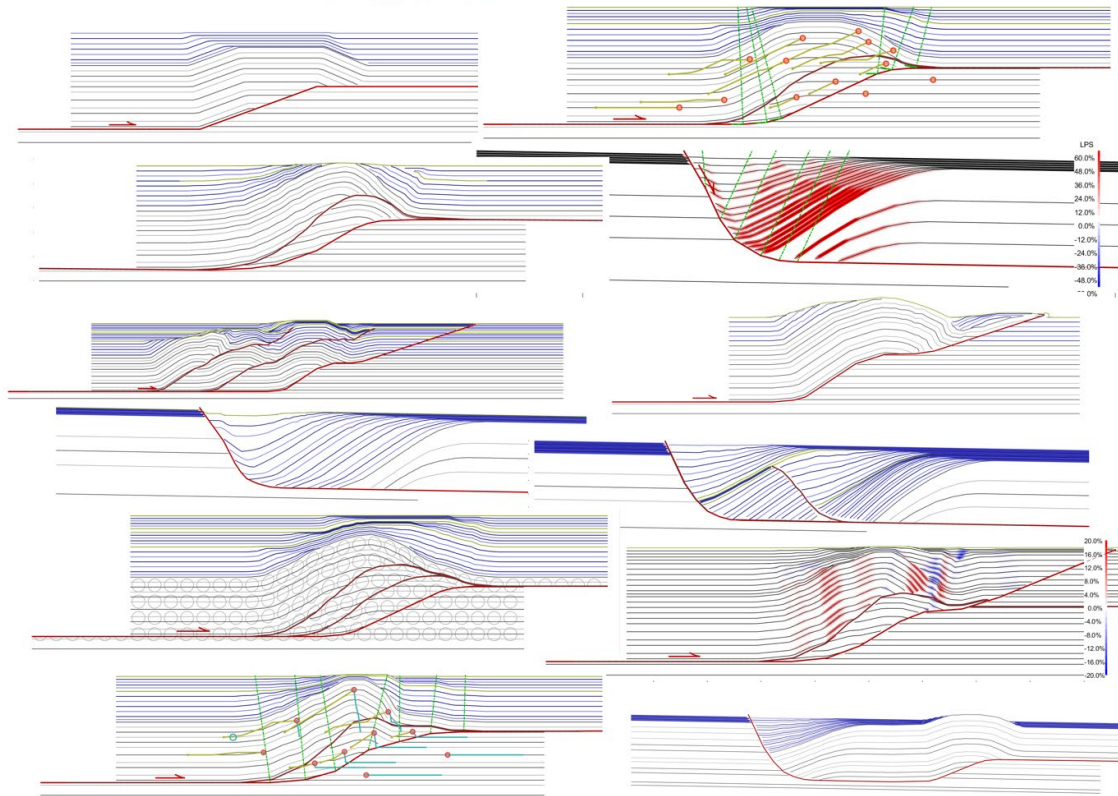


fbfFor Manual, v 1.20
Chris Connors, connorsc@wlu.edu
Jan, 2021

fbfFor – a forward kinematic modeling program for fault-bend folding

Chris Connors, Copyright (c) 2002-2021



Contents

| | |
|--|----|
| Installation..... | 3 |
| Background..... | 4 |
| General description of use | 5 |
| File Menu..... | 7 |
| Prepare Model Menu..... | 9 |
| Slip Panel | 11 |
| Velocity Boundary Panel..... | 12 |
| Layers Panel..... | 13 |
| Fault Panel | 15 |
| Icon Bar..... | 16 |
| Fault digitizing and editing | 16 |
| Display Options Menu | 18 |
| How To's | 21 |
| Model Constraints..... | 27 |
| Numerical Solution Implications | 28 |
| Notes of Caution and Guidance for Best Practices | 28 |
| Known Bugs..... | 32 |
| Potential Future Improvements..... | 33 |
| Acknowledgements..... | 33 |
| References..... | 33 |
| Copyright Notices..... | 34 |

Installation

The program fbffor is written in MATLAB. It should run on Windows 7 through 10, 64bit. A version for macOS is coming soon. The program only comes as a compiled stand-alone application. Source code is not being provided. You do not need to have a license for MATLAB to run fbffor, but the application requires that 9.9 (R2020b) of the MATLAB Runtime is installed. If you do not have that version of MATLAB Runtime installed, then during the installation you will be prompted to download this library from [mathworks.com](https://www.mathworks.com) as part of the installation process (this does not require a MATLAB license). After installation you should be able to double click the desktop application fbffor and it should run.

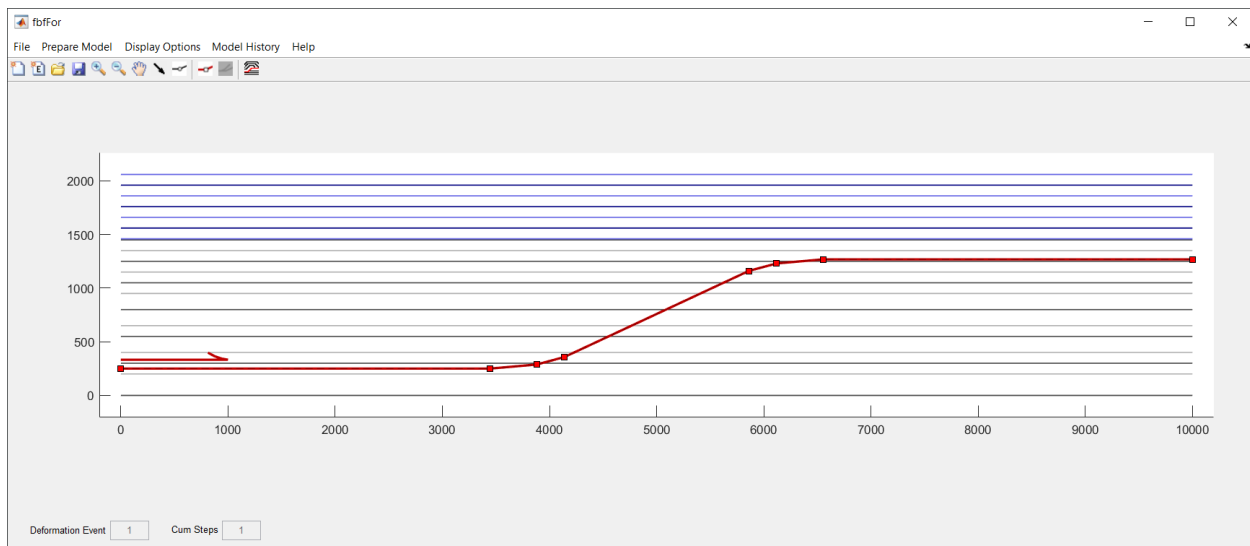
Background

fbfFor is a kinematic forward modeling program for fault-bend folding in contraction and extension. The modeling approach can handle multiple fault bends of different geometries (e. g. fault bends not stepping up from a detachment), imbricates, and variable velocity-boundary orientations with corresponding varying slip ratios. Inclined shear and flexural slip have been unified into one algorithm. fbfFor starts with input layers (polylines), slip and fault parameters, and then for successive timesteps calculates the deformation. Growth strata are modeled as layers that have an age, and come in at a given timestep. When modeling contraction, at each timestep the cutoff and fault-bend angles are determined, and the fault-bend fold equations (Suppe, 1983) are solved for the slip ratios and velocity-boundary orientations. The folds produced conserve cross-sectional area, and in general layer thickness, and line length. In cases where there is no solution to the fault-bend equations, such as large anticlinal bends or interfering kink bands, changes in bed length and layer thickness are computed, but cross-sectional area is still conserved. If compaction is incorporated then implied mass is conserved, but not cross-sectional area. The thrust sheets produced are dominantly parallel folds with localized bed thinning or thickening. Rounded-hinge, parallel folds are easy to produce, as well as more angular, stylized structures. When modeling extension the velocity boundary orientations are independently defined from fault shape, and the fault-bend folding equations are used to calculate slip ratios. The modeled structures conserve area, resulting in a solution equivalent to the inclined-shear approach, such as that described in Xiao and Suppe (1992). Extensional fault-bend folds, such as rollover structures with growth, are easily produced. Strain, particle tracking, and other attributes of any model over time can be displayed.

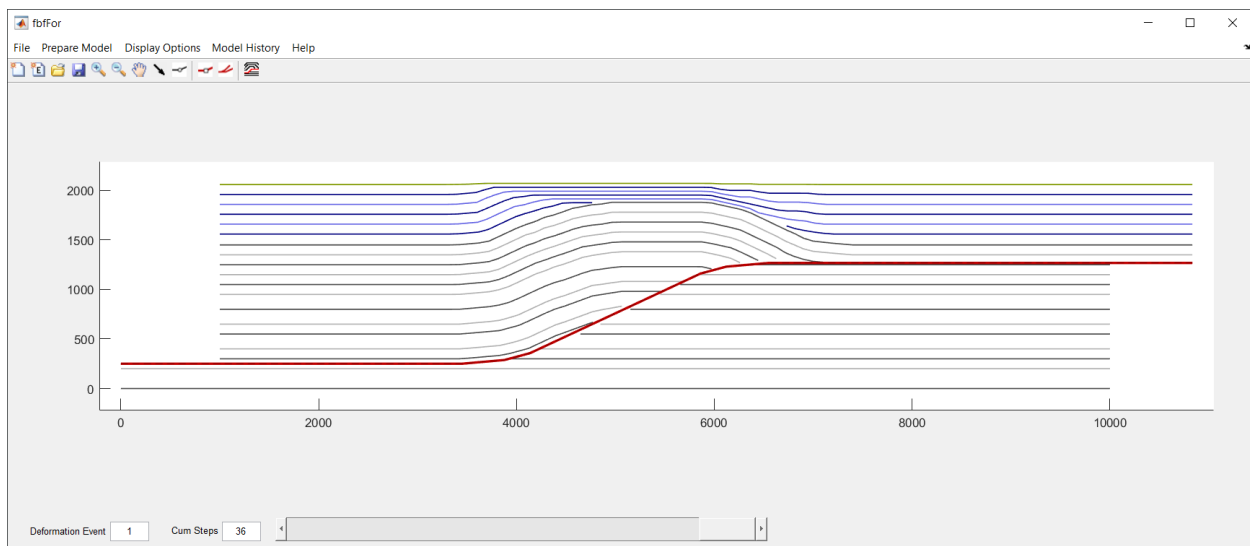
The theory behind the program is detailed in Connors et al. (2021). If you use fbfFor for research that you publish, please cite that article. If you find a bug, please email connorsc@wlu.edu. If you have a desire for new features, likewise feel free to email, and at some point, they may be incorporated, but the program is not intended to be a replacement for more feature-rich, commercial structural modeling packages. fbfFor is a simple program that we use in our research and teaching, and we are sharing it for free with others with the idea that some structural geologists may find it useful. We provide it “as is,” with no implied warranties. See the copyright notice at the end of this manual or the About Menu in fbfFor for more details.

General description of use

When the program is invoked, the default figure is presented with initial layers and a fault, ready to be edited.

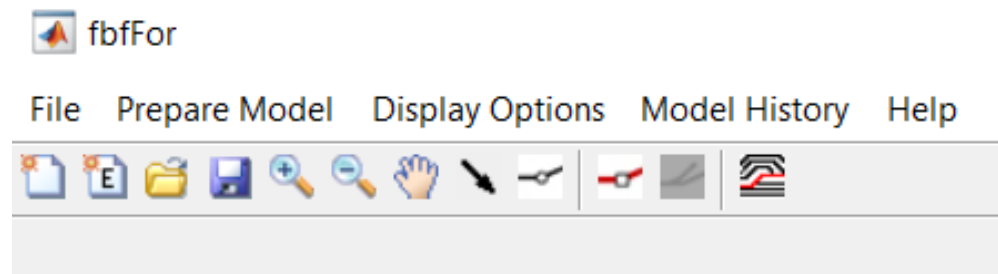


One can run a model immediately by choosing **Prepare Model Menu → Deform Layers**, or typing *Control-D*. Note that once a model is computed a slider appears at the bottom of the figure allowing one to change the timestep interactively, as well as the deformation event. A “timestep” is a discrete increment of slip with corresponding fault-bend fold angles, and velocity boundaries computed for a given fault shape. A “deformation event” refers to a collection of these timesteps for one fault. Making an imbrication (new fault) is equivalent to a new deformation event, but an event could also simply be a different slip rate, or percent subsidence with the same fault shape. These options are explained in more detail later in this manual.



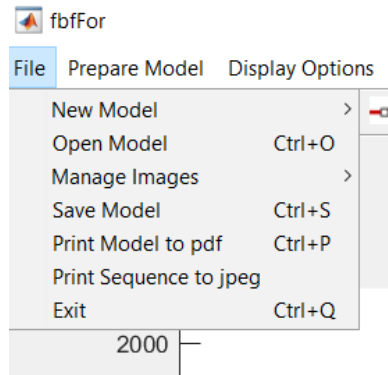
Normally one would like to alter a default fault shape by using the Menus and Toolbar as described below.

The menus are set up from left to right in the general workflow of building a model. So, one would in general first access the ***File Menu***, then the ***Prepare Model Menu***, then the ***Display Options Menu***. In addition to menus the ***Icon Bar*** is designed for completing common tasks such as making and modifying faults and running a model. As with the menus, workflow is generally left to right. In the subsequent sections we describe each of these menus and icons. Near the end of the manual there is a section on [Notes of Caution and Guidance for Best Practices](#) that we encourage the user to read after reading about general use of fbFor.



File Menu

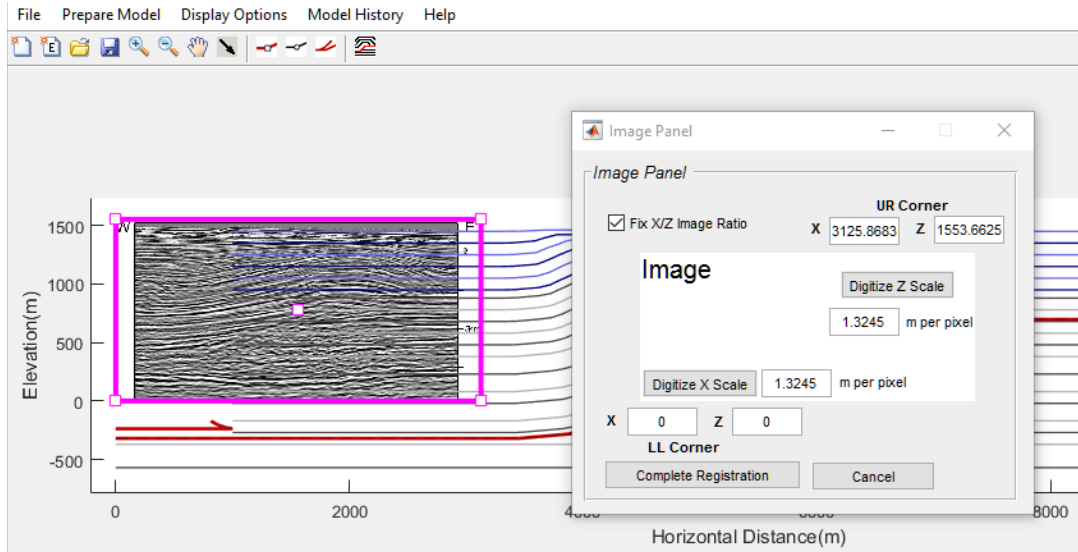
The **File Menu** allows for creating a new model, opening a pre-existing model, loading a jpeg or tiff image as a backdrop, printing a model, etc.



Save Model saves the geometry and deformation history of the model as a .mat file that can be later reread by the program by using the **Open Model** command. The file format is really only meant to be used internally by the program, and at the present time the internal file format will not be described. While the .mat file contains a Matlab structure that may be interrogated within Matlab, note that the program has specific expectations for the contents of the .mat file, so modifying these in any way may render the file unreadable by fbfFor.

Our approach with fbfFor is that it is designed to be used by a geologist who needs to know nothing about Matlab and just wants to make balanced, sequential, forward models of fault-bend folds. So, save a model if you like the result and want to come back to it, or to make a template stratigraphy and/or fault that you want to access in the future. A word of caution here. The models contain many different data structures to make plotting and further calculations easier. So, they can get large, if lots of points, lots of layers, and most importantly lots of imbricates are made.

One particular menu item relevant to users who want to model real structures is **Manage Images** → **Load Image**. If one chooses a .tif, .jpg, or .png image then the **Image Panel** is displayed along with the image ready for resizing graphically by grabbing the pink rectangle or entering the values in the panel.



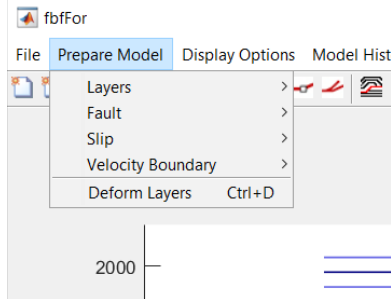
If you want to maintain the vertical/horizontal scaling ratio of your image ensure that the values for X scale and Z scale in the entry boxes for each are equal, and that the Fix X/Z Image Ratio checkbox is selected. Alternatively, if you want to rescale your image to correct for vertical exaggeration, this is where you would enter different values in order to do that.

The **Print Model to pdf** prints the current model at the deformation event, and timestep, with annotations visible on screen and saves it to a PDF. The **Print Sequence to jpeg** produces a sequence of jpeg images saved to disk of the evolution of the model at the increment you choose.

Prepare Model Menu

The ***Prepare Model Menu*** is where input parameters for the Layers, Fault, Slip, and Velocity Boundary are accessed through the ***Layers, Fault, Slip, and Velocity Boundary Panels***.

Generating a model (Deform Layers) can be run from this menu also.



Each Panel should be self-explanatory, but are also explained below. Some fields are only possible, and thus active or populated, at certain times. Numerical values can be entered or modified in many of the fields of the panels. Note after you click on the field with the value and edit the number, you have to hit “Enter” for your change to be recorded. Smart defaults for many values are implemented. If obviously impermissible values are entered such as negative slip, or faults that double back on themselves these are not accepted. It is certainly possible however to still create unrealistic geometries, such as corrugated faults that have repeated negative and positive slopes. We do not recommend making these fault shapes, and it is possible that the algorithms will not work right, including potentially not conserving cross-sectional area.

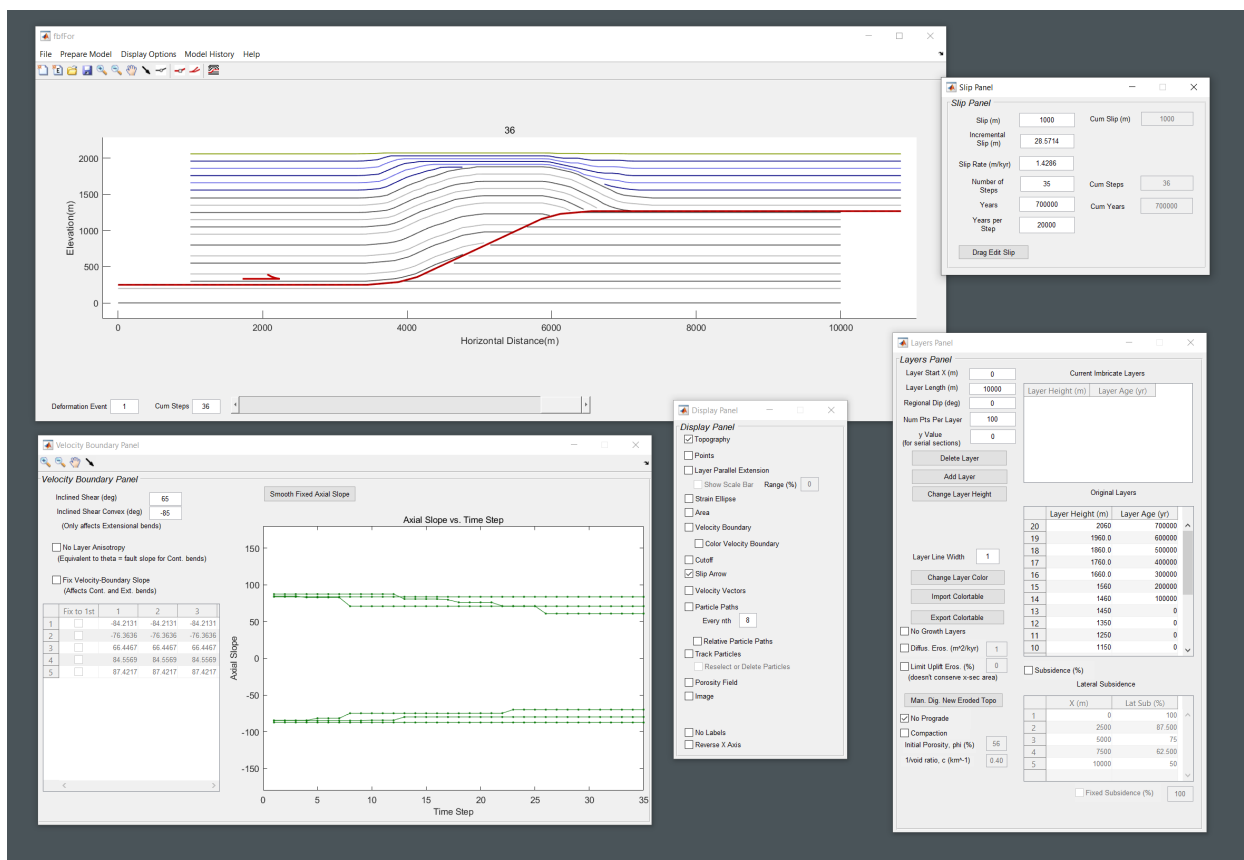
Each of these Panels have editable boxes that are tied to other boxes. So, for example if you change Slip it will change Slip Rate, or vice versa.

If a button is pressed then input into the plot window is expected.

So, for example, if Change Layer Height in the ***Layers Panel*** is pressed then the user is expected to left click and drag a layer to another position.

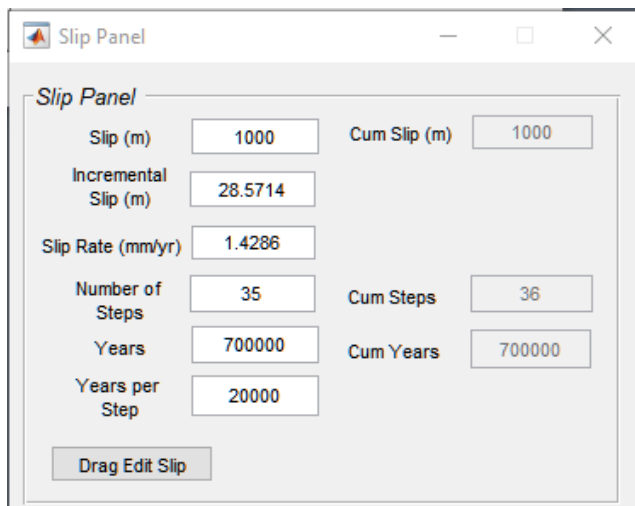
If Man. Dig. New Eroded Topo in the ***Layers Panel*** is pressed then you should use the left button to click the new topography you want and end with the right mouse button.

If Drag Edit Slip in the ***Slip Panel*** is pressed then you should use the left button and drag the arrow tip to the new amount of input slip you want.



Slip Panel

The ***Slip Panel*** allows for the magnitude of input slip for a deformation event, the length of time over which the slip occurs, as well as how this is discretized over a series of timesteps. A deformation event is composed of all timesteps, plus the original undeformed ($t = 0$) step. Like all the panels, some boxes will not allow unacceptable entrees. For example, Slip must be positive, as does Number of Years. Similar to the editing faults, you can only edit the Slip values for the current deformation event (imbricate). The Cumulative Slip and the like are calculated based on previous imbricate values. The Years has implications for when you have a growth layer appear. The definition of this is done through the ***Layers Panel***.

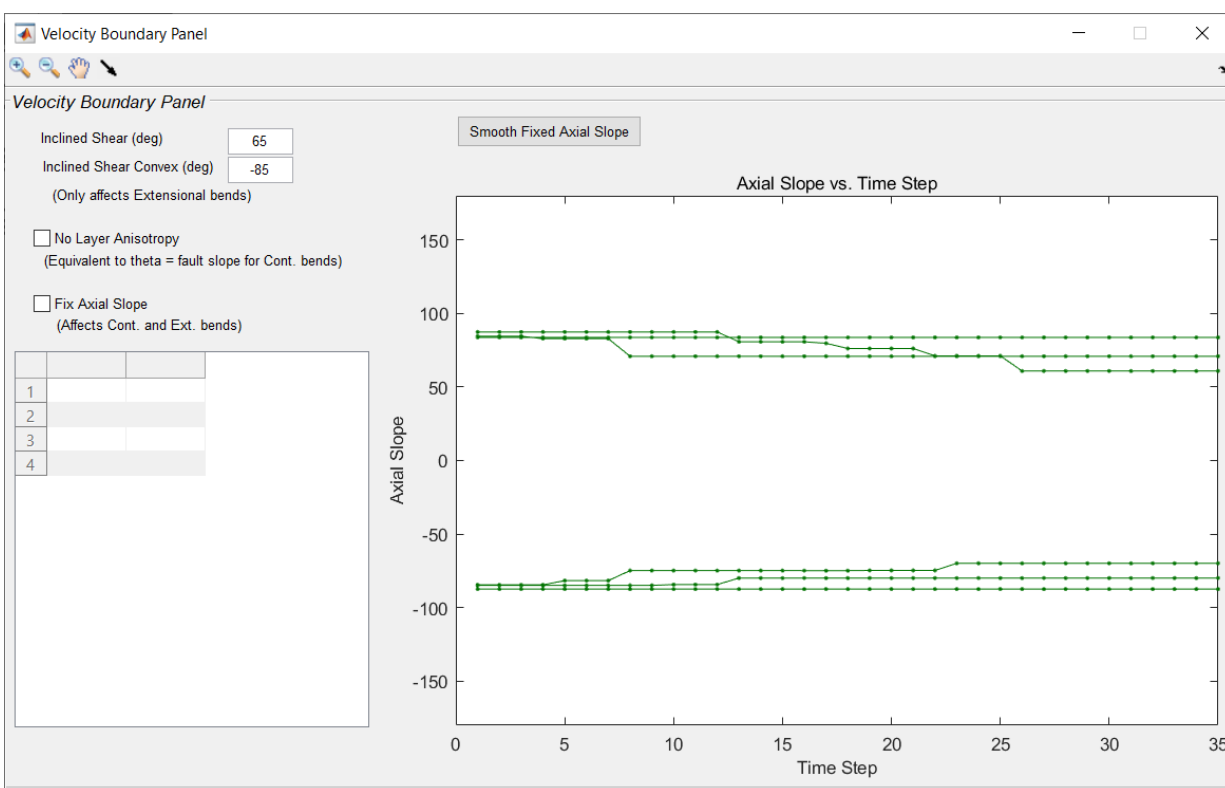


The image shows a software window titled "Slip Panel" with a standard Windows-style title bar (minimize, maximize, close buttons). The panel contains several input fields and calculated values arranged in two columns. At the bottom left is a button labeled "Drag Edit Slip".

| Parameter | Value | Parameter | Value |
|----------------------|---------|--------------|--------|
| Slip (m) | 1000 | Cum Slip (m) | 1000 |
| Incremental Slip (m) | 28.5714 | | |
| Slip Rate (mm/yr) | 1.4286 | | |
| Number of Steps | 35 | Cum Steps | 36 |
| Years | 700000 | Cum Years | 700000 |
| Years per Step | 20000 | | |

Velocity Boundary Panel

The **Velocity Boundary Panel** allows for changing the inclined shear angle, which might be particularly useful when modeling normal faults. These must be between 0 and 90 deg., for antithetic shear (concave bends), and 0 and -90 deg. for synthetic (convex bends). In order to maintain an extensional sense of shear to the beds the orientation is determined based on the fault-bend geometry. For contractional bends the velocity boundaries are automatically determined by `fbfFor` from the fault-bend fold equations in order to have no layer-parallel strain (LPS) or to minimize the strain. In this panel you can edit the orientation of any velocity boundary, including those associated with contractional bends, spatially and temporally to create variable, inclined shear. This will almost always result in LPS and so should be used sparingly for contractional structures.



layer height is lower than that of the whole topography. This is not by itself a problem for fbFor, but there are times when fbFor can get confused by this with multiple such layers. In general, do not make younger layers lower than older layers unless you really know you want to do this.

Subsidence is modeled as a percentage of the growth layer heights, by clicking the Subsidence % checkbox. 100% subsidence means no base level change, causing each growth layer, and all older layers, to subside as much as the thickness of that layer. The table of layer height values displays the non-subsided layer heights. That is, if you were inputting the values of how thick a layer was above an adjacent layer, then these would be the total layer heights in an external reference frame before deformation occurred. Selecting the *Fixed Subsidence (%)* checkbox applies a laterally uniform subsidence as a percentage of the thickness of the growth section, which is the default. Laterally variable subsidence is otherwise implemented if the *Fixed Subsidence* checkbox is unchecked, and individual percentages can be input. Lateral subsidence is a percentage relative to growth layer height such that rudimentary changes to the dip of layers occurs over time, independent of any regional dip applied.

Erosion is primitive at this point. The Diffus. Erosion checkbox will cause simple diffusion-based smoothing of the topographic surface, serving to redistribute material from regions of high topography to adjacent low areas, after the methodology described in Pelletier (2008). Note that the rate specified here is tied to the same timescale for growth sedimentation and total model time and slip rate specified in the *Slip Panel*. The diffusion constant can be varied. Cross-sectional area of the hanging wall should be preserved if diffusion-based erosion occurs without growth sedimentation.

Within a timestep both erosion and deposition of a layer occurs. The Man. Dig. New Eroded Topo button is a way of “cutting out” unwanted topography at the end of a deformation event. This can be used to say create an unconformity before having deposition of new layers in the next deformation event.

A general compaction algorithm has been implemented based on the empirical compaction curves in (Sclater and Christie, 1980) and Allen and Allen (2013). These are parameterized as an intercept (initial phi) and a negative exponential term (1/void ratio). These terms vary with lithology, so there is an implied mass. Only one choice for all layers can be made at this time. In this approach, compaction is calculated based on burial depth below the topographic surface, with the additional feature that burial history is tracked, so that, for example material that is buried, then advected to the surface, is not subsequently “re-compacted.” The subsidence generated by compaction is independent of any imposed subsidence percentage. Obviously when compacting cross-sectional area is no longer conserved, but implied mass is still conserved.

Fault Panel

The **Fault Panel** allows one to specifically type in the values of the fault points, but generally you will want to just graphically grab the vertices and move them as discussed below. You might want to tweak the fault slopes by entering their values here. Like all other boxes in the panels, changing fault slopes changes vertices and vice versa.

Fault Panel

| | X | Y | Z |
|---|------------|---|------------|
| 1 | -1.0156 | 0 | -120.0188 |
| 2 | 3.4476e+03 | 0 | -56.2190 |
| 3 | 3.8787e+03 | 0 | -10.5297 |
| 4 | 4.1373e+03 | 0 | 63.5594 |
| 5 | 5.8617e+03 | 0 | 899.5183 |
| 6 | 6.1203e+03 | 0 | 973.6074 |
| 7 | 6.5514e+03 | 0 | 1.0193e+03 |
| 8 | 1.0610e+04 | 0 | 1.0944e+03 |

| | Slope (deg) |
|---|-------------|
| 1 | 1.0599 |
| 2 | 6.0501 |
| 3 | 15.9843 |
| 4 | 25.8645 |
| 5 | 15.9843 |
| 6 | 6.0501 |
| 7 | 1.0599 |

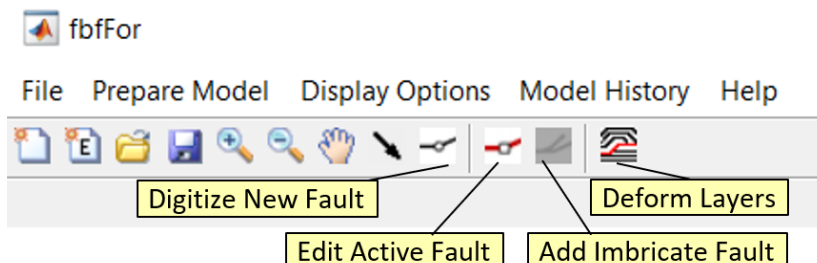
Buttons: Digitize New Fault, Edit Active Fault, Smooth Active Fault, Add Imbricate, Remove Last Imbricate, Export Fault, Import Fault, Reactivate Last Fault (experimental) [checkbox], Fault Line Width: 2, Don't reset to 1st time step when editing fault [checkbox]

By default, all previous faults during imbrication are passive features that deform like the layers do. If the Reactivate Fault box is checked, during contraction the second fault will cause the first fault to deform, and slip will be induced on the first fault in order to better maintain layer thickness (see Connors et al., 2021). This is available for the first, and at this point, only the first imbricate. Even for the first imbricate, it sometimes doesn't work perfectly. There are numerical instabilities that can result, and cases where it is impossible to reactivate, such as when the previous fault is cut by the new fault. If before deformation starts fbfFor can detect that the older fault is cut by the new fault, then it will turn reactivate off automatically. Also, you cannot edit velocity boundaries in the **Velocity Panel** when a fault is reactivated. Consider Reactivate Fault a feature that is quite experimental.

Icon Bar

In addition to menus the *Icon Bar* is designed for completing common tasks such as making and modifying faults and running a model. As with the menus, workflow is generally left to right.

Some items are only possible and thus active when viewing certain timesteps.



Fault digitizing and editing

When editing a fault graphically the interface is similar to any graphics program. For example:

- Clicking a fault point selects it. The selected point turns white.
- Dragging a fault point changes its position.
- Holding down the *Shift Key* and then clicking allows for selecting multiple points.
- Holding down the *Control Key* and then clicking adds a fault point.
- Hitting the *Backspace* or *Delete key* removes any selected points.

If digitizing a new fault, use the *Left Mouse Button* to define points and the *Right Mouse Button* to end the digitization. This erases the previous fault and makes the digitized fault ready for further editing. **To make a thrust start digitization to the left of the layers and digitize points. To make a normal fault start digitization above the pregrowth layers.** fbfFor will truncate the fault to the limits of the layers at any timestep including during this digitization step.

The reason for starting faults in this way is that fbfFor chooses whether fault bends are contractional or extensional based on the slopes of the fault segments. Because input slip can only be towards the right and is parallel to the first fault segment, positive slopes are in general contractional (accommodate shortening), and negative slopes extensional (accommodate extension).

We default to contractional slopes as any that are ≥ -3 deg. The reason for this somewhat arbitrary, non-zero distinction is that some detachments may subtly cut down section and we have found this alleviates imposing inclined shear related to undulating, or gently negatively dipping detachments when it is not necessary, or desirable. This can be overridden by defining the velocity boundary separately in the *Velocity Panel*.

All of the above apply only to the current imbricate fault, not any previous faults. If you want to edit those you have to remove the imbricate from the *Fault Panel*. You might want export the

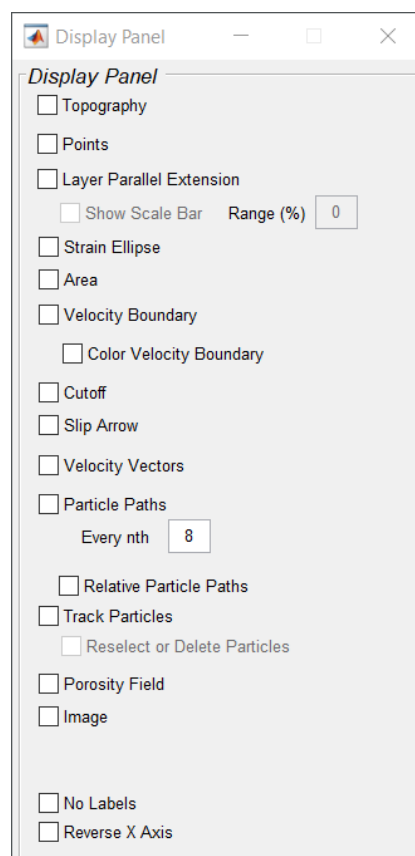
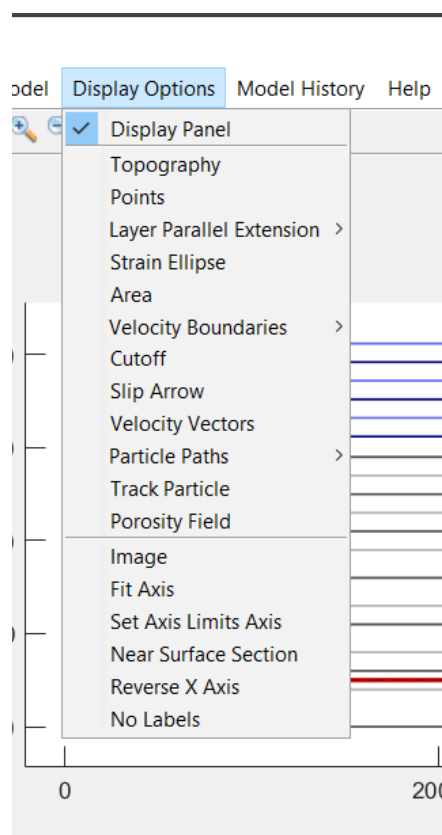
existing fault, and later import it, if you know you will want to use that fault again for a later imbricate.

Whenever you are editing a fault, you are editing the fault for the first timestep of that imbricate's run (before deformation, subsidence, etc. for movement along that fault). The fault will subside after that. This can be confusing if you are trying to model a structure that you have imported as an image. So, in those cases make sure subsidence is turned off.

If you really like a fault and want to use it again, use the Export Fault and Import Fault buttons. Some day I might make it easier to have default faults of one's choosing, but for now, that is what you have to do.

Display Options Menu

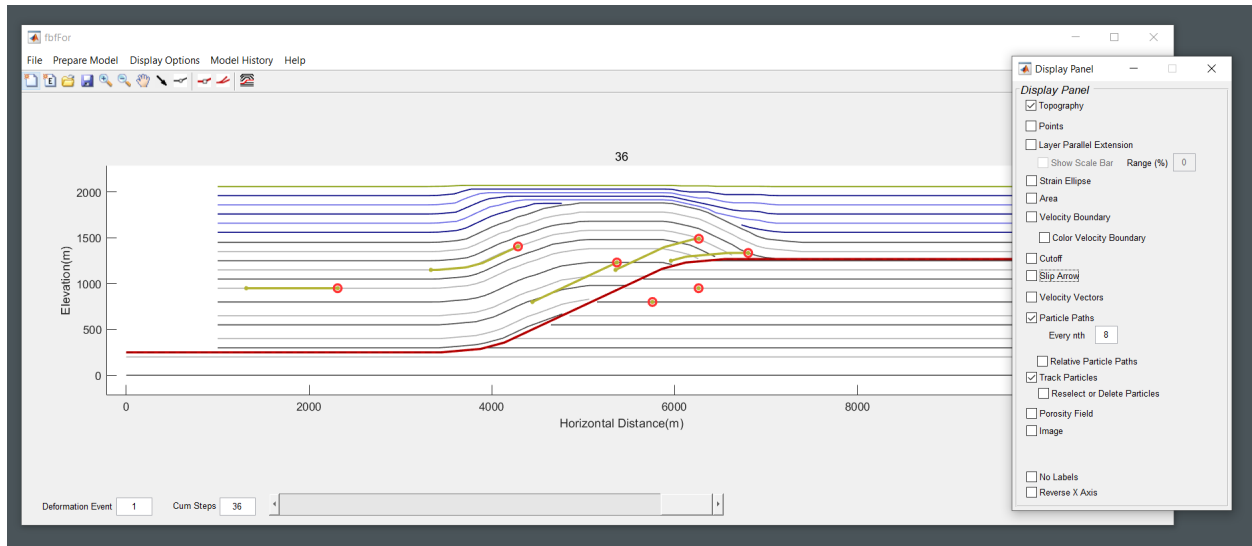
The **Display Options Menu** allows for a model to be visualized in different ways, some are only relevant, and thus active, after a model has been run. Some require additional computation which is done automatically when the particular choice is made. These are mostly self-explanatory.



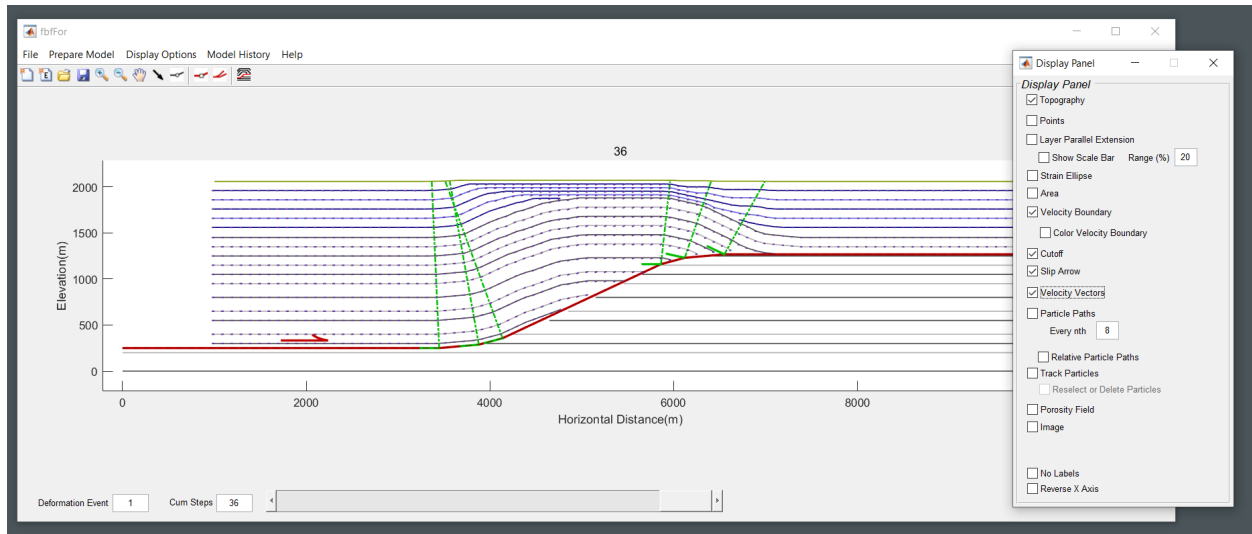
Choosing the **Display Panel** brings up a panel that allows one to choose most of the same options as in the **Display Options Menu**. Note these are coupled so choosing an option in one is equivalent to choosing the option in the other.

Particle Paths will show the motions of points in a model. The default is to sample every 8th point on a layer. If you have chosen to track particles (whether they are displayed or not) then you are choosing to show those paths only. In order to get back to every nth sampling, one has to delete the tracked particles in the **Display Panel**.

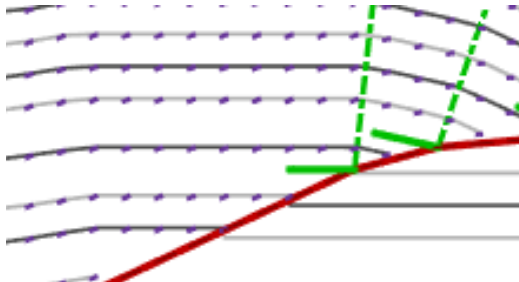
For example, here are six *Tracked Particles Velocity Boundary* and their *Particle Paths* relative to the external reference frame.



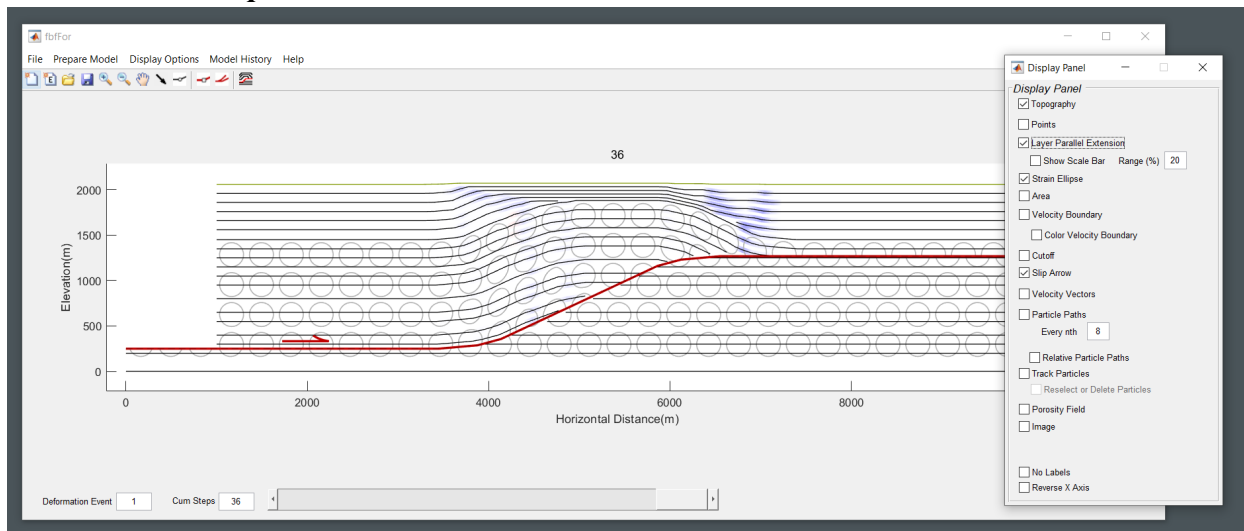
Choosing *Slip Arrow*, *Velocity Boundary*, *Cutoff*, and *Velocity Vectors* shows the following:



Note that the velocity vectors are for a given timestep, so you need to zoom in to see the real slip amount for that step.

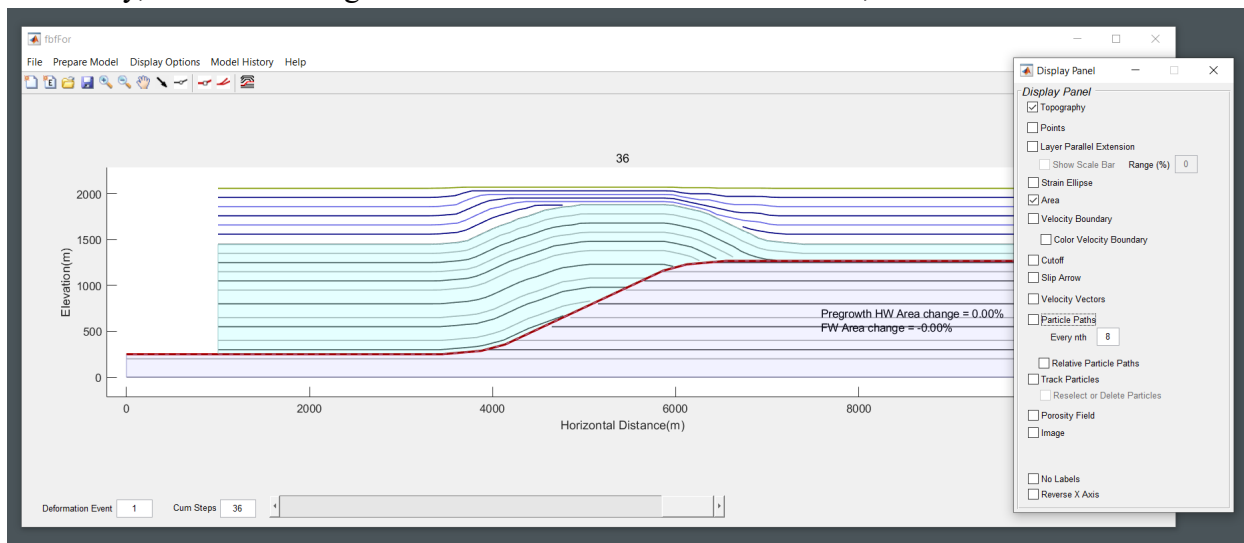


When certain items are chosen additional computation needs to be done, such as *Layer Parallel Extension* and *Ellipses*.



The layer-parallel extension scale defaults to 20% range. To determine the full range of strain, choose the menu item for *Layer Parallel Extension* → *Define Extension Range* and it will default to the full extension. Note, this might only be for one segment and so may be spurious.

An important display item is to check for area balance. Area should be conserved in the pregrowth hanging wall and footwall, unless erosion or compaction has been modeled, but fbffor uses a numerical approach, and there could be limits in the ability of fbffor to calculate the velocities. So, changes of $< |0.5\%|$ should be considered area balanced. If there is a larger change than that, it is likely that model is too weird (such as corrugated faults) to model accurately, or there is a bug in the code. To check for area balance, turn on the *Area* box:



The **Model History Menu** allows one to cache a model before running a different model, and likewise recall a previously cached model. At this point for reloading a model when later invoking the fbffor, you can only save the currently visible model by **File Menu → Save**.

The **Help Menu** should call up this document if fbffor can find the location of the file.

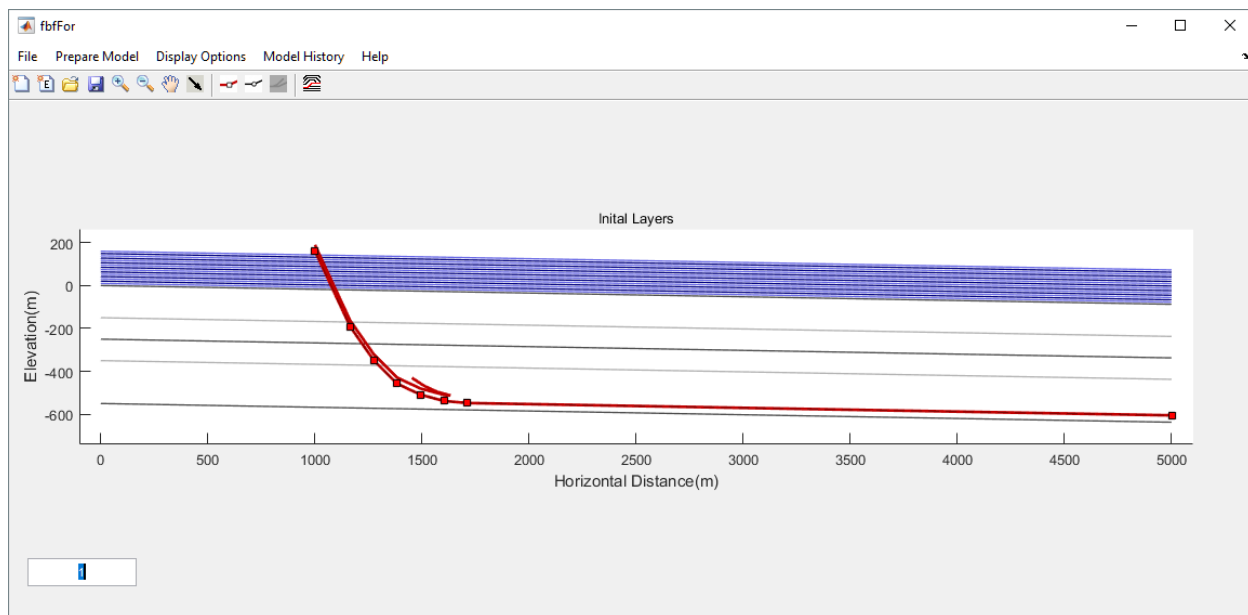
How To's

Below is a simple FAQ on how to do specific things with fbffor. Read the previous sections for general use of the program and how the different panels allow one to build and change models.

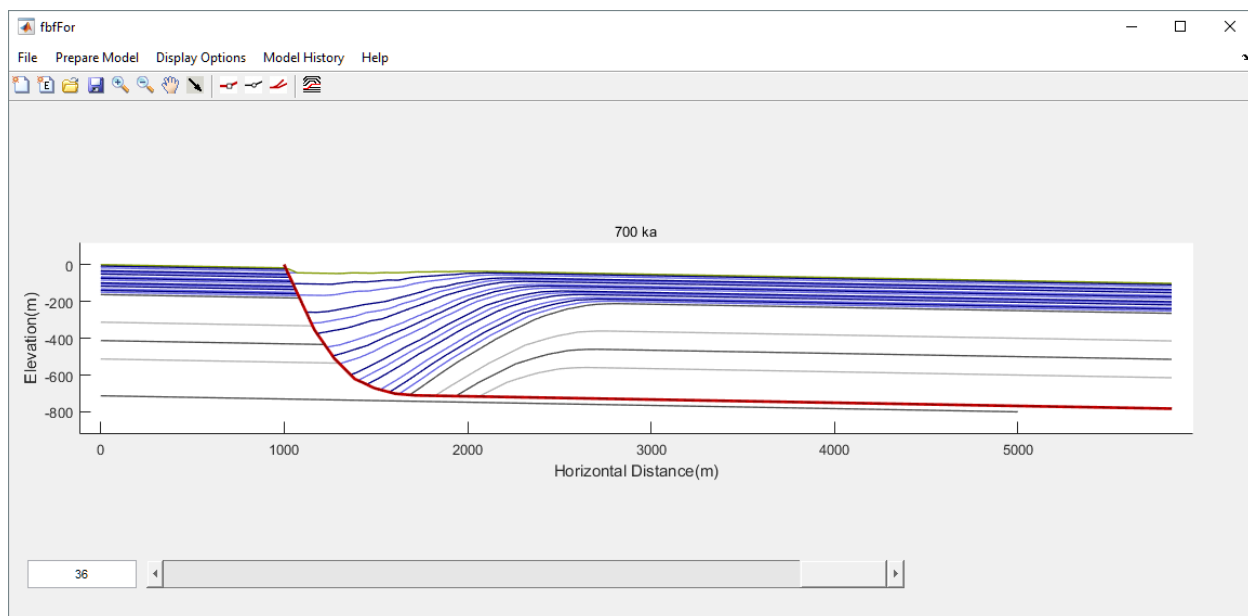
1. How do I bring in seismic or a sketch of the structure I want to model?
Choose **File Menu → Manage Images → Load Image**. See the description in the previous section. Note if you want to model an image, turn subsidence off in the **Layers Panel**.
2. How do I set up default stratigraphy, fault or input slip?
Use the Layer, Slip and Fault Panels to make your own defaults and choose **File Menu → Save Model** or **Control-S**. Any future time you want to run a model, choose **File Menu → Open Model** or **Control-O** and load the model you saved previously. Related choices are the Export Fault and Import Fault buttons in the **Fault Panel** and the Export Color Table and Import Color Table buttons in the **Layers Panel**.
3. How do I incorporate erosion, subsidence and compaction?
Use the **Layers Panel**. See the description in the previous section.
4. How do I make a movie of a model?
There is currently no direct way to do this, but if you choose **Print Sequence to jpg** you can generate several images at different timesteps. You can use your own method of combining these into a movie.
5. How do I determine if a fault bend is contractional or extensional?
This is determined automatically based on the fault segment slope and shape. Generally, this is obvious. That is, is the hanging wall moving up or down. See the section on fault editing for more details, and likewise where to start digitizing a fault (left of the model layers for contraction and above the model layers for extension).

6. How do I make a rollover structure?

File Menu → New Model → New Extensional Model or click the appropriate icon on the **Icon Bar**.



Then deform layers.

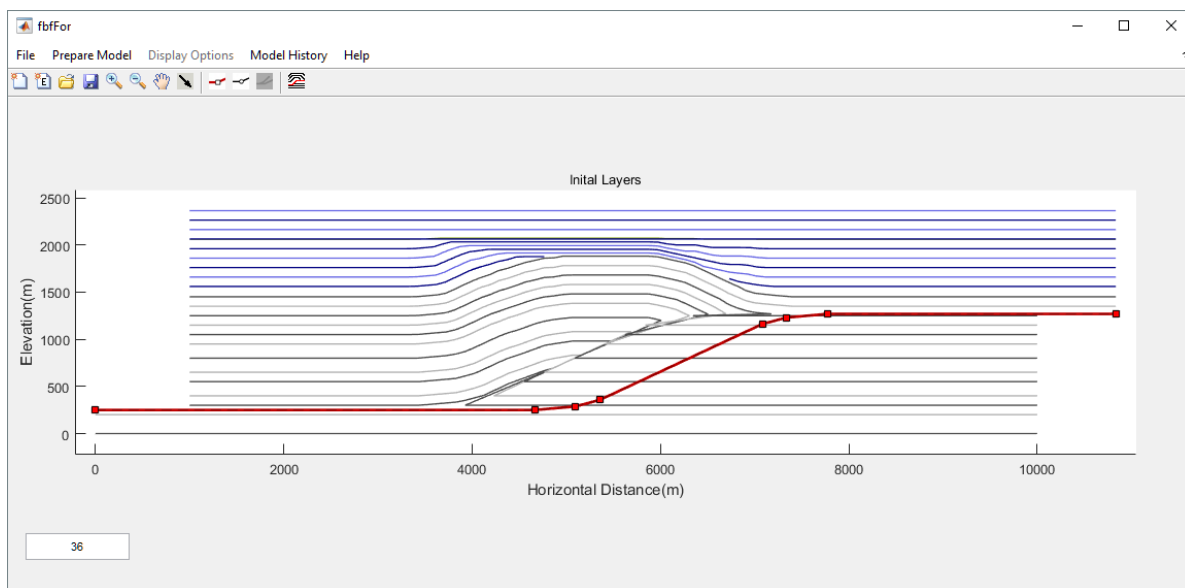


Note, there are some defaults on this choice, such as Subsidence 100%, Regional Dip -1° , closely spaced initial growth Layer Heights that are below sea level after subsidence, etc. If in general you are interested in modeling extensional fault-bend

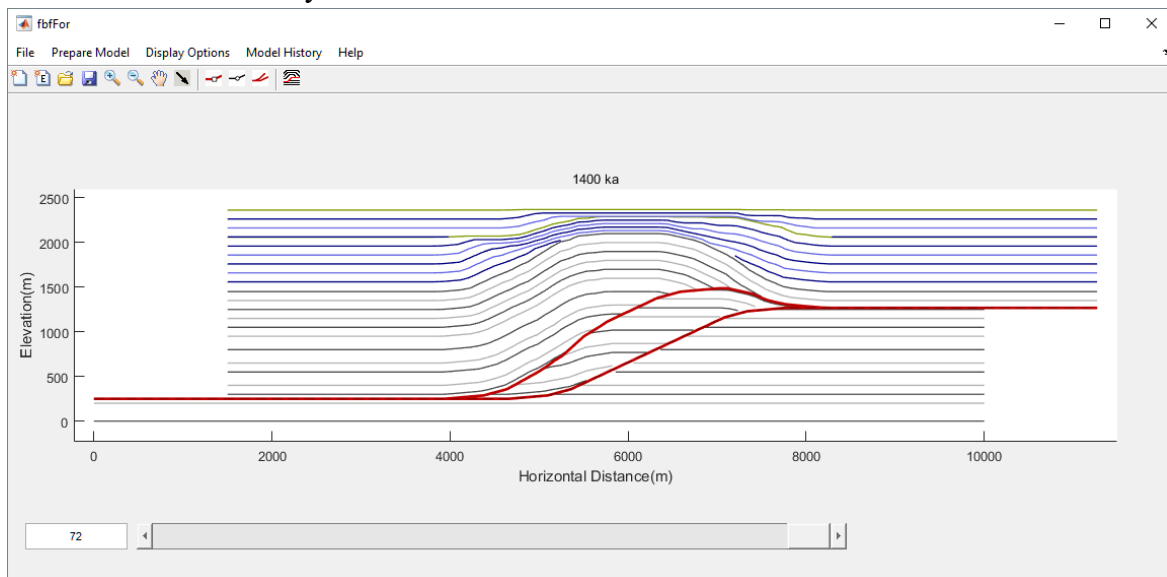
folds (rollover structures) then it might make sense to save an initial fault bend and layers as a template to reload for the future.

7. How do I make a duplex?

After making a model, then add an imbricate fault from the **Fault Panel** or **Icon Bar**. The default initial imbricate shape will often make a fault that has a common floor and roof thrust. If the first and second faults do not share common detachments then edit the imbricate fault so that it does have the same detachments as the original.

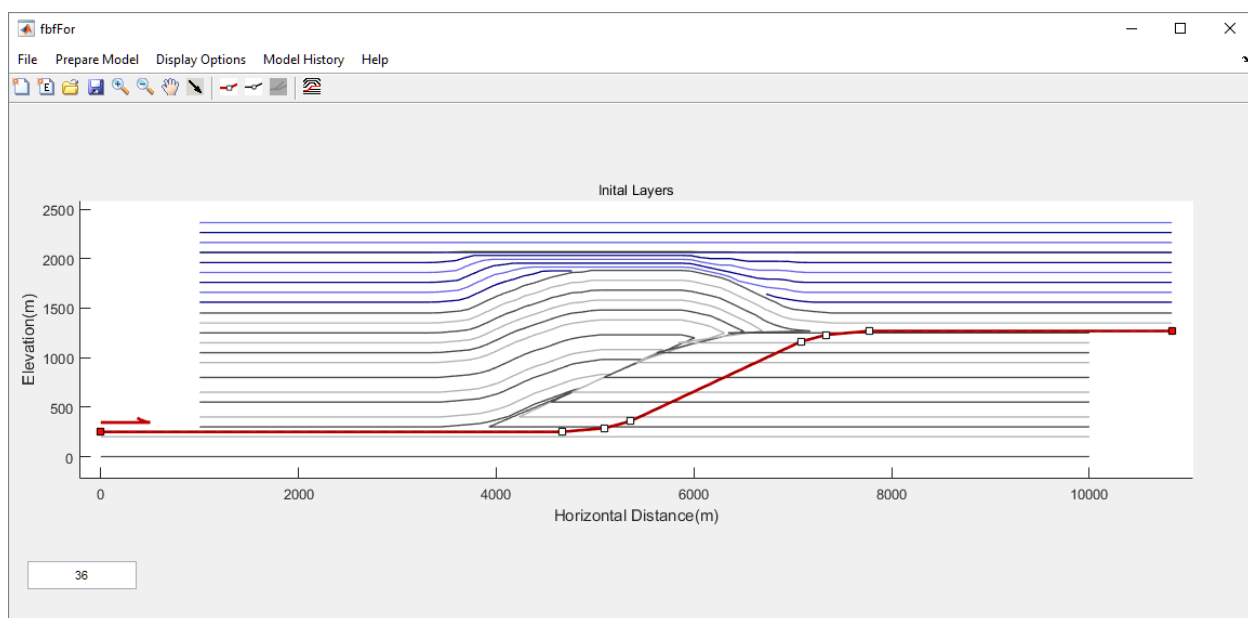


And then deform layers.

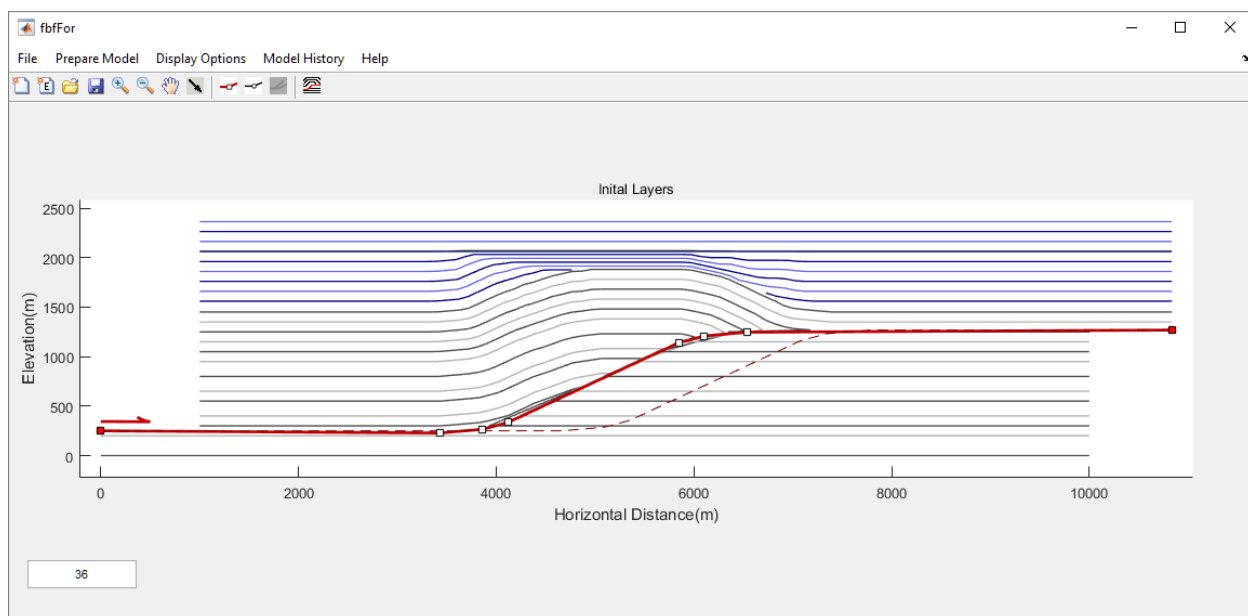


8. How do I move again on an existing fault?

After making a model, then add an imbricate fault. Select most of the points of the imbricate. Note the points are white if they are selected.



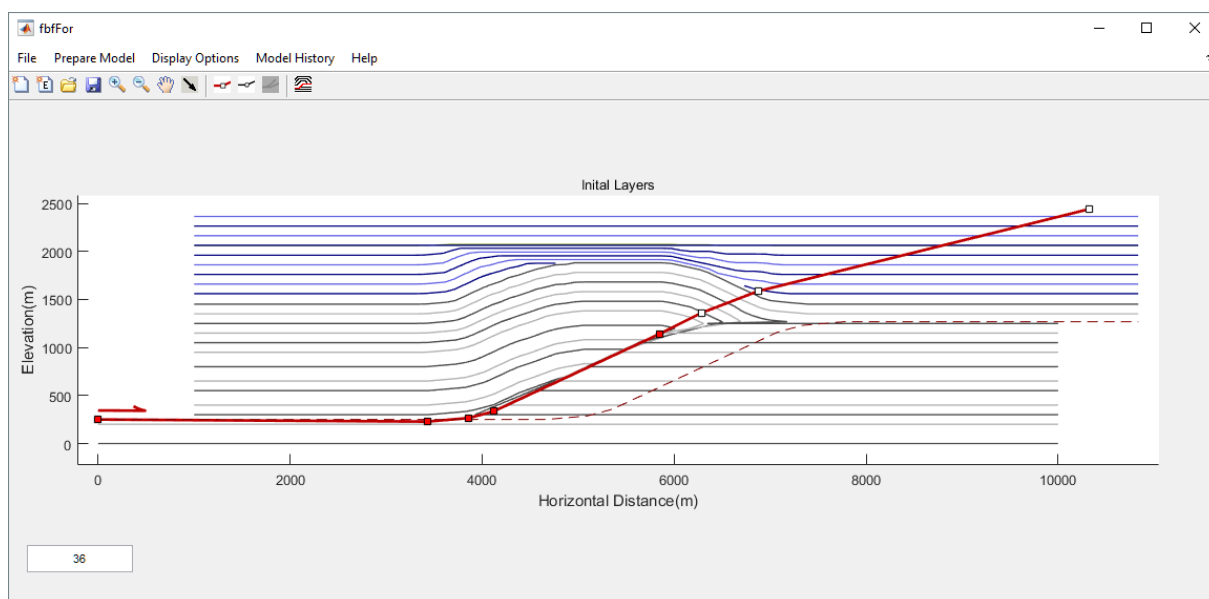
And slide them back to the original fault as shown below. Or export the first fault, make an imbricate, and import that first fault to replace the second fault.



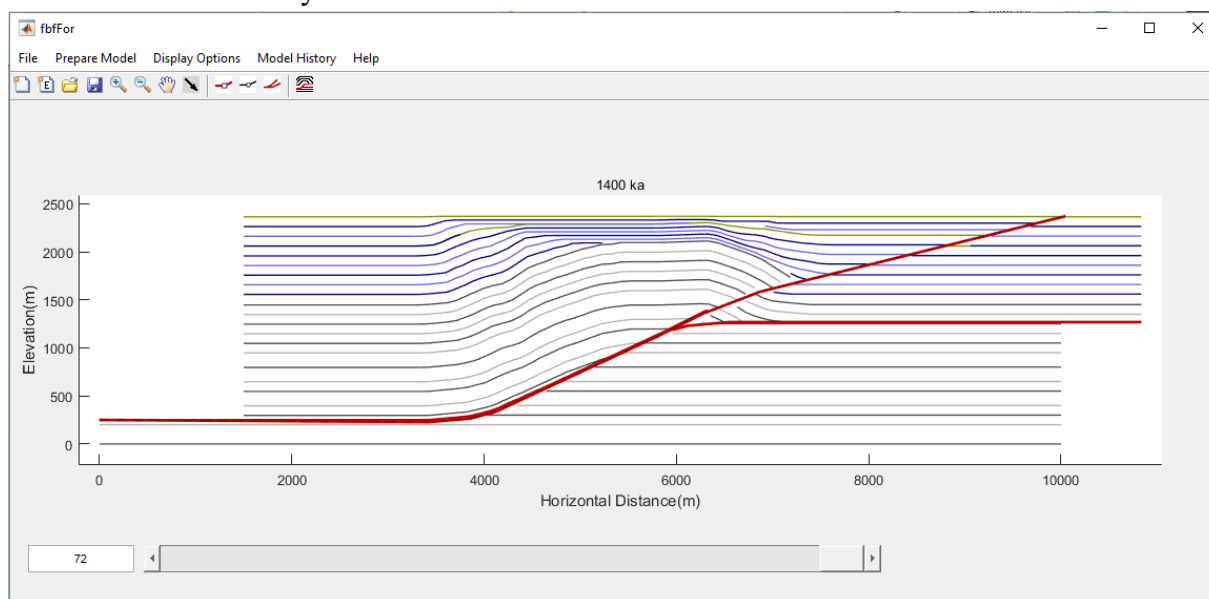
Then deform layers (not shown). Why would you want to do this? If you wanted to change the slip rate or vary the growth layers, but not the fault shape, then this would be reasonable. More commonly one would do this as an intermediate step on the way to making a fault splay, as described below.

9. How do I make a fault splay?

Follow the workflow above and then edit the fault points on the leading edge to breach the surface:

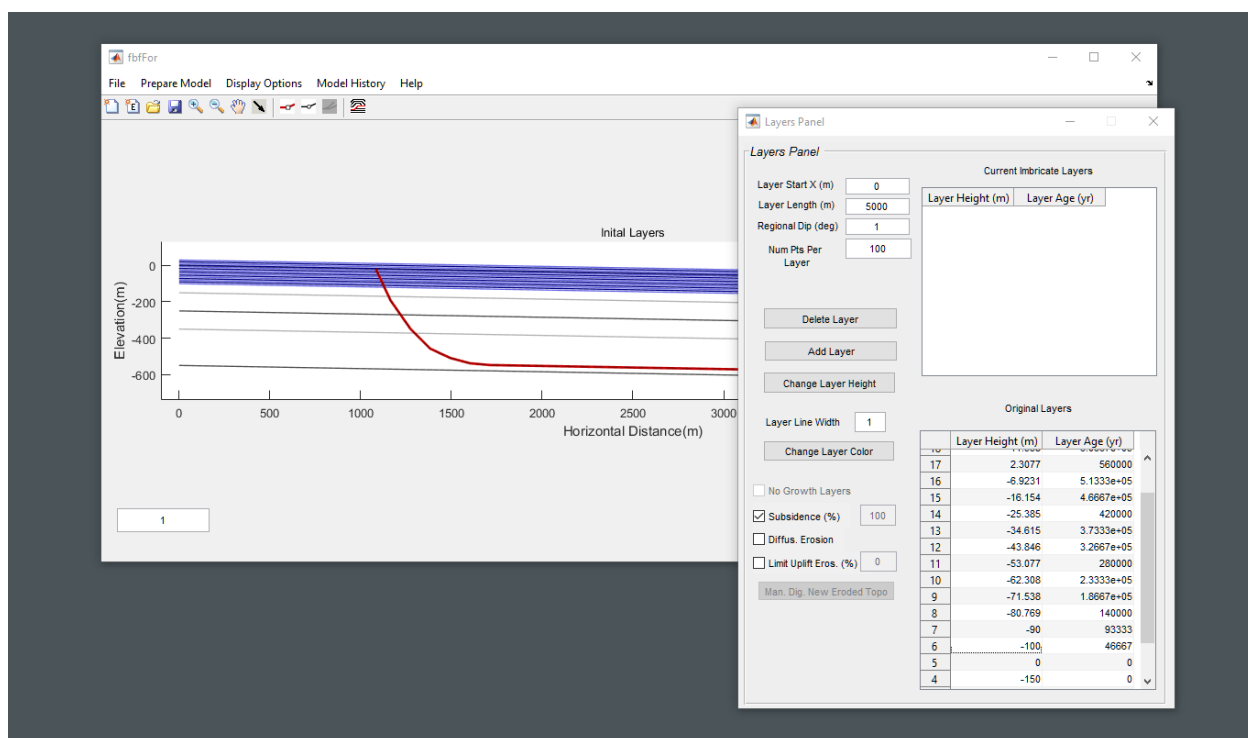


Then deform the layers.

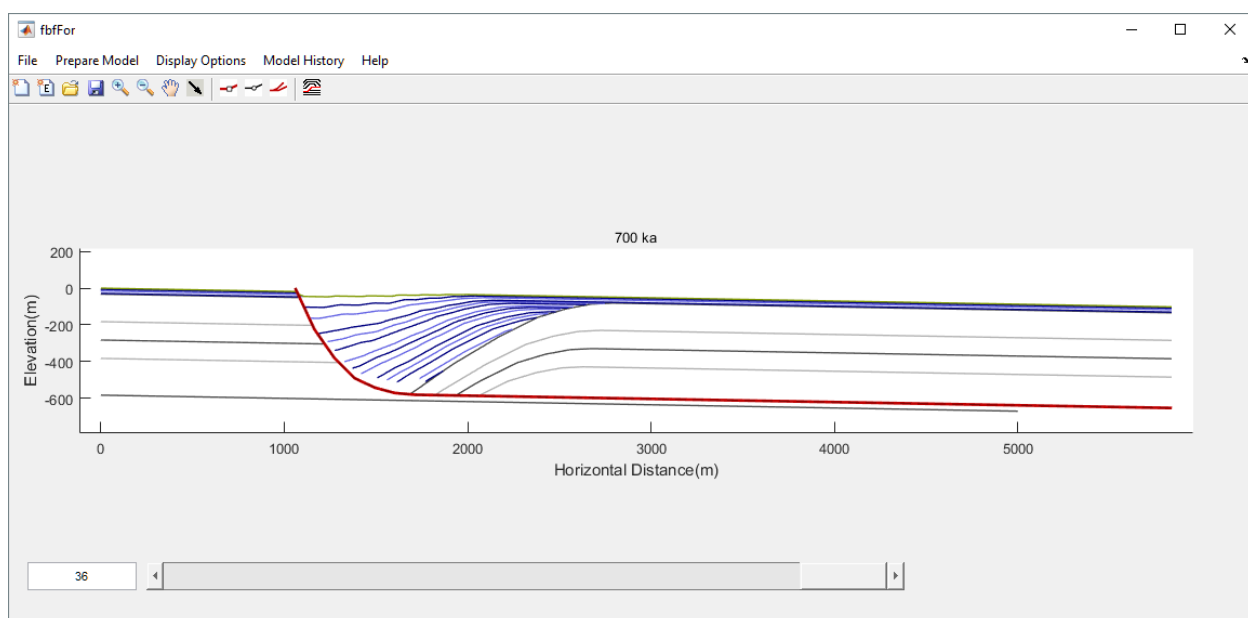


10. How do I model an under-filled basin?

Make some of the growth layers a lower initial height than the pregrowth layers.



Then deform layers. You will commonly see pinch out of some of the growth against other layers.



Model Constraints

Several modeling constraints are imposed to make computation possible and practical. These include:

- Pre-deformational layer geometry is flat or has a regional dip (from -6 to 6 deg.).
 - More realistic sedimentation is not currently implemented
- The number of points within one layer is constant for all imbricates.
 - but may vary between hanging wall layers based on growth and whether a fault cuts a layer (user does not control this, only the nominal number of points per layer).
- Heights are relative to an external, fixed reference frame so a new layer height adds to the model.
- Subsidence is a percentage of this layer height.
- Pregrowth layers must increase in height.
- If a growth layer height is lower than any older layer then it will only show up (and exist) if the deformation of an old layer creates sufficient accommodation (such as in an under-filled basin) or subsidence is great enough to make accommodation.
- Slip is only positive to the right, no inversion yet.
- Initial (input) slip rate is constant for a given imbricate (deformation event).
- The number of timesteps is constant for all imbricates in a model.
- Faults cannot double back on themselves, sorry no wedges yet.
- Faults extend to the boundaries of the model space for that timestep, no fault propagation folds yet.
- For each timestep, faults extend based on growth and deformation to the boundaries of a model.
- The fault slope of a contractional segment cannot exceed 65 deg.
- The fault slope coming out of an extensional convex bend cannot exceed the inclined shear angle.
- The active fault is that for the current timestep visible, and deformation will be calculated for that imbricate.
- Deformation of older faults during imbrication is generally passive.
 - So, the refolded limb will not conserve layer thickness, but will conserve area.
- Reactivation of faults currently is limited to the first imbricate, and not on by default.
 - In this case the refolded limb will theoretically conserve layer thickness more closely, but there may be layer-parallel strain.
- Velocity boundaries are determined based on the fault bend and slip.
 - But these can be overridden. The dip of the velocity boundary imposed is relative to an external reference frame. Layer-parallel strain will almost always result.
- Strain ellipses don't exist in growth strata.
- The relative particle path point cannot be in a growth layer.
- Track particles cannot be in a growth layer.

Numerical Solution Implications

The program solves numerically for the fold shape from a given fault shape, input slip and velocity boundaries at specified timesteps. There are several implications of such an approach:

- There is finite precision to the models. You can never get a perfectly angular fold. You always have a small amount of extension or shortening in segments that span a velocity boundary.
- Calculation time increases linearly by the number of points, number of timesteps.
- Calculation time increases almost linearly by the number of fault bends.
- Calculation time increases by $\sim 2^n$ for number of imbricates. File sizes similarly get huge when making lots of imbricates.
- Cutoffs for the first timestep are based on horizontal layers. This can create a small artifact at the beginning time of a model that exists throughout.

Notes of Caution and Guidance for Best Practices

Numerical tradeoffs

- There is a tradeoff between precision and how long it takes a model to run. This can easily take a very long time if you are not careful. Good rules of thumb include:
 - The ratio of number of points per layer to number of steps works best at about 3:1.
 - Keep the ratio of layer length to number of points per layer to no greater than 100.
 - Don't have a lot of small fault bends very close together. If you do you might get numerical precision artifacts. You also might need to tweak the number of points per layer and/or the number of steps.
- Trying more than four imbricates may take a long time, and may need more points and longer layer lengths than originally anticipated. This will also result in large saved files.
- Cutoffs can only be calculated on layers that exist. If there are only a few layers, there will not be very precise cutoff calculations. Even if you have a lot of layers, a very complicated fault model may result in poor definition of cutoffs. Sometimes adding layers will help with this.
- Likewise, if there are too many fault bends there may be no points within a velocity domain. The best solution to this is to decrease the number of bends.
- Cutoffs are based on the closest layer segment (usually, though there are heuristics to figure out "better" segments). Key point is that more points per layer is not better always as you might create issues with cutoffs being very small and not representative.
- Sometimes for the final timestep or two there is no representative cutoff left to get folded. This can create a small artifact at the end time of a model. If that happens make the model longer on the left side or shift the fault to the right.

Forward modeling tradeoffs

- If you want to bring in an image, subsidence and compaction will be turned off in the *Layers Panel* because the image is fixed to an external reference and not altered by the program during deformation. The reason for this is that modifying a fault (associated with a seismic section for example) is naturally done in the deformed (already subsided, compacted) state. The approach fbffor is forward modeling, and thus one doesn't know the un-subsided, un-compacted fault shape, or for that matter the initial layers positions. At some point, some smarts may be incorporated into the program to provide good first-order assumptions for these values instead, but without that kind of input, iteratively guessing what those values are is time-consuming and difficult, and is not recommended.
- A similar limitation exists for imbricates in that you edit a new fault before deformation occurs. This makes it a challenge to forward model a complicated structure through a series of steps in order to exactly match in a deformed state. One of the drawbacks of pure forward modeling, I suppose, relative to classical restoration.

Layers and compaction problems

- Compaction is only updated when a new growth layer is introduced, so if there are few growth layers, small artifacts can occur because there is a jump in compaction after some timesteps where none occurred. The best way to solve this is to have many growth layers so that there is very little time that passes without a new layer coming in.
- Note, for subsidence this generally isn't a problem because the fractional amount between layer heights is determined and applied automatically, but there is the possibility for a similar problem with complicated emergent structures.

Layers and growth strata problems

- If you've generated a model in which none of your growth strata seem to have deposited, check the ages that you have specified for the growth strata in the *Layers Panel* relative to the Cum Years designated in the *Slip Panel* to ensure that they are internally consistent.
- Another thing to check for is the layer height of the growth layers. If the structure fully takes up the accommodation space, then there is no place for the layer to be deposited. This can happen when you are trying to model underfilled basins, and don't fully consider the uplift of the structure, and perhaps associated subsidence.
- This will result in layers that have no points because the layer height is lower than the whole topography. This is not by itself a problem for fbffor, but there are times when fbffor can get confused by this with multiple such layers. In general, do not make younger layers lower than older layers unless you really know you want to do this.

Defining the fault shape

- To make a thrust start digitization to the left of the layers and digitize points. To make a normal fault start digitization above the pregrowth layers. fbffor will truncate the fault to the limits of the layers at any timestep including during this digitization step.
- The reason for starting faults in this way is that fbffor chooses whether fault bends are contractional or extensional based on the slopes of the fault segments. Because input slip can only be towards the right and is parallel to the first fault segment, positive slopes are in general contractional (accommodate shortening), and negative slopes extensional (accommodate extension).
- You can create thrust faults that enter from the bottom/top of the model and/or exist the top of the model. Use this sparingly. A particular tricky case is where a fault enters and exists from the top of the model. This in general works, but can confuse fbffor, if you are making imbricates that do this. Likewise having an imbricate where one fault enters from the left, and then another enters from the bottom may confuse fbffor. The program really expects a thrust to start from the left, and a normal fault from the top.

Faults emerging to the surface

- There are potential instabilities in eroded topography such that you might see unrealistic spikes or regions in some runs. This can happen when thrusts are emergent (which is allowable) where there is lots of slip and little sedimentation. So, consider this choice of having emergent thrusts experimental at this point.

Faults that are corrugated, scoop-shaped, or have lots of small segments

- You should in general, try to not make corrugated or scoop-shaped faults if possible, unless fault slopes are less than a few degrees. By “corrugated” I mean faults that have slopes that change from positive to negative repeatedly. Besides being geologically unrealistic, they are very difficult to determine the velocities for, and it is possible that the algorithms will not work right, including not conserving cross-sectional area. A simple scoop-shaped should work. By “scoop-shaped” I mean a fault with a segment that has a negative slope and then immediately followed by a positive slope.
- Another thing to be mindful of is to not have too many fault points that are really close together. Besides taking longer to compute, each fault point defines a velocity boundary. If you have many fault points close together, then boundaries are very close together, and it is possible to have no cutoffs within a velocity domain. While fbffor should not usually break under these circumstances, it is possible to create numerical instabilities. The upshot is that area cross-sectional area may not be conserved.
- If for some reason you want to have one of these faults and you notice that cross-sectional area is not conserving you could try to add layers, particularly where they pass through these weird bends. That may solve the area-conservation issue, but of course does not mean it is geologically reasonable.

Smoothing faults

- Smooth Fault in the *Fault Panel* is designed to create less angular fault shapes, but does of course add fault points because more fault segments are needed. Smooth Fault attempts to remove redundant points, but sometimes needs to be run twice to achieve this. Running it many times however, may just create too many points and then you can have problems with improper definition of velocity domains and cutoffs, as discussed above.

Faults that reactivate

- If the Reactivate Fault box is checked, during contraction the second fault will cause the first fault to deform, and slip will be induced on the first fault in order to better maintain layer thickness (see Connors et al., 2021).
- If after deformation you notice that the fault did not actually reactivate it is because the previous fault was cut by the new fault. Currently in fbffor you cannot reactivate any fault that is cut by the older fault. To fix this, move the points of the new fault down so that they do not cut the old fault. Sharing the same detachment should not cause the old fault to be cut.
- Consider Reactivate Fault a feature that is quite experimental. Only use it when you think you really need it. Besides taking much longer to compute, this is a very difficult thing to get right, and sometimes fbffor does not calculate this correctly.
- If you really want to use Reactivate Fault and it seems to not be working properly, you can try and add more layers that pass through fault bends. You can also try and add more points per layer and more timesteps (these should be done in tandem as discussed elsewhere in this manual).

Known Bugs

- The area calculation uses a routine that has to deal with shapes that are both convex and concave locally and so defining a fault block can occasionally not be geologically accurate. This happens say when two detachment levels are implemented or when one fault enters from the left and one from the bottom of the model layers. In these cases, showing the area in the *Display Panel* will make it appear as though area has not been conserved. It is not that the deformation has resulted in non-area-conserving structures, but rather is an error of the subsequent calculation of the blocks that define the areas.
- Eroded hanging walls can sometimes cause the definition of the full hanging wall block to be off, with an incorrect area being reported for the hanging wall. Again, this is about defining the block shape, not that the deformation is not conserving area.
- Manually digitizing erosion of the leftmost point of the highest pregrowth layer can cause the definition of the hanging wall block to be off.
- Faults that breach the topographic surface sometimes do not get truncated correctly, or sit slightly above topography. This may cause the defining of fault blocks, layers, and the topographic surface to not be correct. This may also cause growth layers to not exist across the whole model.
- Faults that breach the topographic surface can cause the topographic surface to be irregular. This is an artifact of the way the topographic surface is rebuilt at each timestep.
- Reactivate can get confused. This happens when fbffor cannot figure out how to reactivate a fault that is also being cut by another fault. Reactivate should be considered experimental.
- Editing a fault with subsidence may shift to the final subsided state, rather than the initial state for a deformation step. If this happens, you have to reload the model, or recreate the last imbricate.
- Compaction sometimes creates artifacts if there are only a few growth layers. This occurs because compaction is only calculated when a new layer appears. This can also occasionally cause a point on a layer to move across a fault or another layer. Right now, the only solution is to create more growth layers so there isn't an abrupt change in the amount of material added at a timestep.
- Using a fixed Axial Slope can sometimes result in odd results. While the model will calculate a geometry for a wide range of fixed axial slope situations, not all of them are physically reasonable, and overriding the default options should be done with caution. Low slope values are particularly problematic because they create velocity domains that interfere with each other extensively.
- Corrugated faults may not conserve area. They should be avoided. By "corrugated" I mean faults that have slopes that change from positive to negative repeatedly. Besides being geologically unrealistic, they are very difficult to determine the velocities for. A simple scoop-shaped should work. By "scoop-shaped" I mean a fault with a segment that has a negative slope and then immediately followed by a positive slope.

- If a model has lots of imbricates, with lots of layers and points then the model size can exceed the size that can be saved. Try not make models with more than about 4 imbricates. More than that will also slow fbfFor down.

Potential Future Improvements

(don't hold your breath for any of these, but I plan to try and get to these as time and resources allow)

- Incorporate a near-surface plotting of contacts and attitudes from a geologic map.
- Implement erosion, subsidence, and compaction in a more sophisticated fashion.
- Implement more realistic sedimentation, perhaps with eustasy.
- Extend to 3D a plane strain solution.
- Improve the contractional imbricates with more robust reactivated fault.
- Implement structural wedges (triangle zones) and inversion structures into the gui.
- Add trishear, detachment folding, and shear fault-bend folding.

Acknowledgements

My co-authors in our paper on kinematic forward modeling of fault-bend folding Amanda Hughes and Stephen Ball have worked on various aspects of fbfFor over the years, and provided feedback on the material presented in this manual. Several Washington and Lee undergraduate research students participated in early aspects of the development of fbfFor, in particular Jeanne Upchurch, Michael Braunscheidel, Anne Lindsey McColloch, and Meredith Townsend. Simon Levy and Joshua Stough provided programming concept insights. Chevron provided a grant to Washington and Lee for the development of an early graphical user interface for fbfFor.

References

- Allen, P., and Allen, J., 2013, Basin Analysis, Wiley-Blackwell.
- Connors, C., Hughes, A., and Ball, S., 2021, Kinematic Forward Modeling of Fault-bend Folding, Journal of Structural Geology, v. 143, p. 1-21, doi.org/10.1016/j.jsg.2020.104252.
- Pellitier, J., 2008, Quantitative Modeling of Earth Surface Processes, Cambridge Univ. Press.
- Sclater, J., and Christie, P. 1980, Continental stretching: An explanation of the post-mid-Cretaceous subsidence of the central North Sea basin, Journal of Geophysical Research. 85, p.3711-3739.
- Suppe, J., 1983, Geometry and kinematics of fault-bend folding, American Journal of Science, v. 283, p. 684-721.
- Xiao, H., Suppe, J., 1992. Origin of Rollover. AAPG Bulletin, 76, 509-529.

Copyright Notices

fbfFor v. 1.20, Jan. 2021

Chris Connors.

other contributors: Amanda Hughes and Stephen Ball

contact: connorsc@wlu.edu

fbfFor Copyright (c) 2002-2021, Chris Connors

All rights reserved.

Redistribution and use in binary form, without modification is permitted provided the above copyright notice and the following disclaimers are provided with distribution.

Neither the name of any organization nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.