LIFFE: LITHOSPHERIC FLEXURE WITH FINITE ELEMENTS

USER MANUAL

BY MARIANO S. ARNAIZ-RODRÍGUEZ

MARIANOARNAIZ@GMAIL.COM

MARCH, 2020

Introduction **3**

Where to begin? **4**

A LIFFE project, Input and Master files **5**

The Input Files 5

DATA.txt 5

loads.txt 6

LIFFE.m: The Master File 10

Parameters of the Flexure 10

PARAMETERS of the Elements and Plate 12

TE VARIATIONS 13

The Finite Element Method Problem 14

TEST EXAMPLES **16**

Example 1: Offshore broken plate with constant elastic thickness. 16

Example 2: Inland graben with variable elastic thickness. 18

Example 3: Long term deformation of the crust. 20

Some tricks **22**

Reading the Report 22

Plotting Sections of the Plate 22

Stresses 23

Figures in Postcript Format 23

INTRODUCTION

Welcome to LIFFE!

We developed LIFFE, because every time we wanted to study a flexure system we found problems that the existing codes could not address, such as the variation of elastic plate thickness and a hungry basin. Also, we wondered: What would happen if we considered compression at the edge of the plate?, or how is the stress field in the interior of the plate after the flexure?. We wanted to be able to model lithospheric flexure considering as many parameters as we could, which is not always easy with the approaches that were available to us. One problem in particular drove us here, What happened if the plate is broken at both ends?. We had no way to model this situation, we even asked Anthony Watts (from Oxford University) for his advice and he gave us one of his codes, but as the vertical load of the topography were not the only ones acting on the plate, we were unable to use it to finish our research. We decided to build a Finite Elements (FEM) code that would allow us to do almost anything we wanted: consider elastic thickness (Te) variations, previous deformations, complex load systems, temperature effects, decoupled infill loads and specify how much of the basin is filled.

We hope that this code can help you model the flexure of the lithosphere in an easy and flexible way (*pun intended*).

LIFFE is and OPEN ACCESS code. Feel free to download it, distributed it, modify it. If you use it, like it, or have a comment, feel free to send us an email!

WHERE TO BEGIN?

The first thing you will need to use LIFFE is MATLAB. Our code has been used with MATLAB 2012b, 2015b, 2016b and 2018a, and it seems to work in all versions. We have tested it on Windows and Mac OS. Everything you need should be in the directory LIFFEv1 that contains some examples of how to run the code. Within each Example# directory you will find the bin directory (containing all the functions that make up LIFFE); the distmesh directory (having the *distmesh* distribution by developed by Persson and Strang, 2004, **running "mex -setup" in MATLAB's console might be necessary to use this package**); and three files: DATA.txt (observed data file), loads.txt (a file containing the loads to be applied) and LIFFE.m (master Matlab file).

Within the LIFFEv1 directory you will also find a readme.txt, this manual and an Extras directory that contains the ExamplePaper directory (with all the files needed to test the Hawaii case found in the paper) and the create_empty_loads_file.m that will help you create an empty loads file. I would recommend that you begin with one of the Example# directories, opening the master LIFFE.m file and running it on MATLAB. **IF everything is OK**, you should get a bunch of images and a summary on the command window. If something fails, try another example or make sure that your MATLAB has a compiler so the *distmesh* distribution is able to run.

A LIFFE PROJECT, INPUT AND MASTER FILES

LIFFE was designed to operate in any way you desire, but we found that a nice way to keep the modeling organized is by using a directory for each profile, that is why the *Examples#* were created. You could also add the *bin* and *distmesh* directory to the Matlab path and have all the functions available to you anytime and just use the input and master files. Let's discuss about them now.

THE INPUT FILES

LIFFE requires two input files: *DATA.txt* and *loads.txt*. The first one holds the observed information along the profile while the second holds the forces to be applied to the elastic plate.

DATA.TXT

The *DATA.txt* file (Fig. 1) should be a 4 columns space or tab delimitated ascii file. Each column must have the same amount of lines. The columns are, respectively: distance along the profile (km), Topography and bathymetry (m), Free Air gravity anomaly (mGal) and Bouguer gravity Anomaly (mGal). The four columns are necessary, but if you don't have one of them, you can fill it with *0s*. The files in the "Examples#" directories have a header (Fig. 1) that is neglected by MATLAB, when the file is read. You can do the same if you desire. <u>Also, if you desire to use a different name for this input file, you will need to change the expected name on the **LIFFE.m** master file at line 122.</u>

```
%Distances Topography FAA BA
0     -5000 -12.497 5.6756
0.001 -5000 -12.497 5.6756
0.002 -5000 -12.497 5.6755
0.003 -5000 -12.498 5.6755
0.004 -5000 -12.498 5.6754
0.005 -5000 -12.498 5.6753
0.006 -5000 -12.498 5.6753
0.007 -5000 -12.498 5.6752
0.008 -5000 -12.498 5.6752
0.009 -5000 -12.498 5.6751
0.01 -5000 -12.498 5.6751
0.011 -5000 -12.498 5.675
0.012 -5000 -12.498 5.675
0.013 -5000 -12.498 5.6749
0.014 -5000 -12.498 5.6749
0.015 -5000 -12.498 5.6748
0.016 -5000 -12.498 5.6748
0.017 -5000 -12.498 5.6747
0.018 -5000 -12.498 5.6747
```

Fig 1. Example of the input file DATA.txt

LOADS.TXT

The *loads.txt* file (Fig. 2) holds the information of the topographic load on top of the plate and all the other forces applied to its edges. It (Fig. 2) should be an 8 columns, space or tab delimitated, ascii file. Each column must have the same amount of lines, which are respectively: distance along the profile for the on-top loads (km); height of the topography at the previous distance (m); distance along the profile for the bottom loads (km); force of the load at the previous distance (Newtons, N); depth of the right side of the plate (km); force of the load at the previous depth (N); depth of the left side of the plate (km); force of the load at the previous depth (N). The eight columns are necessary but, if there is no force or topography, just use *0s*. The files in the "Examples" directories have a header (Fig. 2) that is neglected by MATLAB when the file is read. You can do the same if you desire. <u>Also, if you desire to use a different name for this input file you will need to change the expected name on the **LIFFE.m** master file at line 121.</u>

```
%x_top  h_top   x_bot   f_bot   z_right f_right z_left  f_left
0       0       0       0       0       0e13    0       -0e12
20      0       20      0       1       0e13    1       -0e12
21      0       40      0       2       0e13    2       -0e12
60      0       60      0       3       0e13    3       -0e12
80      0`      80      0       4       0e13    5       -0e12
100     0       100     0e10    9       0e13    9       -0e12
139     0       120     0e10    12      0e13    12      -0e12
140     0       140     0e10    13      0e13    13      -0e12
145     0       142     0e10    14      0e13    14      -0e12
150     0       144     0e10    15      0e13    15      -0e12
155     0       146     0e10    16      0e13    16      -0e12
160     0       160     0e10    17      0e13    17      -0e12
161     0       180     0       18      0e13    18      -0e12
279     0       200     0       19      0e13    19      -0e12
280     5000    220     0       20      0e13    20      -0e12
283     5500    240     0       21      0e13    21      -0e12
286     5750    250     0       22      0e13    22      -0e12
289     5700    260     0       23      0e13    23      -0e12
292     5625    270     0       24      0e13    24      -0e12
295     5800    280     0       25      0e13    25      -0e12
298     5600    290     0       26      0e13    26      -0e12
300     4500    300     0       27      0e13    27      -0e12

%h_top = heigh of topography
%fbot = negative is upward force
%f_left = usually zero, negative is extension to the left
%f_right = negative is compression
```

Fig 2. Example of the input file *loads.txt*

The first two columns of this file referred to the topography on the top of the plate. You just need to input the height of the column at each distance and the code will compute the weight. <u>The nodal coordinates and the distances on this file are independent from each-other.</u> This is done for flexibility, and means that you do not have to manually assign the loads to a node. To accommodate this, loads are linearly interpolated to the location of the nodes, which means that the topography needs to be contained. For example, if we consider a load that is a box of 10 km long and 3000 m heigh, you would need to input something like the example in Fig 3. Notice that the load goes from 20 to 30 km in the profile, but we limit it between 19 and 31 km. If the 5 km spacing would have been kept, the load would extent from 15 to 35 km as the information is interpolated to 1 km spacing. Therefore, we need to put a limit to it. <u>This system is employed in all the columns pairs.</u>

```
%x_top h_to
0      0
5      0
10     0
15     0
19     0
20     3000
30     3000
31     0
40     0
45     0
50     0
```

Fig 3. Example of the containment of a load.

To input loads at the bottom of the plate, you can follow the same procedure. So, you should assign values to the distances where the load will be applied and limit them to 1 km before and after the section of the plate to be loaded. <u>You should keep in mind that the minimum spacing accepted in 1 km</u>. In the next column, fill out the forces in Newtons (usually these values are quite large). Forces at the bottom could be upward forces (that would cause uplift) or downward (that would cause subsidence). So, if we wanted to add an upward force of $1x10^{11}$N that acts from 35 km to 45 km to the previous example, the loads.txt file would be similar to the one shown in Fig. 4. <u>Notice that that the distances in columns 1 and 3 are completely independent and that loads can be designated to any distance in the profile.</u> If the force was downwards then all that needs to be done is change the sign.

```
%x_top  h_top   x_bot   f_bot
0       0       0       0
5       0       1       0
10      0       2       0
15      0       34      0
19      0       35      -1e11
20      3000    42      -1e11
30      3000    45      -1e11
31      0       46      0
40      0       0       0
45      0       0       0
50      0       0       0
```

Fig 4. Example of both top and bottom loads.

Sometimes compressional or extensional forces need to be considered. This is achieved by filling the last 4 columns. If the plate is being compressed at its right end by a force of $1x10^{12}$N but the left one is fixed and there are no forces acting on it, then the file would look like the one in Fig 5. Notice that the sign convention for the forces is the same as the one for the coordinates system (positive to the right and down). Therefore, if a force should deformed the body towards a direction, the sign is the same as the one of such direction. Unlike in the case of the vertical forces, where the maximum distance to apply a force should be the length of the plate, for the horizontal ones the range does not matter since forces out of range will be discarded. Also, the force does not have to be the same for all the depth, it can vary or be applied just to a section of the plate.

%x_top	h_top	x_bot	f_bot	z_right	f_right	z_left	f_left
0	0	0	0	0	-1e12	0	0
5	0	1	0	1	-1e12	2	0
10	0	2	0	2	-1e12	5	0
15	0	34	0	3	-1e12	15	0
19	0	35	-1e11	4	-1e12	15.5	0
20	3000	42	-1e11	5	-1e12	15.6	0
30	3000	45	-1e11	6	-1e12	15.7	0
31	0	46	0	7	-1e12	17	0
40	0	0	0	8	-1e12	18	0
45	0	0	0	9	-1e12	20	0
50	0	0	0	25	-1e12	25	0

Fig 5. Example of top, bottom and right side (compression) loads

If the plate were in compression at its both ends, then the file would change. Notice the different signs for the same kind of force. If the plate suffered from extension, the sings just need to be swapped.

TIP: Remember that forces and boundary conditions must be in agreement for the model to work.

LIFFE.M: THE MASTER FILE

The *LIFFE.m* file is the master MATLAB script in which all the global variables and boundary conditions are defined; it also summons the functions on the *bin* and *distmesh* directories. This also the file where elastic thickness (*Te*) and the shape of the plate are changed. The master file begins with a short presentation of the code. After that, the code is set to clear the Matlab environment, add the directories to the path and keep track of the elapse time (lines 47 to 57). After that, you can begin to define the parameters to be used during the calculation.

PARAMETERS OF THE FLEXURE

The first parameters section (*Define PARAMETERS of the FLEXURE*) is used to define the mechanical, thermal and environmental parameters. The variables are:

MOHO: The thickness in meters of the undeformed crust. This value will be added to the flexure on top of the plate to simulate the flexed base of the crust. It is only considered for the gravity anomalies of the model.

BASEL: The base line of the flexure is defined as the elevation of the plate before the flexure in meters. For example, usually the plate is considered to be at sea level (BASEL=0;) but, if it were a plateau at 1km above sea level, then BASEL=1000. If the plate is underwater at 3 km depth, then BASEL=-3000.

WATER: This flag variable is design to specify if the flexure is in water: WATER=1 for underwater flexure, and WATER=0 for on-land flexure. This is important as the load's weigh is different in both cases.

v: Is the Poisson's ration of the plate (usually 0.25);

E: Is the Young's Modulus of the plate in Pa (usually 70e9 Pa or 100e9 Pa).

muk: Is the coefficient of kinetic friction between the plate and the plate next to it. This parameter is used to damp the flexure at an end considering another plate exists next to it. It is only needed for one boundary condition.

fillindex: Is the percentage of material infilling the flexural moats. If fillindex=1, then the depression is 100% filled, while fillindex=0 means that the depressions are empty. Under-filled and over-filled depressions are allowed (0.25 is a 25% filled basin, and 1.25 is a 125% filled basin).

T0: Is the temperature at the top of the plate in Celsius. If the plate is underwater, use the appropriate value.

GradT: Is the geothermal gradiente in Celsius/meters.

alfa: Is the coefficient of thermal expansion of the plate in 1/Celsius.

AtmGratT: Is the thermal gradient of the medium around the plate (air for land and water for sea).

grav: Is the gravity acceleration in meters/seconds*seconds.

PARAMETERS OF THE ELEMENTS AND PLATE

The second section (Define PARAMETERS of the ELEMENTS AND PLATE) is used to set the size of the elements, the dimensions of the plate and the boundary conditions. The variables are:

te: The thickness of the plate in the direction perpendicular do the model (in and out of the computer's screen). This value has no effect on the flexure and is defined just as a formality.

L: The length in meters of the plate (or profile). This parameter is widely used in the code, be sure to specify it properly and be in agreement with the input files (*loads.txt* and *DATA.txt*).

H: The height of the plate in meters. H can be used to define a plate of constant elastic thickness. This variable is overlooked if it is not called at BOTTOM (line 126).

sizem: The size in meters of the sides of the elements. Sizem works great between 1000 m and 5000 m.

TIP: L, H and sizem are variables required to create the finite elements mesh. Sometimes distmesh can not solve for a particular case and this values need to be altered. Usually a different sizem solves the issue (e.g. change sizem, L or H from 3000 to 3001, 3100, 3500; or a combination of them).

R: A vector used to define the restrictions on the plate ends. The left (first) value acts on the left side and the right one (second) on the right side. The accepted values are:

R=0: NO Restrictions on the end

R=1: Horizontal displacement restricted. The nodes at the end can't be displaced in the horizontal direction.

R=2: Horizontal and Vertical displacement restricted. The nodes at the end can't be displaced.

R=3: Horizontal displacement is damped according to the crustal density.

R=4: Horizontal displacement is damped according to the crustal density and the vertical displacement is reduced by friction according to **muk.**

AFTER ALL THIS PARAMETERS ARE SET, LIFFE READS THE INPUT FILES loads.txt AND DATA.txt IN THE SECTION ENTITLED "READING LOADS and DATA FILES".

TE VARIATIONS

This section is exclusively assigned to set the shape of the plate. Only two vectors are inside this section: **TOP** (that defines the shape of the top of the plate) and **BOTTOM** (that defines the elastic thickness variations on the bottom of the plate). The vector **TOP**(:,1) begins at the end of the plate (**L**) and grows smaller until it goes to distance=0 m. The shape of the bottom (0 if is constant) is defined with respect to **BASEL**. If any value on **TOP**(:,2) if equal to **BASEL** then it should be equal to 0. If a section of the plate is depressed (lower than **BASEL**), then, a positive value should be in place. If a section is uplifted, a negative value is needed. For example, a plate with length L and a 2000 m depression around 200000 m:

TOP=[**L** 0; 400000 0; 300000 0; 210000 0; 205000 2000; 190000 2000; 180000 0; 100000 0 ; 0 0];

The vector **TOP** can be used to model any situation where the shape of the top of the plate cannot be considered flat, like in the flexure at river deltas. Take under consideration that a reduction or increase of this vector affects the stiffness of the plate.

The variable **BOTTOM** is used to define the elastic thickness (Te) variations along the plate. Unlike **TOP**, **BOTTOM** requires the distances from 0 to **L**. If Te is constant, BOTTOM(:,2) can be set equal to **H**. For larger Te just set a larger values. For example, if a plate has a Te that rages from 5000 m at one end to 35000 m at another with a thinner region in the middle:

BOTTOM=[0 5000; 100000 10000; 170000 12000; 200000 6000; 225000 6000; 245000 15000; 300000 20000; 400000 35000; **L** 35000];

The vector **pv** holds the close polygon that *distmesh* requires to build the finite element mesh. You do not need to edit this variable.

TIP: TOP and BOTTOM are variables required to create the finite elements mesh. Sometimes distmesh cannot solve a particular case, this values need to be slightly altered. Be aware that TOP and BOTTOM do not cross each other. BOTTOM can be filled with any value, even 0, but we can not guarantee the stability of the plate solution. At the ends of the plate, using 0 has proven useful in some modeling scenarios.

THE FINITE ELEMENT METHOD PROBLEM

Having come this far, all the parameters required by LIFFE have been defined. From here onwards the code will used them to solve for the plate flexure. We will only describe the process as each function has its own description. The code first calls *distmesh* to create the

mesh and the connectivity table (function: *MAKEdistmeshNODES*) and uses this information to compute all the initial matrixes (function: *FEMmatrixes*). Then, it creates the force vector with the information in loads.txt (function: *AddLoads*) and sets the necessary restrictions to the matrixes (function: *Restrictions*). The strain due to *loads.txt* is computed (function: *STRAIN*) and the sedimentary load is computed (function: *Sload*). From here the process is repeated, the new matrixes are recomputed based on the new geometry, restrictions applied and a secondary strain computed. Stresses due to none thermal loads are computed (*doSigmas*). Thermal loads are computed (function: *dTEM*P) and the process, repeated; but the stresses are computed differently (function: *doSigmasT*). Final strain and stresses are computed at lines 169 and 170. In the end, the plots are made and the gravity anomalies calculated by the plotting functions *plotFIGURES* or *JustModel* (both work identically, so just the name needs to be changed).

TEST EXAMPLES

Three test examples are provided with LIFFE. Here we present each and comment on the resulting flexure and principal stresses. Other figures created by the plotting routines are not presented in the manual but can be easily summoned by running the examples.

EXAMPLE 1: OFFSHORE BROKEN PLATE WITH CONSTANT ELASTIC THICKNESS.

The first example represents the underwater flexure of a broken plate with Te=25 km and a topographic load on its broken end. This is a simple model that is representative of the flexure of many foreland basins (e.g. Karner and Watts, 1983), but in this example the flexure occurs underwater. Here the right side is considered as a "Free End"; and the left one is considered as a "Clamped End". The load produces an initial depression that is completely filled with sediments. Since the computed Free-air and Bouguer gravity anomaly do not match the observed values, the topographic load must be incremented or the Te reduced to fit the data.

The deformation of the constant Te plate by a vertical load on its end produces high values of $\sigma 1$ on top of it, that drive the appearance of normal faulting on top of flexed crusts (Owens, 1983). Contrariwise, the bottom of the plate suffers from compressional stresses that are represented as negatives values of $\sigma 2$. Furthermore, Fig. 6f also shows compressional stresses on the top of the plate associated to the region affected by the topographic load. Finally, high values of τmax appear in both the top and the bottom of the plate, while the low ones accumulate far away from the deformation and in the centroid of the plate.

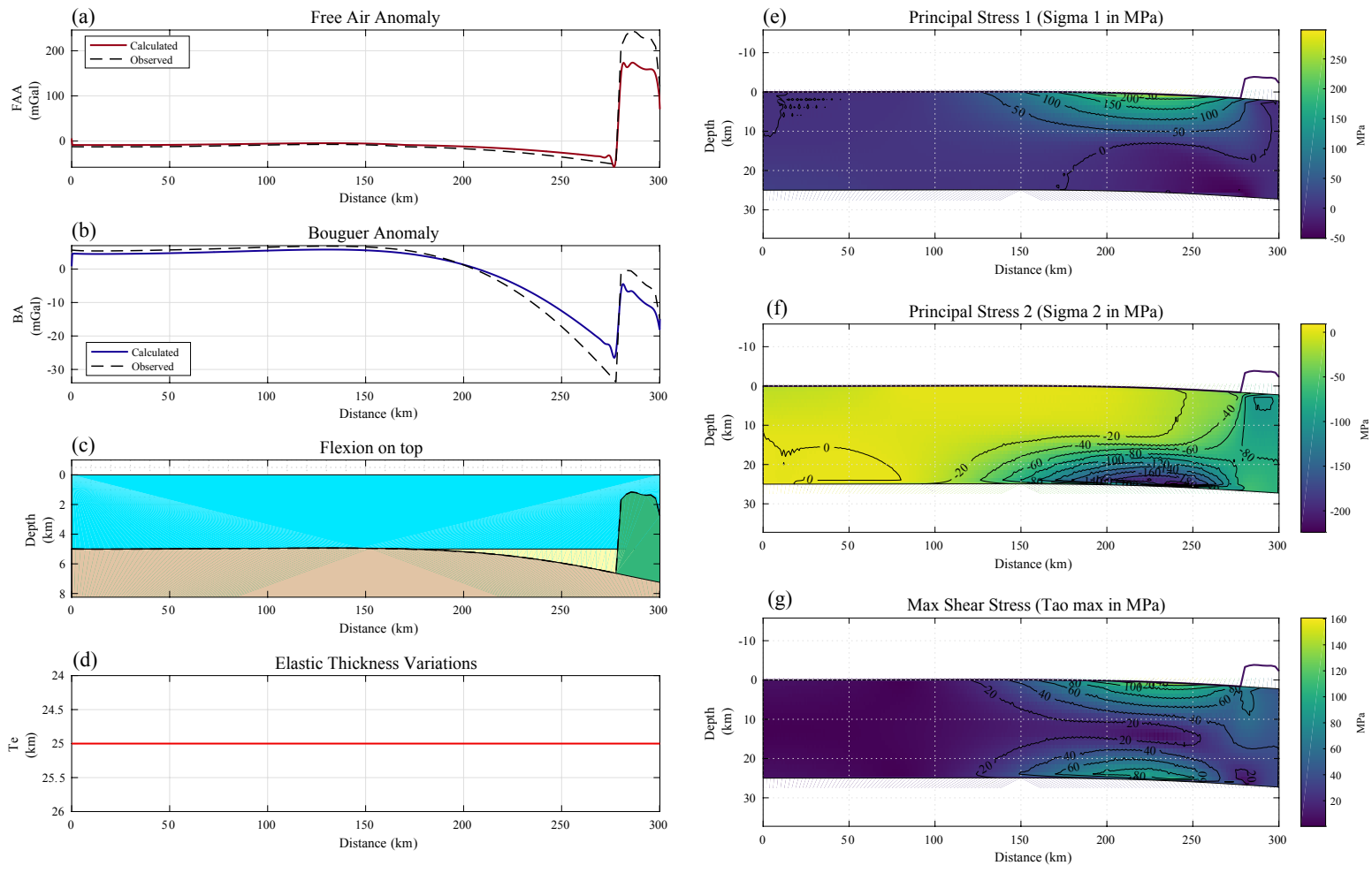

Fig. 6: Example 1

EXAMPLE 2: INLAND GRABEN WITH VARIABLE ELASTIC THICKNESS.

The second example shows the deformation of an originally 500 km plate with variable elastic thickness and an initial deformation. In this case, the deformation before the flexure takes the form of a graben that is filled by basalts (driving load), and that eventually fills up to 75% of its capacity with sediments. Elastic thickness was reduced at the center of the structure and extensive horizontal stresses are still affecting the plate at both ends (e.g. Brun, 1999; Thybo and Artemieva, 2013). The ends are treated with the "Horizontal Damp & Vertical Friction" to ensure that there cannot be free body displacement and to constrain the horizontal deformation. In this way, the system of loads deforms the plate by deepening the graben basin (this effect can be better appreciated in other figures that LIFFE produces) and by stretching the plate (note that the original length of the plate was 500 km and now it is longer). Like in the previous example, the computed gravity anomalies do not match the observed anomalies, therefore, the flexure must be larger. If the size of the basaltic layers is constrained, then the Te needs to be lower to fit the anomalies.

The stresses in the plate are characterized by high values of $\sigma 1$ and τmax on the bottom of the crustal section of the graben. These stresses in the base of the elastic section of the crust might be the response of crustal stretching/thinning associated with rift development that eventually leads to massive mafic intrusion/underplating (Thybo and Artemieva, 2013). Therefore, as the crust is weaker on the bottom, the magma intrudes it from bellow and from batholiths at the base of the crust.

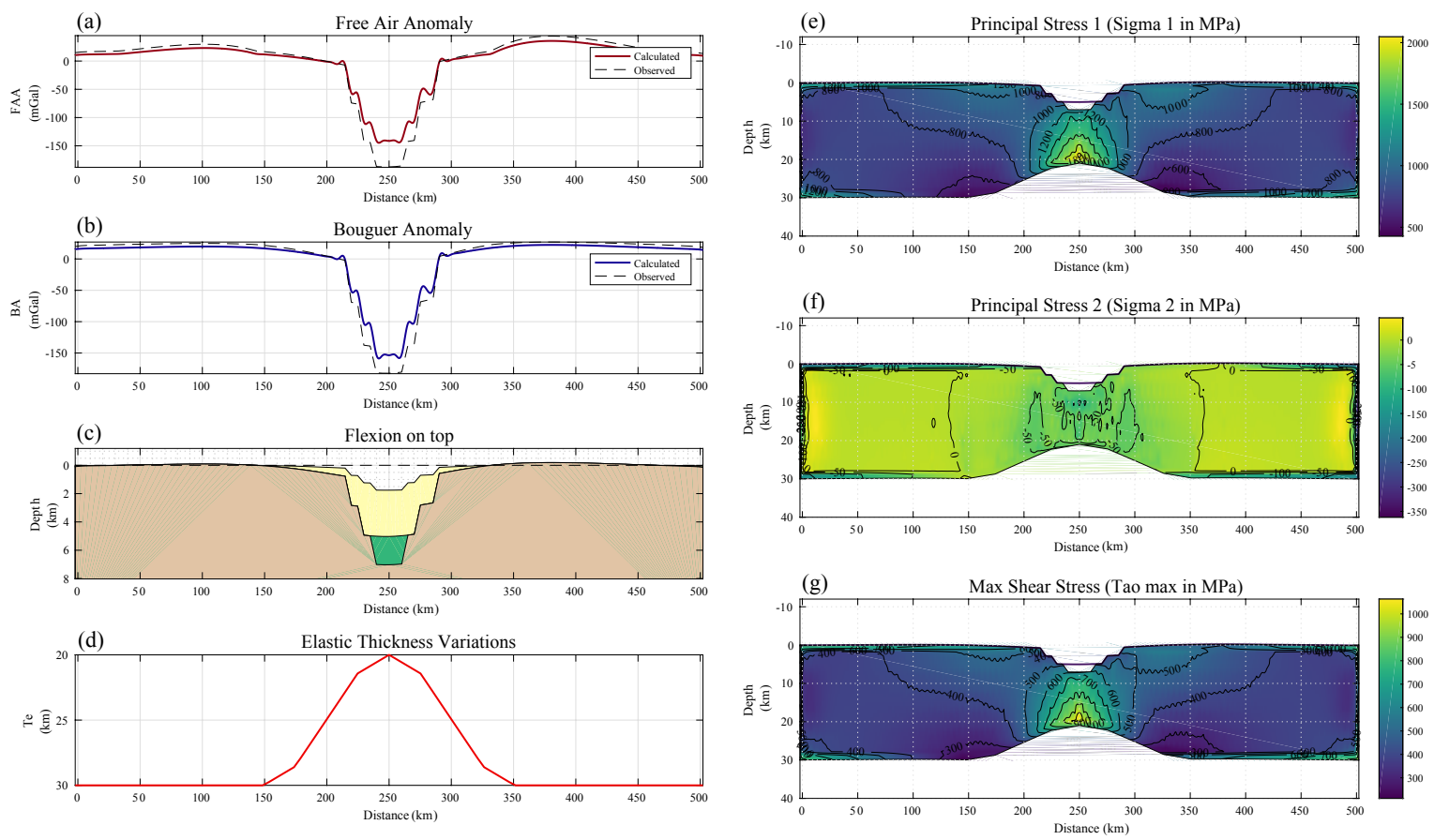

Fig. 7: Example 2

EXAMPLE 3: LONG TERM DEFORMATION OF THE CRUST.

Even though the elastic behaviour of a body is time invariant and time variant processes are better represented by viscous, viscoelastic and viscoelastoplastic rheologies, for long periods of times (>1 Ma) the uppermost lithosphere behaves almost elastically (Watts, 2001). In this sense, the final example that will be offered here is a simplistic way to model the long term response of the elastic section of the lithosphere to a growing vertical load on top of it (Fig. 8a), as well as an increasing compressional stress field (Fig. 8b). As the time passes by, the flexure changes dramatically (Fig. 8c and Fig. 8d). The growing load deepens and shortens the flexural basin, while the compression shortens the plate and further develops the flexural bulge. **The quantitative evolution of the features is consistent with the viscoelastic flexural model (e.g. Walcott, 1970) but it is not our intention to compare both approaches. This example shows how the Matlab functions of LIFFE can be called into a separate script and can be used to perform flexural modeling without the user being restricted.**

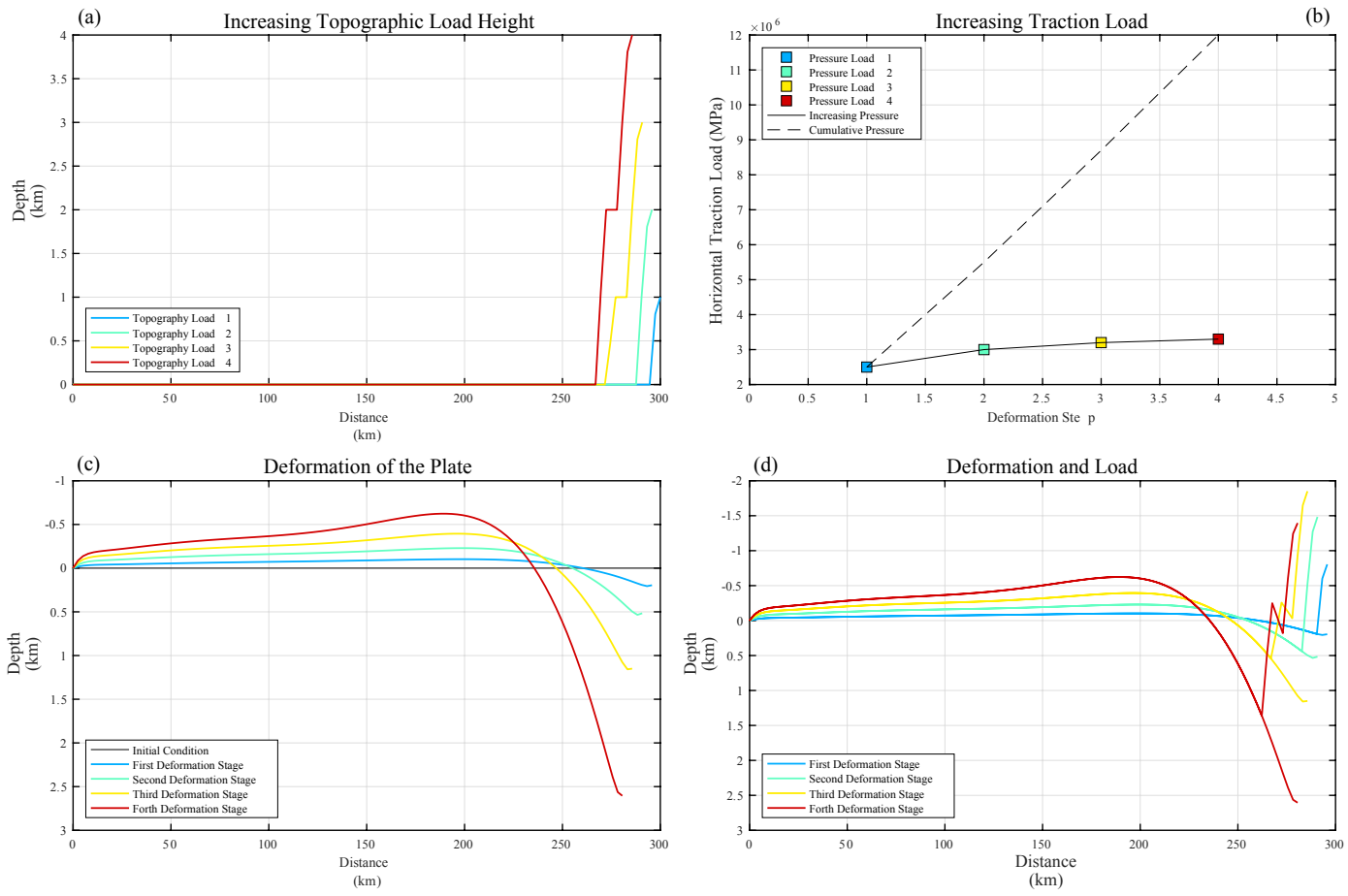

Fig. 8: Example 3

SOME TRICKS

You can produce a lot of results from the global variables that exist in the Matlab's Workspace after LIFFE has been used. Here there are some tricks that you might use.

READING THE REPORT

LIFFE shows a report in the command window as it is running. There you can find what the code is doing at each step, how many nodes are being loaded, check what restrictions were applied and some values about the flexure. If an error occurs, first change the size of the parameters that are input to *distmesh* and the loads file (LIFFE does not reorder any vector, so make sure that the distances are in the correct order all the time).

PLOTTING SECTIONS OF THE PLATE

To plot the top, bottom (like in the paper) or sides of the plate just use the vectors: nodesontop, nodesonbottom, nodesofleft or nodesonright. These hold the names of the nodes on each side of the plate (all the names are in vNodesNames). Also, every displacement is named vQ# (where # is the displacement) and the nodal positions are saved accordingly in vXnodes_Q# and vZnodes_Q#. To plot all the displacement that the top of the plate suffered during the computations you can do something like:

```
figure
plot(vXnodes(nodesontop),vZnodes(nodesontop))
hold on
plot(vXnodes_Q1(nodesontop),vZnodes_Q1(nodesontop))
plot(vXnodes_Q2(nodesontop),vZnodes_Q2(nodesontop))
plot(vXnodes_Q3(nodesontop),vZnodes_Q3(nodesontop))
set(gca,'Ydir','reverse')
```

You can plot the bottom by just switching nodesontop for nodesonbottom.

STRESSES

All the information of the final stresses is saved in the vector SIGMA_V_F. The columns of this vector are:

Xcentroid Zcentroid SigmaX SigmaZ TaoXZ Sigma1 Sigma2 TaoMax

FIGURES IN POSTCRIPT FORMAT

Matlab figures can be usually saved manually but if further editing is necessary, or if a vectors image is required, you may click o a figure and use:

```
print('Figure7','-dpsc','-bestfit','-painters')
```

This will print it as a color postcript.