bend elevations in order from the highest to the lowest, all phi values in order from the ones corresponding to a tip in the highest segments to the lowest, all elevations at which phi changes in order from highest to lowest, all P/S values in order from the ones corresponding to a tip in the highest segments to the lowest, all elevations at which P/S changes in order from highest to lowest, the concentration factor s, and the shear angle (for inclined shear only). For each of the three things that can change (ramp angle, P/S, and phi), the number of change elevations must be one less than the number of values. For example, a fault with three segments of different dips (i.e. two ramp angle changes), one change in P/S, and a constant phi value will have three ramp angles, followed by two bend elevations, then two P/S values, then the elevation at which P/S changes, then one phi value.

After choosing this fault type, you will be prompted to enter the number of fault segments, phi values, and P/S values:

Enter number of fault segments with different ramp angles
Enter number of different phi values
Enter number of different P/S values

For each of these prompts, enter the number of different values. This must match the number of entries for that parameter in the parameters file.

You will next see the options:

Choose type of backlimb deformation:
(1) Fault-Parallel Flow (Fold axes bisect fault bends)
(2) Fault Bend Folding (Preserves line length)
(3) Inclined Shear

This allows you to choose the type of deformation that occurs in the hanging wall of the fault outside of the trishear zone and thus shapes the backlimb of the fold. Fault-parallel flow is the model used for the previous trishear fault types, which assumes that fold axes bisect all fault bends (Ziesch et al., 2014). This will preserve fault slip across fault bends. Fault bend folding uses the fault-bend folding theory of Suppe et al. (1983) to determine the orientation of the fold axis. Important note: The program assumes for this calculation that the pre-folding bedding orientation is horizontal. Assuming that is the case, this method will preserve line length across fault bends. Finally, inclined shear assumes that all fold axes dip at the same specified angle and the backlimb deforms by simple shear (White et al., 1986). If the shear is vertical, then this method will preserve heave across fault bends. This program uses the convention that 90° is vertical and 0° is horizontal. This is the same convention used in Move, but the opposite convention is used in some papers on inclined shear deformation.

After choosing the fault type, you will see the prompt:

Must fault tip stay above a specified fault bend at all times? (0=no, 1=yes)

This allows you to require the fault tip to start and finish above a specified fault bend. For example, if your fault rises up from a detachment, you could use this option to prevent the fault tip from the starting within the detachment. If you choose yes (1), then you will see the prompt:

Enter bend number that tip must stay above: (1 is the highest bend.)

Bends are numbered from highest to lowest. So bend 1 is where the highest and second highest fault segments come together, bend 2 is where the second and third segments join, etc.

If you are fitting for the initial fault tip position (and if you are not using one of the circular, elliptical, or spline fault geometries that subsequently call the multi-bend code), you will also see the prompt

Must initial fault tip position be at one of the bends? (0/1)

This allows you to require the fault tip to start exactly at a bend in the fault. 0 is no and 1 is yes. If you choose 1, you will be asked to choose which bend the tip must start at:

Enter bend number that tip must start at: (1 is the highest bend.)

The bends are numbered the same as above, downwards from highest to lowest. Also note that if you choose this option, you will not need a parameter for the elevation of this bend, since it is required to be at the tip position.

The next prompt will ask you:

Must phi be less than the ramp angle in any fault segment the tip passes through? (0=no, 1=yes)
(This prevents lowering of the footwall due to a trishear zone extending downwards)

If you choose yes, then any models in which phi is greater than the ramp angle in any of the fault segments that the tip passes through will be rejected. The reason you may wish to require this is to prevent the lower boundary of the trishear zone from extending downwards into the earth, which results in far-field lowering of material in the forelimb. If you choose this option, ramp angles less than phi may still be allowed if they occur in a segment below where the fault tip starts.

The next prompt says:

Can the trishear zone boundary intersect the backlimb fold axes above some elevation? (0 = no, 1 = yes)

Ordinarily, the program rejects any models in which the boundary of the trishear zone intersects any of the backlimb fold axes. This is done because the axes separate regions of different material velocity. Allowing an axis to intersect the trishear zone boundary would result in material at different velocities entering different parts of the trishear zone, which would violate the assumption of constant hanging wall velocity that is used by Zehnder and Allmendinger (2000) to derive their trishear velocity field. Choosing yes to this option allows that requirement to be relaxed if the intersection occurs above a specified elevation, which must be above all the data. For instance, if an elevation above ground level is chosen, then intersection at that elevation shouldn’t matter, since no deformation is occurring there anyway. If you do choose yes (1) for this option, you will see the prompt:

Enter elevation above which trishear zone and fold axes can intersect: (must be above all data) to which you should simply enter an elevation. The program will not check that the elevation is above all data. You must check that for yourself to avoid any problems.

The next prompt says:

> Must the final fault tip position be in the uppermost fault segment? (0 = no, 1= yes)
> (If not, some fault segments may go unused.)

This option allows you to add a constraint on the final fault tip position that it be in the uppermost fault segment. Without this, it is possible to have the fault tip end below one of the bends, making segments above it unused.

After this, you will see the prompt:

> Do you want to limit the minimum and maximum x values of fault bends? (0/1)

Fault geometry is parameterized by the dips of segments and the y-coordinates of bends, with an x coordinate only for the fault tip, so the limits in the parameters file do not put any constraints on the x coordinates of fault bends. This fact may result in faults that are unrealistically long in the x-direction. This option allows you to constrain the x-coordinates of fault bends even though they are not model parameters. Choose 0 for no or 1 for yes. If you choose 1, you will see the prompt:

> Enter minimum and maximum allowed x values for fault bends.

For this, enter two values separated by a comma, with the minimum first followed by the maximum.

The next prompt is:

> Must the fault be concave upward at all points? (0 = no, 1= yes (default))

In previous versions of the program, faults were required to be concave upward, and choosing 1 will continue to enforce this requirement. Choosing 0, however, will allow faults that are convex upward along part or all of their length.

The next prompt for this fault type is:

> Do you want to put constraints on the x position of the backlimb syncline? (0/1)

The backlimb syncline is where the backlimb of the fold ends and beds return to their regional line. Its position is controlled by the fold axes. This option allows you to constrain where that syncline should occur and reject models that fail the constraint. As in other options, 0 means no and 1 means yes. If you choose 1, you will also see the prompts:

> Enter elevation (y coordinate) at which to measure the backlimb syncline position.

and

> Enter minimum and maximum allowed x position of backlimb syncline at y.

For the first prompt, enter a single value for the elevation at which to apply the constraint (such as the elevation of the ground surface). For the second, enter the minimum x value for the syncline at the specified elevation followed by the maximum, separated by a comma.

2.4.7 Listric Fault (circular or elliptic, approximate)

This option allows you to model a listric fault with a circular or elliptic shape. The curved shape of the fault is approximated by a number of straight segments. This option will set up a multi-bend fault based on the listric geometry and then call code from the multi-bend fault type.

The fault geometry is defined by a lower straight segment (may or may not be horizontal), a listric segment (circular or elliptical), and an upper straight segment. The dips of the lower and upper segments define the minimum and maximum dips of the listric segment. With a horizontal detachment and circular listric segment, this is the fault geometry used by Cardozo and Brandenburg (2014).

Model parameters are (in the order that they appear in the parameters file): fault tip position (x, y), total slip, maximum fault dip (uppermost segment dip), minimum fault dip (lowermost segment dip), elevation of the base of the listric segment, listric parameters (see below), all phi values in order from the ones corresponding to a tip in the highest segments to the lowest, all elevations at which phi changes in order from highest to lowest, all P/S values in order from the ones corresponding to a tip in the highest segments to the lowest, all elevations at which P/S changes in order from highest to lowest, the concentration factor s, and the shear angle (for inclined shear only). The listric parameters (defining the shape of the listric segment) are just the radius of curvature, but are two parameters for an ellipse. These may be the horizontal (a) and then vertical (b) semiaxis lengths or the elevation of the top of the listric segment followed by the eccentricity (here defined as e = sqrt(1-b^2 / a^2) for positive e and e = sqrt(1-a^2 / b^2) for negative e in order to allow either axis to be the longer one.) Eccentricity should never be >1 or <-1 or you will have problems.

The first prompt for this fault type is:

Shape:
(1) Circle
(2) Ellipse

This simply allows you to choose whether the listric segment of the fault should be circular (1) or elliptical (2). If you choose an elliptical fault geometry, you will see the prompt:

How should the ellipse geometry be parameterized?
(Note: Base and top elevation are where the fault becomes straight.)
(1) base elevation, horizontal semiaxis, vertical semiaxis
(2) base elevation, top elevation, eccentricity

This option allows you to choose between the two variations of the listric parameter described above.

If you are fitting for the initial (rather than final) fault tip position, you will also see the prompt:

Must detachment depth be at initial fault tip depth? (0/1)

Choose 0 for no or 1 for yes. If choosing 1, then the depth (or elevation) of the lower straight segment (detachment if horizontal) will be at the depth (or elevation) of the initial fault tip. This is useful if creating a fault that propagates up from a detachment. If you choose 1, then you will not need a parameter for the elevation of the base of the listric segment, since that is the same as for the fault tip.

After this, you will see the same parameters as for a multi-bend fault. Note that the "number of fault segments with different ramp angles" that you are asked for will now refer to the number of segments that the fault will be discretized into, so a fairly large number should be chosen (I usually use at least 30 and sometimes many more) so that the curved shape of the fault is approximated well.

2.4.8 Spline Fault

Warnings:

(1) This option should only be chosen if you are fitting for the final (rather than initial) fault tip position.

(2) The final fault tip position should be above all spline knots. Otherwise, there are likely to be problems.

(3) This option is not very well tested.

This option is for a fault geometry parameterized by a spline. As for the circular and elliptic listric faults, the fault geometry is defined by upper and lower straight sections with a curved (here spline) section between them. Also, as in those faults, the fault is discretized into a number of straight segments and then run with the multi-bend model. The interpolation is into a fixed number of segments along a constant x interval, rather than a constant distance, which is probably not ideal but is simple to do.

The splines used to define the fault are clamped splines defined by (x,y) coordinates of control points and the fault dip at the two ends. The splines are of the form

$$y_i = a_i + b_i * (x - x_i) + c_i * (x - x_i)^2 + d_i * (x - x_i)^3.$$

The fault tip is not a spline knot. Instead, the fault goes straight above the top knot and the fault tip is the end point of that straight segment.

Model parameters are (in the order that they appear in the parameters file): fault tip position (x, y), total slip, maximum fault dip (uppermost segment dip), minimum fault dip (lowermost segment dip), x coordinates of all spline knots, y coordinates of all spline knots except the uppermost one, all phi values in order from the ones corresponding to a tip in the highest segments to the lowest, all elevations at which phi changes in order from highest to lowest, all P/S values in order from the ones corresponding to a tip in the highest segments to the lowest, all elevations at which P/S changes in order from highest to lowest, the concentration factor s, and the shear angle (for inclined shear only).

Note that the first spline knot is parameterized by only its x coordinate (since the y coordinate can be calculated) while other knots are parameterized by their x and y coordinates. Thus, for example, for a spline with 3 knots, the knot parameters would be: x_1, x_2, x_3, y_2, y_3 where knots are numbered from top to bottom.

The first prompt for a spline fault is:

> Enter number of spline knots (control points).
> This includes the bottom end point, but not the fault tip.

For this simply enter an integer ≥ 2. The more knots you add, the more complex the fault shape can be, but the more model parameters you will have to fit for. Since the fault consists of a spline-shaped section between two straight segments, there must be at least two knots to define the starting and ending points of the spline section.

The number of knots is the only spline-specific prompt. After this, you will see the same prompts as for a multi-bend fault. Note that the "number of fault segments with different ramp angles" that you are asked for will now refer to the number of segments that the fault will be discretized into, so a fairly large number should be chosen (I usually use at least 30 and sometimes many more) so that the curved shape of the fault is approximated well.

2.5 Parameters File

The next prompt is:

Input Parameter File Name (with extension)

This requires you to enter the name of a file defining the parameter space, which you must set up by hand. This is simply a text file, with three columns specifying the minimum, maximum, and step size values for each parameter (in that order), followed by two additional lines. All model parameters must have a minimum value, maximum value, and step size specified. If a specific parameter is known or assumed (such as knowing the ramp angle or assuming that s = 1), then simply enter the minimum and maximum as the same value and enter any step size. The meaning of the step size varies depending on the inversion method chosen. Even in methods for which it is not needed, some value must be put here. For grid based methods, the difference between maximum and minimum values must be a multiple of the step size.

The order of parameters is as follows. Not all parameters will be present for all fault models. Note that the parameter phi is half the trishear apical angle, not the full angle.

Tip x coordinate
Tip y coordinate
Total slip (must be positive)
Fault ramp angle(s) (from uppermost to lowermost)

Bend y coordinates (multi-bend model only)
Base of listric segment y coordinate (circular or elliptical listric faults only)
Radius of curvature (circular listric faults only)
Horizontal semiaxis (elliptical listric faults only with parameterization 1)
Vertical semiaxis (elliptical listric faults only with parameterization 1)
Top of listric segment y coordinate (elliptical listric faults only with parameterization 2)
Eccentricity (elliptical listric faults only with parameterization 2)
Spline knot x coordinates (spline model only)
Spline knot y coordinates (excluding first knot, spline model only)
Phi (may have multiple entries if using multi-bend model or those that call it)
Y coordinates for changes in phi (if multiple phi entries)
P/S (may have multiple entries if using multi-bend model or those that call it)
Y coordinates for changes in P/S (if multiple P/S entries)
Concentration factor s
Radius of curvature (for fault type 4 listric faults only)
Ramp angle (lower segment for fault-bend model only)
Detachment or bend y coordinate (detachment or fault-bend models only).
Inclined Shear Angle (for multi-bend model or those that call it if using inclined inclined shear)
Slip to restore growth strata (one entry for each growth bed).
Slip to restore marine terraces (one entry for each terrace).
Restored state dip(s) (if fitting for these)*
Restored state bend x-coordinates (multi-segment restored beds only)*
Restored bed elevations (if fitting for these)*
Uncertainty in beds (if fitting for uncertainty)†
Correlation length (for correlated errors, if fitting for this)
Second, correlated uncertainty in beds (for results to calculate type 5 only, if fitting for this)†
Uncertainty in dip data (if fitting for uncertainty)
Separate restored state uncertainty in dip data (if including this and fitting for uncertainty)
Uncertainty in terrace data (one entry for each terrace, if fitting for uncertainty)
Uncertainty in terrace inner edges (one entry for each terrace, if fitting for uncertainty)
Uncertainty in fault point data (if fitting for uncertainty)
Uncertainty in restored bed data (if fitting for uncertainty, 2 entries if x and y are separate)

Following the parameters are two additional lines, with only one column each. The first is the sense of slip, which should be 1 for thrust faults and -1 for normal faults. The program has not been extensively tested for normal faults, so use with caution.

The second extra line is for the increment of slip. This is used in the trishear zone and in any other case where slip must be calculated incrementally. This is simply a number, and will be in whatever units the section coordinates are in. A smaller step size will make for a more accurate trishear model but will take longer.

*When fitting for restored bed parameters, the order differs with different choices. If fitting for restored dips and bed elevations for flat beds (option 5 for objective function, below), then list first all dips and then all bed elevation / y-intercepts. If fitting to multi-segment restored beds, then group the parameters by bed. For each bed, enter parameters for: dips of all segments in order from left to right, x-coordinates of all bends from left to right, y-coordinate of the first bend. Then repeat for each subsequent bed.

†When fitting for bed uncertainty, the number and order of entries depends on several choices. If all beds have the same uncertainty, there is only one entry. If all beds have different uncertainties or if groups of beds have different uncertainties, then there is one entry for each bed or each group. If x and y uncertainties are different, then the x entry or entries (for all beds or groups) come first, followed by the y entry or entries.

2.6 Inversion method

This prompt allows one to choose the inversion method to use from any of seven possibilities allowed by the program.

Choose Method:
(1) Grid Search
(2) Grid Monte Carlo
(3) Monte Carlo from a normal distribution
(4) Metropolis-Hastings Algorithm
(5) Adaptive Metropolis
(6) Robust Adaptive Metropolis
(7) Adaptive Parallel Tempering

Warning: I mostly use Adaptive Parallel Tempering (option 7) and to a lesser extent Robust Adaptive Metropolis (option 6). Other options are not as well tested, especially with newer versions of the program.

2.6.1 Grid Search

This is the grid search method described by Allmendinger (1998). A multi-dimensional grid of values is specified in the parameter file, and the program tests all models on the grid. This method is best suited to situations in which some of the trishear parameters are known, and the others lie within a limited range. For many dimensions and large ranges, either the number of models will be very large and thus the model will take a long time to run, or the grid will be too sparse to adequately sample the parameter space.

2.6.2 Grid Monte Carlo

This is a Monte Carlo simulation that randomly draws samples from a grid of values, specified by the parameter file. If choosing this model, you will see the prompt

Enter number of models to run:

For this simply enter the number of models. Typically, a larger parameter space will require more models to sample adequately. This method requires fewer model evaluations than the grid search to achieve a similar quality of result, but it can still grow unwieldy when the parameter space and number of dimensions are large.

2.6.3 Monte Carlo from a normal distribution

This method also takes random samples, but it does so from a multivariate normal distribution. Samples are drawn from the continuous parameter space, not from a grid, so the step size in the parameter file is ignored. Choosing this method will bring up two additional prompts:

Enter number of models to run:

and

Enter filename to read proposal distribution from:

The first is simply the number of samples to take. The second asks for a text file which will define the normal distribution from which samples are to be drawn. The file must have a number of columns (N) equal to the number of model parameters. The first line lists the mean values for each parameter for the normal distribution. This must be followed by a blank line. After that, the next N lines and N columns give the covariance matrix for the multivariate normal distribution.

This method is useful if one already has a good estimate of the best model parameters. It is not useful for parameters that are poorly constrained a priori or for multimodal probability density functions.

2.6.4 Metropolis-Hastings Algorithm

This is the Metropolis-Hastings Markov chain Monte-Carlo (MCMC) method. The proposal distribution is a multivariate normal distribution, with standard deviations given by the step size for each parameter and with covariance assumed to be zero. In my experience, I tend to get very low acceptance rates, but in some situations this algorithm may work well. Nonzero covariance might help, and may be added to a future release, but in general it is difficult to choose a good proposal distribution without some prior knowledge of the expected target distribution.

When choosing this algorithm, one will see the prompts

Enter number of models to run:

and

Should initial model be (1) random or (2) specified?

The number of models is self-explanatory. A random initial model will be chosen from a uniform distribution over the parameter space. A specified initial model is useful if one has some prior constraint on the expected values of the model parameters or if the model has a lot of constraints (beyond just the parameter limits) such that a random initial model may be an impossible one. If choosing a specified initial model, one will then see

Enter initial values for all N parameters:

where N will be replaced by the number of model parameters. Initial values for all parameters should then be entered in order (same order as the parameters file) separated by commas. Initial values should be within the limits of the parameter space.

Warning: There may be some cases in which additional parameters are added after this prompt. In that case, this method will not work. This is fixed for the Adaptive Parallel Tempering method but not yet for other methods.

2.6.5 Adaptive Metropolis

This is the adaptive metropolis (AM) algorithm of Haario et al. (2001). The proposal distribution is a multivariate normal distribution, with initial standard deviations given by the step size for each parameter and with initial covariance assumed to be zero, but the covariance matrix is adapted as the run proceeds. The scaling parameter (s_d) is $(2.4)^2/d$, as suggested by Haario et al. (2001), where d is the number of dimensions of the parameter space, which is the number of parameters, excluding any parameters for which minimum and maximum values are the same. In my experience, I have not had very much success with this algorithm, but I have not tested it extensively.

Additional questions that the user will be prompted to answer for this algorithm include

Enter number of models to run:

and

Should initial model be (1) random or (2) specified?

both of which are the same as for the Metropolis-Hastings algorithm. If the initial model is specified, this is followed by

Enter initial values for all N parameters:

Warning: There may be some cases in which additional parameters are added after this prompt. In that case, this method will not work. This is fixed for the Adaptive Parallel Tempering method but not yet for other methods.

These questions are followed by the prompt

Enter model number after which to begin using adapted covariance matrix

The algorithm will initially run as a standard Metropolis-Hastings algorithm, up to a number of models specified by the user, at which point it will begin to adapt the covariance matrix.

2.6.6 Robust Adaptive Metropolis

This is the robust adaptive metropolis (RAM) algorithm of Vihola (2012). As for the AM algorithm, the proposal distribution is a multivariate normal distribution, with initial standard deviations given by the step size for each parameter and with initial covariance assumed to be zero, but the covariance matrix is adapted as the run proceeds. In this case the covariance matrix is adapted to target an acceptance rate of 0.234. I have found that this algorithm works well when the target distribution is unimodal but not as well for multimodal distributions.

The prompts for this algorithm are the same as for AM:

Enter number of models to run:

and

Should initial model be (1) random or (2) specified?

If the initial model is specified, this is followed by

Enter initial values for all N parameters:

Warning: There may be some cases in which additional parameters are added after this prompt. In that case, this method will not work. This is fixed for the Adaptive Parallel Tempering method but not yet for other methods.

2.6.7 Adaptive Parallel Tempering

This is the adaptive parallel tempering (APT) algorithm of Miasojedow et al. (2013). This algorithm uses multiple chains at progressively higher “temperatures,” meaning that the higher temperature chains can more easily move about the parameter space. The target density, π is replaced with a tempered density, π^{β}. β is 1 for the lowest temperature level and approaches 0 at the higher levels. Chains are allowed to swap states, which results in the higher temperature chains exploring the parameter space to find probability maxima and the lower temperature chains exploring the vicinity of these maxima. The final results that are saved are the points visited by the lowest temperature ($\beta = 1$) chain. The covariance matrix of each chain is adapted using the RAM algorithm targeting a 0.234 acceptance ratio, and the β values are adapted as well, targeting an acceptance ratio of 0.234 for swaps. I have found this algorithm to be the most consistently useful. It is especially preferred for distributions that may be multimodal, such as when there is a large parameter space with poor initial constraint on the parameter values. The drawback to this algorithm is that it is slower than the AM and RAM algorithms, although it remains much faster than the grid search.

As for the preceding three methods, the program prompts the user to

Enter number of models to run:

In this case, the number of models entered is the number per chain and is the number of results that will be saved. The total number of models that will actually be run by the software is this number times the number of temperature levels. Thus, this algorithm will take longer than others to obtain an equivalent number of results.

The next prompt is:

Do you want to save only every n models? (0/1)

This allows you to subsample the results and only save the model parameters at some specified interval. This will reduce correlation between samples in the Markov chain and produce more nearly independent samples of the probability density function. It will also result in a smaller results file size for a given number of models run. 0 is no and 1 is yes. If you choose yes (1), then you will be asked to

Enter interval at which to save model results:

For this simply enter an integer (n). Only every nth model result will be saved.

After this, as for similar methods, one is asked

Should initial model be (1) random or (2) specified?

If the initial model is specified, this is followed by

Enter number of model parameters: (currently expecting at least N)

If parameters for restored beds (such as dips, elevations, etc.) are included, the program will not know about them yet. For that reason, you are prompted to enter the correct number of parameters here so the program knows how many initial values to expect. You are then prompted:

Enter initial values for all N parameters:

This is followed by another prompt:

Enter number of temperature levels to use:

This is simply the number of chains that will be run and allowed to swap states. The number must be an integer greater than or equal to 2. More temperature levels can better search the parameter space to identify multiple probability maxima and avoid getting stuck in local maxima. The more temperature levels that are used, however, the longer the program will take to run (all this is mitigated by using parallel processing).

2.7 Objective Function

After specifying an algorithm to use, the user will be prompted to choose what objective function the data should be fit to.

Objective Function:
(1) Fit to flat line.
(2) Fit to known dip.
(3) Fit to best fit line / mean dip
(4) Fit to known line.
(5) Fit for restored dips and/or bed elevations as model parameters.
(6) Multi-segment restored beds.

This choice determines how the error is calculated for bed and dip data.

2.7.1 Fit to flat line

In this case, the data are fit to a horizontal line. For beds, the best-fit horizontal line to the restored points is calculated for each bed, and the error is the distance from each point to this line. For dips, the error is the difference between each dip and 0.

2.7.2 Fit to known dip

In this case, the expected dip of the restored data is known, but it is not necessarily 0. This is useful if there is a regional dip to bedding that is independent of the structure being modeled. This option will prompt the user to

Enter regional dip in degrees. (Negative is down to the left, positive right)

This should be a number between -90 and 90, which should be negative if the restored regional dip is down to the left and positive if it is down to the right. For beds, the best-fit line with this dip will be calculated (slope of the line is -tan(dip), given the sign convention). Errors are the distances between the restored points and this line. For dips, errors are the difference between the restored dip and the expected regional dip.

2.7.3 Fit to best fit line / mean dip

This option allows the program to calculate the best-fit dip and fit to that. For beds, a best-fit line (slope and intercept) will be calculated for each restored bed and errors calculated from the distance of restored points to that line. For dip data, the error will be the difference between each dip and the average dip. An important limitation to this method is that each bed will be fit separately and dip data will be fit separately from bed data. Thus it does not fit a single best regional dip to all the data. Typically, it should be used only if data are points along a single bed (or multiple beds not required to be perfectly parallel) or are dip data only. A second limitation is that the program does not currently output the regional dip that was fit.

Choosing this option will produce the prompt:

Do you want to put limits on restored bed dips? (0=no, 1=yes)

This option allows you to limit the allowed best-fit restored-state dips to some reasonable values so that models that give values outside that range will be rejected. If you choose 1, you will see the prompt.

Enter number of beds:

After entering the total number of beds in the dataset, you will get the prompt:

Note: Bed dips are negative down to left and positive down to right.
Enter minimum dips for all N beds:

Enter all the minimum dip values separated by commas in the same order as the beds in the beds data file. After this, you will see the prompt:

Enter maximum dips for all N beds:

for which you should enter all the maximum dip values.

2.7.4 Fit to known line

This option is only available if one of the chosen data types was beds. For this method, one must know both the expected regional dip of restored beds and the y-intercepts of the lines to which all beds in the beds data file should be fit. Errors are the distance from the restored point to the known line for points along a bed and the difference between the restored dip and the known regional dip for dip data. If the regional dip is 0°, then a known line equates to knowing the correct restored / undeformed elevations of horizons, but if it is not 0° then known restored /

undeformed elevations at a given point must be extrapolated to calculate a y-intercept at x = 0 within the cross-section coordinate system. The uncertainty introduced by this extrapolation is not currently considered by the model, so extrapolating over long distances is not recommended.

Choosing this option will lead to the prompts

Enter regional dip in degrees. (Negative is down to the left, positive right)

which is the same as for the Fit to known dip option. This is followed by

Enter number of beds:

This prompt only appears in Setup_InvertTrishear, not when entering options manually in InvertTrishear, since in the latter case the program will already know the number of beds. For this, simply enter the number of different beds in the bed data file. After this, you will be asked to

Enter y intercepts of all N beds in order

where N will be replaced by the number of beds. For this prompt, enter the y-intercepts of the beds, in the same order that they occur in the beds data file, separated by commas.

2.7.5 Fit for restored dips and/or bed elevations as model parameters

This option allows you to fit for expected restored dips or bed elevations as parameters. Parameter ranges and step sizes will need to be specified in the parameter file. This option is often preferable to the “Fit for dip” option, since it allows one to limit the range of allowed dips and provides output (by way of the model parameters) telling what the expected pre-folding dips that best fit the model are. The downside, however, is that it increases the model dimension, sometimes considerably if there are many beds.

If one chooses this option, it will be followed by several additional prompts. The first of these is:

Fit for dips? (0/1)

Since one can fit for expected dips and/or bed elevations, this question simply asks if one would like to include dips. As usual, 0 is no and 1 is yes. If one chooses no, one will be prompted to enter a single regional dip to fit all data to, in the same manner as if one had chosen “Fit to known dip.” If one chooses yes, and if the data include beds, then one will see the additional prompt:

For bed / contact data:
(1) Use same restored dip for all beds (and dip data).
(2) Fit dip separately for each bed.
(3) Fit dips for groups of beds.

If one chooses the first option, one is fitting for a single expected restored dip for all data. Only one line needs to be added to the parameters file for this. This option is best if all units in an area are expected to be parallel but may have a regional dip. If one chooses the second option, the program will fit separately for each bed. A line for the dip for each bed (in the order that they appear in the data file) is thus needed in the parameters file. In addition, if dip data are included in the model as well as bed data, then another line in the parameters file is needed in order to fit for the expected restored dip for the dip data, which will come after the lines for the beds. Dip data must all be fit to the same expected dip. If one chooses the third option, then subsets of the beds can be fit all to the same dip. This option will be followed by the prompt

Enter number of groups:

to which one should enter an integer specifying the number of groups. All the beds in a group must be in order in the data file. After entering the number of groups, you will be asked to enter the number of the first bed in each group. For the first group, this number should be 1. All beds from 1 until the bed before the first bed of the second group will be considered part of group 1. Then all beds from the start of the second group until the bed before the start of the third group will be considered part of group 2, and so on. All beds from the start of the last group until the end of the beds data file will be considered to be part of that group. In the parameters file, you will need to include one entry for the dip of each group. If you are also fitting dip data, you will need an additional entry for the dip to which those data should be restored, which will be the same for all the dip data.

The next prompt will be:

Fit for y intercepts of beds (0/1)?

This is asking if one wishes to fit for the expected restored elevation of the beds. As usual, 0 is no and 1 is yes. The fit is for the y-intercepts of these beds, as in the “Fit to known line” option, except that the values are to be fit rather than specified. If one chooses yes, one should add a line to the parameters file for the y-intercept of each bed. These lines will go after those for dips, if expected dips are being fit as parameters as well.

2.7.7 Multi-segment restored beds

This option allows you to fit to a restored bed geometry that is not a straight line but consists of multiple line segments. If you choose this option, you will next be prompted to

Enter number of segments per bed:

For this prompt, simply enter an integer to specify the number of line segments to be used in each restored-state bed. If there is more than one bed in the model, the same number of segments will be used for all beds. If using multi-segment restored beds, you will need to add entries into the parameter file grouped by bed, in the order that the beds appear in the beds data file. For each bed, with n segments per restored bed, you will need to add entries for, in order: dips for all n segments from left to right (with dips down to the left negative and to the right positive as above), x coordinates of all n-1 bends in the restored bed, and the y coordinate of the first bend. If the model includes dip data, these will all be fit to the same restored dip, and you must include an entry in the parameters file for this dip, which will come after all the entries for all the beds.

2.8 Beds Order

If beds are one of the data types, then the next prompt is:

Are beds in order by age in the data file (as read from top to bottom)?
(0) No order
(1) Youngest to oldest
(2) Oldest to youngest

This option allows you to specify the stratigraphic order of the beds and have the program reject any models in which the restored beds are out of order. These options refer to the order of the beds in the data file, as read from top to bottom. Thus, if you choose (1), the first bed in the file must be the youngest, and if you choose (2), the first bed in the file must be the oldest, and in either case all the beds must be in the specified order. If you choose (0), then the program will not pay attention to the order of the restored beds, and they can be in any order in the data file.

If you choose (1) or (2), then you will see the prompt:

How do you want to determine order?
(1) Use order at x = 0.
(2) Prevent beds from crossing only in the domain of x coordinates spanned by both restored beds.
(3) Prevent beds from crossing anywhere within a specified domain.

If you choose (1) for this option, the bed order will be tested at x = 0 after the beds are restored. For beds that are allowed to have different dips in the restored state, it is important to be careful. If the beds are in the right order at x = 0, the model will be permitted, even if they cross and are out of order somewhere else in the cross section. Alternatively, if a bed below an angular unconformity would project above younger beds at x = 0, then the model will be rejected, even if the bed does not actually extend to x = 0. If you choose (2), then the model will be rejected if the beds cross (or are out of order) at any point within the domain of x values (in the restored state) of both beds but will not be rejected if they would cross outside that domain. If you choose option (3), you can specify the x values between which the program will check for out of order or crossing beds. If you choose option (3), you will see the prompt:

Enter minimum and maximum x coordinates of the domain to consider.

for which you should enter two values.

Warning: Specified bed order may not work properly if you are fitting data to multi-segment restored beds. In that case, you should choose “(0) No order.”

2.9 Terrace Options

If your data include marine terraces, you will next see some prompts specific to this data type. The first is:

Enter number of terraces

After this, you will see:

Enter original terrace surface dip at time of formation in degrees for all N terraces (Negative is down to the left, positive right):

where N is the number you entered in the previous prompt. This is the assumed seaward dip of the terrace surface at the time it formed and is typically small. (0° to 1° is likely, but it may be higher in some cases.)

The next prompt is:

Enter terrace inner edge elevations at time of formation for all N terraces.

For this you should enter N elevations, separated by commas, specifying the original inner edge elevations of the terraces. These will typically be the elevation of sea level at the sea level high stand at which each terrace formed.

Finally, you will be asked:

Are terraces in order by age?
(0) No order
(1) Youngest to oldest
(2) Oldest to youngest

As with beds, this option allows you to specify whether or not the terraces in the terrace data file are in order by age and, if so, whether the youngest is at the top (option 1) or the oldest is at the top (option 2). Younger beds require less fault slip to restore than older ones, and models in which this is not the case will be rejected if option 1 or 2 is chosen for this prompt.

2.10 Results to Calculate

The next prompt is to choose the type of results to calculate and save for each model.

Results To Calculate:
(1) RMS Error
(2) Probability (unnormalized) for uncorrelated data
(3) Chi-square statistic (Cardozo, 2005)
(4) Probability (unnormalized) for data correlated along a bed
(5) Uncorrelated and correlated probability along a bed (combination of 1 and 4)

This determines how the goodness of fit of each model will be evaluated. If more than one data type is being used to constrain the model and/or if one is using any of the Markov chain and similar methods (Metropolis, AM, RAM, or APT), one must choose probability, either uncorrelated (2), correlated (4), or both (5). Options 4 and 5 will only be available if beds were chosen as one of the data types.

2.10.1 RMS error

This is the commonly used root-mean-square error. It provides a straight-forward way to calculate how well a model fits to data of any given type. It cannot be chosen, however, if there are multiple data types, since a single RMS error cannot be calculated for them all together.

2.10.2 Uncorrelated Probability

With this option, each error between data and model is used to calculate a probability, assuming Gaussian errors with a standard deviation specified for each data type. All the probabilities are multiplied together, allowing a single probability to easily be calculated for multiple data types. Markov chain Monte Carlo algorithms decide whether to accept or reject a model based on a comparison of its probability with that of the current state of the chain, and therefore this option or option (4) or (5) must be used with those algorithms. This option assumes that errors in the data are not correlated with each other. The probability that is calculated and saved will not be the true probability of the model, as it is not from a normalized probability density function, but it will be proportional to the true probability.

If one chooses this option, it will be followed by additional prompts. The first is

Should data uncertainties be fit for as a model parameter? (0/1)

0 is no and 1 is yes. If no, you will soon be asked to enter the data uncertainties. If yes, the uncertainties will be model parameters, appropriate entries should be included in the parameters file, and the inversion will attempt to find the uncertainty value that best fits the data. The next prompt is

Do you want to propagate errors? (0/1):

again 0 is no and 1 is yes. If no, the error in the restored section is used to calculate a probability, with a standard deviation that is the same for all data points. If yes, the standard deviation (either specified by the user or fit for as a model parameter) is considered to be for the data in the deformed state (in which they were measured), and this error is thus propagated through the restoration for each point or dip. In general, if you specify an uncertainty based on knowledge of the deformed state, it is a good idea to propagate errors, but if you are fitting for uncertainty as a model parameter, it is probably not necessary, since you can fit for restored state uncertainty as well as deformed state uncertainty.

If bed data are included in the model, then the next prompt will be

Are uncertainties different for different beds?
(0) All bed uncertainties are the same.
(1) All beds have independent uncertainties.
(2) Group of beds have different uncertainties.

This allows you to have different uncertainty values for each bed, although within a bed, all the points must have the same uncertainty. If you enter 1, you will be prompted to

Enter number of beds:

to which you must enter the number of beds that are in the beds data file, so the program knows how many uncertainty values to expect. If you enter 2, you will see the prompt

Enter number of groups:

to which you enter the number of groups. For each group, you will then see the prompt

Enter number of first bed in group i

where i is the group number. The bed number is its order in the beds data file, starting with 1 for the first bed in the data file. All beds in a group must be together in the data file, so if i_1 is the first bed in group 1 and i_2 is the first bed in group 2, all beds from i_1 to i_2-1 will be in group 2.

If bed data are included in the model, you will next see the prompt:

Are x and y uncertainties different for bed data? (0/1)

As always, 0 is no and 1 is yes. If you choose no, then the x and y uncertainties in position will be the same and you will only need to enter one value for uncertainty in bed data. If you choose yes, then you can have different uncertainties in the x and y coordinates of the points that make up the beds.

After this, the user will see prompts to enter uncertainty for each data type being used in the model. These will look something like

Enter Uncertainty in Bed Data:

Enter Uncertainty in Dip Data:

Enter Uncertainties in Terrace Data for all N terraces:

Enter Uncertainty in all N Terrace Inner Edge Original Elevations

Enter Uncertainty in Fault Point Data:

Enter Uncertainty in Restored Bed Data:

For each of these, enter one or more numbers, which represents the one standard deviation uncertainty in the data. This value will be used to calculate the probabilities for the data (after error propagation, if applicable).

If the x and y uncertainties for bed data are different, then you will see two prompts, with the first asking for the uncertainty in the x position and the second for the uncertainty in the y position. If uncertainties are allowed to be different for each bed or for groups of beds, then for the bed uncertainties you will need to enter a number of values equal to the number of beds or groups. Otherwise, you will just need to enter one value for each prompt.

If dip data are included in the model, you will also see the prompt:

Do you want to include a separate uncertainty in the restored state dips (which will not be propagated through restoration)? (0/1)

0 is no and 1 is yes. If choosing yes, one will be prompted:

Enter Uncertainty in Restored State Dip Data:

This option allows an uncertainty in restored state dip in addition to the uncertainty in the deformed state dip (the dip measured in the field). This restored state uncertainty will not be propagated through the restoration process, while the deformed state uncertainty will. The final uncertainty in the misfit between expected and modeled restored dip will be the square root of the sum of the squares of the two uncertainties. This option is useful if one suspects that there may be local depositional dips or small-scale folding, that are not accounted for by the regional dip or the fault-related fold model.

If the model includes marine terraces, you will be prompted for uncertainties in the data points and in the "terrace inner edge original elevation", meaning the paleo-sea levels to which each terrace is to be restored. For both of these uncertainties, a number of values equal to the number of terraces must be entered, as the uncertainties can be different for each terrace.

For points on restored beds, if the x and y uncertainties in bed data are different, then you will be prompted for different x and y uncertainties for the restored bed data as well. Otherwise, you will be prompted for a single uncertainty, which will apply to both the x and y coordinates. Note that you cannot currently enter different restored point uncertainties for each bed, even if the deformed state uncertainties for each bed are different.

2.10.3 Chi-square statistic

This is the chi-square statistic used by Cardozo (2005) to evaluate models containing both bed and dip data. See that paper for further details. This option is not allowed if the data contain points on the fault. It can be used for bed or dip data alone but is intended to be used for both together.

2.10.4 Correlated Probability

This option is similar to option (2), except that the errors for bed data are correlated along the lengths of the beds. Dip and fault point data are treated the same as for option (2). This option is only available if bed data is one of the data types. Error correlation is calculated using the spherical variogram model, as described in Cardozo and Aanonsen (2009) and requires a correlation length to be specified. The distance along the bed is calculated as the distance from one data point to the next. Therefore, this option is only appropriate when there are points at short intervals along the trace of a bed, as for a bed digitized from a seismic image. It is not so useful for points separated by long distances.

This option gives the same prompts as for option (2) up to those about uncertainty in bed data, but these are followed by the question

Do you want to calculate correlation matrix based on (1) deformed state or (2) restored state bed length?

The correlation matrix is calculated using the spherical variogram model and uses the distance between points, as measured along a bed. This distance can either be calculated in the deformed state (option 1) or in the restored state (option 2). Calculating it in the deformed state is somewhat faster, since it only has to be done once, while for calculating distance in the restored state must be done separately for each model after beds are restored. In either case, the next prompt will be:

Do you want to fit for the correlation length as a model parameter? (0/1)

This is the correlation length for the spherical variogram model. It is assumed to be in whatever units all other distance measurements in your cross-section are in. If you choose 0, then the correlation length will be held fixed during the inversion, and you will be prompted to enter a value for it:

Enter correlation length for bed data:

If you choose 1 instead, the correlation length will be fit for as a model parameter, and an appropriate entry should be included in the parameters file. After this, one will see the prompts to enter uncertainty in other types of data (dips, marine terraces, fault points, restored state beds), which will be the same as for uncorrelated data.

2.10.5 Uncorrelated and Correlated Probabilities

This option is a combination of options (2) and (4). It combines a correlated error with an uncorrelated one, which may have different standard deviations. This could be used, for instance, to represent a large-scale correlated uncertainty due to depth conversion as well as a smaller-scale uncertainty in each individual data point. If you choose this option, you will get the same prompts as for option (4), and if you are using a constant uncertainty, then the first uncertainty you are asked for will be the uncorrelated one. However, after the questions about the correlation length from option (4) above, if you not fitting for uncertainty as a model parameter, you will get a prompt of the form

Enter Second (Correlated) Uncertainty in Bed Data:

This is asking for the uncertainty in the correlated part of the uncertainty model. If uncertainties are different for each bed or for groups of beds, then the prompt will ask for uncertainties for all beds or groups, and if x and y uncertainties are different, then you will get separate prompts for each. If you are fitting for uncertainties as a model parameter, then there will not be an extra prompt here, but you will have to include two entries in the parameters file, with the first corresponding to the uncorrelated uncertainty and the second to the correlated uncertainty.

Warning: If using this option with error propagation, only the first (uncorrelated) error will be propagated. This option is, therefore, best used when you are not propagating errors.

2.11 Results file name

The last prompt is

Input file name to save results to.

For this, you must enter a file name, including the file extension, for a text file in which to save the results of the run, which will include the RMS error, probability, or chi-square statistic calculated for each model that was tested. Note that for probability, the actual value that will be recorded is the natural logarithm of the probability, rather than the probability itself. For all data inversion methods except grid search, this file will also record the values of all model parameters, in the order that they occur in the parameters file. This file will be created in the folder in which you are running Setup_InvertTrishear, and it will overwrite any existing file with the same name.

3. Running the Program

To run the program, copy the InvertTrishear.exe program into a folder with the data files, parameters file, and the options file created by Setup_InvertTrishear. Then run InvertTrishear.exe. You will then see the prompt:

Choose source of run options:
(1) Input manually
(2) Read from file

Choosing (1) will allow you to answer the same questions as in Setup_InvertTrishear, after which the program will run with those options, but without saving them. Choosing (2) will allow you to enter the name of the options file you created with Setup_InvertTrishear and run with those options.

As it reads in the data files, the program will tell you how many beds, dip data, and points on the fault it has read in, so make sure these are the numbers you expect. If there is an error when reading the options file, the program will crash and close. If this happens, it may help to run InvertTrishear again choosing (1) Input manually and to enter the same options as in the options file in order to determine where the crash occurs. If the program crashes when loading one of the data files or the parameters file, check that this file is formatted correctly.

Depending on the number of models to be run, the amount of data, the total slip, and other factors, the time necessary to run the program can vary from minutes to hours. The program will periodically print out the number of models that have been run so far and will print out the total time in seconds at the end of the run.

The program uses OpenMP to parallelize some parts of the computation. It will automatically use the maximum number of processes allowed on the processor it is running on, which will typically be twice the number of cores if the processor allows hyperthreading and equal to the number of cores otherwise. This means that the program is likely to use a large percentage of your CPU. Grid search, grid Monte Carlo, and normal distribution Monte Carlo, will use just about 100% of the CPU. It is recommended that you do not run the program while also using other CPU intensive programs. The program uses very little memory, however, so you are unlikely to have to worry about that.

When the program finishes running, it will print out some information that may be useful and will depend on the algorithm chosen. To close the program, simply press Enter after it has finished running.

4. Analyzing the results

The output of InvertTrishear will be a large text file containing results. For the grid search, this will list only the output values (RMS error, probability, or chi-squared), and the position of each value in the list will correspond to its position in the multidimensional grid, from which its parameter values can be derived. A Matlab script (ReadGridtxtErrs) is provided that reads this text file into a multidimensional array corresponding to the grid. For methods other than grid search, the text file produced will contain this same output value followed by the model parameters, in the same order that parameters are specified in the parameters file. A Matlab script (ReadMCtxtErrs) is also provided to read these results, saving them in an array of size (number of models) x (number of parameters + 1). In both cases, the text file is likely to be large, so reading it into Matlab and resaving as a smaller .mat file is recommended.

The following Matlab scripts are provided to help analyze the data. See the scripts themselves for further descriptions and listings of the arguments they take. Some are directly useful for analyzing results and some are just functions called by other scripts. Good scripts to get started on analyzing results are BasicMCMCPlotting for MCMC results and TrishearPDF_grid for grid search results. IndivModel is useful for analyzing an individual model, such as a best fit model, in more detail.

BasicMCMCPlotting: Plots histograms and figures showing the path taken by the Markov chain for any of the MCMC inversion methods.

BestFit_Grid: Calculates the best fit model from grid search results and provides the subscript for that model.

ContourMCMC: Produces a contour plot of a 2D histogram of MCMC results over two different parameters.

ind2subMat: Converts index of a multidimensional matrix in Matlab into subscripts stored in a single matrix.

IndivModel: Allows the user to analyze an individual model. Note that not all types of models have been thoroughly tested and that the listric fault and parallel fault propagation fold models are not included.

InputGoodOnly: Asks for input from the user and rejects any values not in a list of possible inputs.

MCMC_plot_general: Plots histograms of results from any of the MCMC methods as subplots within a single plot window.

parallel_FPF_func: Function for parallel-fault propagation folding, for points.

parallel_FPF_func_dips: Function for parallel-fault propagation folding, for dip data.

Params_from_Errs_Grid: Turns the grid search results array or a subset of it into a list of parameter values more like the output for other methods.

QuadIntegrate: Integrates a grid search probability density function across a chosen dimension, using Simpson quadrature.

ReadBeds: Reads a bed data file into Matlab.

ReadGridtxtErrs: Reads the output text file produced by InvertTrishear for a grid search.

ReadMCtxtErrs: Reads the output text file produced by InvertTrishear for all Monte Carlo or Markov chain Monte Carlo methods (anything except grid search).

trishear_func: Trishear function for points, for a straight fault.

trishear_func_bend: Trishear function for points, for a fault with a bend in it.

trishear_func_decol: Trishear function for points, for a fault with a ramp from a horizontal detachment.

trishear_func_dip: Trishear function for dips, for a straight fault.

trishear_func_dip_bend: Trishear function for dips, for a fault with a bend in it.

trishear_func_dip_decol: Trishear function for dips, for a fault with a ramp from a horizontal detachment.

trishear_func_dip_multi_bend: Trishear function for dips, for a fault with multiple bends in it.

TrishearPDF_grid: Calculates a probability density function for results of a grid search.

TrishearPDF2D_grid: Calculates a 2D probability density function for results of a grid search.

xy_to_ze: Converts from the Cartesian coordinate system of the cross section to the trishear coordinate system.

ze_to_xy: Converts from the trishear coordinate system to the Cartesian coordinate system of the cross section.

References

Allmendinger, R.W., 1998. Inverse and forward numerical modeling of trishear fault-propagation folds. Tectonics 17, 640-656.

Cardozo, N., 2005. Trishear modeling of fold bedding data along a topographic profile. Journal of Structural Geology 27, 495-502.

Cardozo N., Aanonsen, S., 2009. Optimized trishear inverse modeling. Journal of Structural Geology 31, 546-560.

Cardozo, N. and Brandenburg, J.P., 2014, Kinematic modeling of folding above listric propagating thrusts: Journal of Structural Geology, v. 60, p. 1-12.

Erslev, E.A., 1986, Basement balancing of Rocky Mountain foreland uplifts: Geology, v.14, p. 259-262.

Haario, H., Saksman, E., Tamminen J., 2001. An adaptive Metropolis algorithm. Bernoulli 7, 223-242.

Hardy, S., Ford, M., 1997. Numerical modeling of trishear fault propagation folding. Tectonics 16, 841-854.

Oakley D.O.S. and Fisher, D.M, 2015, Inverse trishear modeling of bedding dip data using Markov chain Monte Carlo methods, Journal of Structural Geology, v. 80, p. 157-172.

Oakley, D.O.S., 2017. Fault-Propagation Fold Kinematics and Deformation Rates in the North Canterbury Fold and Thrust Belt, South Island, New Zealand. Ph.D. Dissertation. The Pennsylvania State University.

Oakley, D.O.S., Fisher D.M., Gardner, T.W., and Stewart, M.K., 2018. Uplift rates of marine terraces as a constraint on fault-propagation fold kinematics: Examples from the Hawkswood and Kate anticlines, North Canterbury, New Zealand. Tectonophysics 724-725, 195-219.

Seeber, L. and Sorlien, C.C., 2000, Listric thrusts in the western Transverse Ranges, California: GSA Bulletin, v. 112, n. 7, p. 1067-1079.

Suppe, J., 1983. Geometry and kinematics of fault-bend folding. American Journal of Science 283, 684-721.

Suppe, J., and Medwedeff, D.A., 1990, Geometry and kinematics of fault-propagation folding: Eclogae Geologicae Helvetiae, v. 83, 409-454.

Vihola, M., 2012. Robust adaptive Metropolis algorithm with coerced acceptance rate. Statistics and Computing 22, 997-1008.

White, N.J., Jackson, J.A., and McKenzie, D.P., 1986. The relationship between the geometry of normal faults and that of the sedimentary layers in their hanging walls. Journal of Structural Geology, v. 8, n. 8, p. 897-909.

Zehnder, A.T., Allmendinger, R.W., 2000. Velocity field for the trishear model. Journal of Structural Geology 22, 1009-1014.

Ziesch, J., Tanner, D.C., and Krawczyk, C.M., 2014. Strain associated with the fault-parallel flow algorithm during kinematic fault displacement. Mathematical Geosciences, v. 46, p. 59-73.