grid_strain

Version 1.0 – March 2006

User's guide

Arianna Pesci
Istituto Nazionale di Geofisica e Vulcanologia,
Bologna, Italy
pesci@bo.ingv.it

Giordano Teza
University of Padua, Department of Geology,
Paleontology and Geophysics, Italy
giordano.teza@unipd.it

Table of contents

1. Introduction

`grid_strain.m` is a software, running in the Matlab™ environment, conceived for the easy ad quick calculation of the 2D deformation field (or the 2D deformation rate field) in a region, starting from the displacements (or the velocities) of a series of observational points adequately distributed in this region. For example, the input data can be provided by GPS measurements. The computed deformation pattern is presented as a set of the principal components of the strain calculated on a regular grid whose limits and passes can be chosen by the user.

The software, that consists of a set of functions called by the principal program (`grid_strain.m`), provides significance evaluations because a result related to an assigned grid point can be really considered only if the observational points are well distributed around it; in particular, three levels of significance are considered (high, mid and low).

The normal and simplest sequence to follow is:

1) data logging, visualization and management
2) grid creation and/or modification
3) definition and/or change of the smoothing factor *d0*
4) strain tensor estimation and significance test execution
5) tensors trace computation (calculations of arial deformation and maximum shear contour maps calculation).

The program `strain_zero` (computation of strain on a point without grid identification) is included in the package.

This software is free. The authors require that the use of this software be intended for scientific use only (no commercial use) and that, in the case in which an article whose results have been obtained using this software is accepted for the publication, the program `grid_strain.m` *and their authors are cited.*

The development of the 3D version of this package is in progress.

Requirements: this package runs under Matlab™ 6.5 or later (the development has been performed using Matlab™ 7.0). For any question, or also suggestion, please contact the authors:

2. Package installation and program running

The package `grid_strain.m` is contained in the directory grid_strain, compressed in the zip file grid_strain.zip. This directory can be extracted and saved in a Matlab directory, usually (but not necessarily) the work directory. For example, if the Matlab work directory is

C:\Programmi\MATLAB7\work,

the package can be saved and accessed with this path:

C:\Programmi\MATLAB7\work\grid_strain.

If Matlab operates in a Unix environment, consider "/" instead of "\".

In the case in which the directory grid_strain is saved in work directory (default directory on which operates the Matlab command window), before the program running this command must be entered in the command window:

> cd grid_strain

The program runs typing

> grid_strain

on this command window. All the necessary functions are automatically called by the principal program.

3. Data importation, visualization and management

When the program starts an iterative window appears and the user can choice to load a new data file or to import previously saved data. The first step is to click the "new calculation of both grid and strain" button and to type the correct file name (figure 1), that must have ASCII format. This file should be contain 10 columns, that represent the following data (in this order):

1. *ns* (station identifier);
2. *e* (east coordinate, expressed in m);
3. *n* (north coordinate, m);
4. *ve* (east velocity, or east displacement, expressed in mm);
5. *vn* (north velocity, or north displacement, mm);
6. *eve* (rms error on *ve*, mm);
7. *evn* (rms error on *vn*, mm);
8. *a* (length of major half-axis of error ellipse, mm);
9. *b* (lenght of minor half-axis of error ellipse, mm);
10. *theta* (azimuth of major axis of error ellipse, degrees).

The last three columns are optional: in the case in which the file contains 7 columns only, the error ellipses are assumed to have the axes parallel to east and north direction respectively, with half-axes length simply expressed by *eve* and *evn* (nevertheless, a warning message is provided by a menu box). If the columns number is smaller than 7, the data load is not performed. Menu and dialog box allows the choice of other filenames. Although the coordinates are conceived as expressed in m and the displacements in mm, a check is performed in order to detect if the fact that displacements are instead expressed in m is suspected. If it happens, a menu box in presented for the confirm or negation of this suspect (the user is always guided in each choice: the package is user friendly).

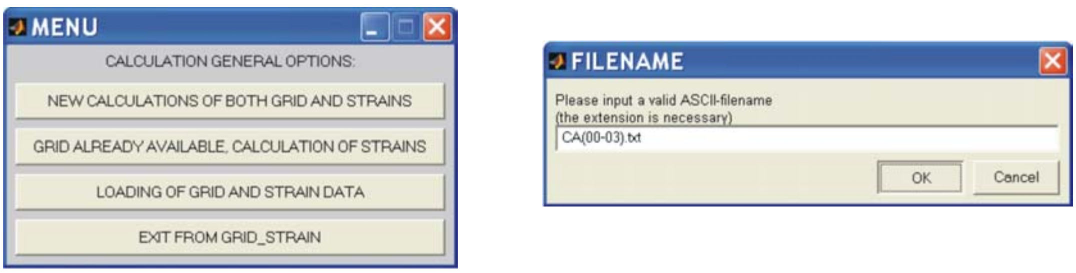

Fig.1. Left: The first menu interface. Right: Dialog box for ASCII filename inserting.

Now, a menu box appears and presents the general saving options (fig. 2). In this way, the user can choose one of the following options:

- Data saving with a common part of the names of the generated files. In this case, the filenames are automatically generated. When a series of data should be saved, and the corresponding file does not exist, the saving is performed. Instead, if a file with the same name exists, another menu box appears (fig. 3) and the user can choose among overwriting, saving with another filename and no saving.
- Data saving with filenames interactively asked (in the case in which a file with the same name exists, the function operates as above).
- No saving. In this case, the saving function is always excluded.

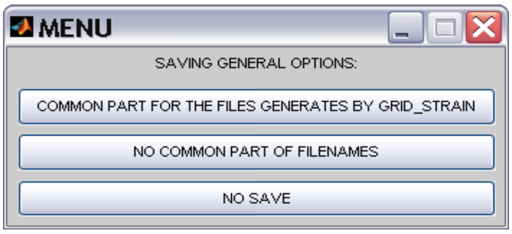

Fig. 2. General options for the data saving

Fig. 3. Menu box for the management of a case in which a filename inserted of automatically generated is the same of an existing file

In the case in which a common part 'compart' is chosen, the filename automatically generated are the following:

compart_2D_grid.mat: 2D regular grid;
compart_2D_strain.mat: strains.

After the general choice of the saving options, the displacement vectors are immediately plotted together with their error ellipses (fig. 2)

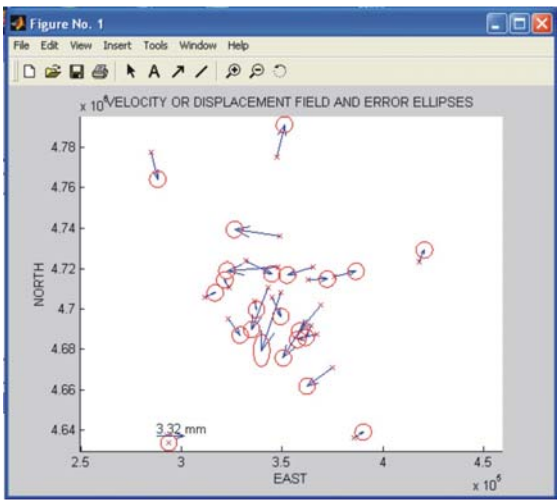

Fig. 4 - Displacements (or velocities) and their error ellipses

Once data are displayed, a menu appears (fig. 5, left) and the user can to exclude one or more points from the strain computation simply clicking the "yes" button on this menu and using the pointer on a mat-figure that is a replica of previous one, but black and white colored, without the error ellipses (fig. 5, right). This operation can be executed more times (until at least a data point is available).

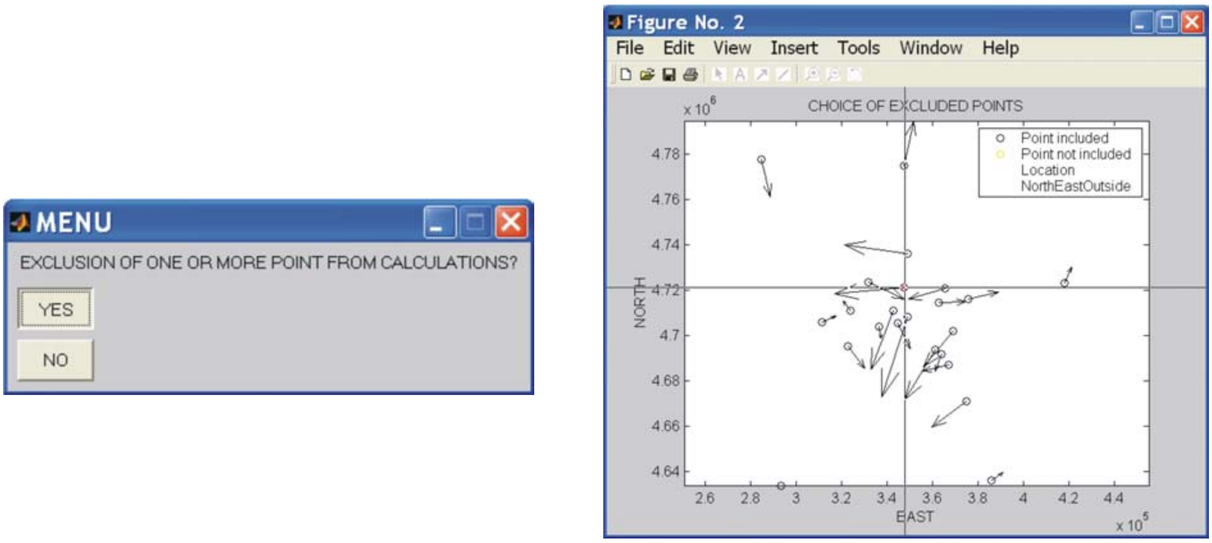

Fig. 5 – interactive exclusion of one or more points form the computations

4. Grid creation and/or modification

The program automatically computes the grid length based on station baselines. The standard deviation of all the inter-distances between point pairs is computed and proposed as default value. A first grid is shown together a menu that reports the data about the grid (fig. 6). Together the grid, both the accepted and refused points in the previous step are presented. The user can accept or refuse the grid acting with this menu command.

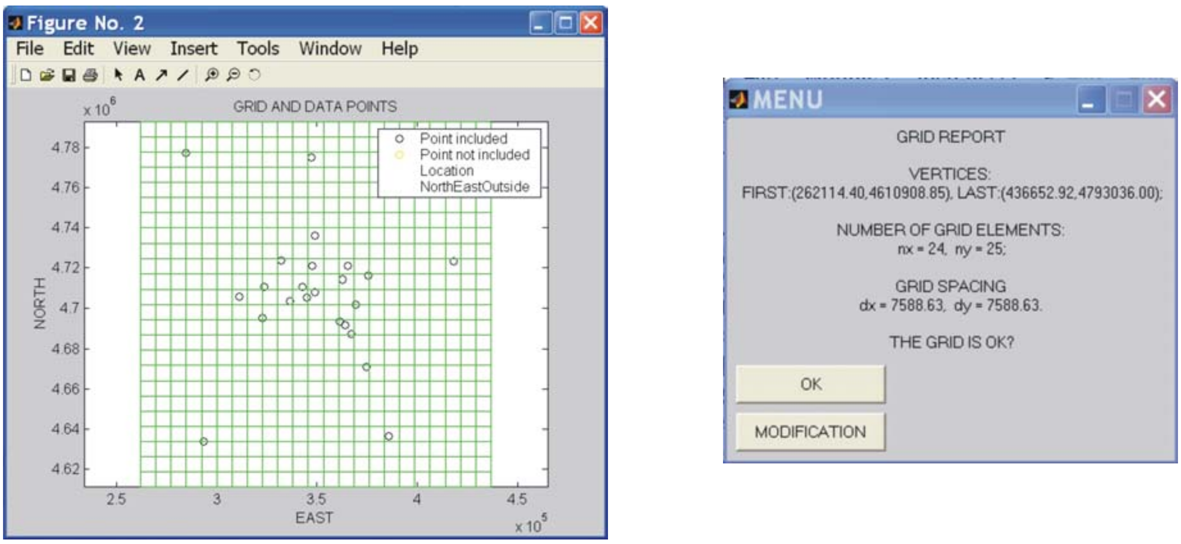

Fig. 6. Left: automatic grid definition: a 7-km spacing was computed. Right: menu for acceptance or modification of the grid.

If the grid is not accepted, both the grid spacing and grid dimension can be modified inserting manually more appropriate values into the blank spaces in an apposite dialog box (fig. 7). Moreover, grid vertices can be changed iteratively on the figure using the pointer to select the lower left and the upper right corners.

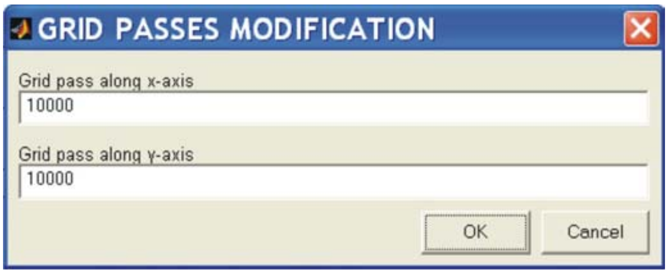

Fig. 7 – Manual change of grid passes

When the grid is the desired, and the data saving is not excluded, the grid data are saved in a Matlab .mat file.

Among the initial options (fig. 1), the user can choice the option "Grid already available, calculation of strains". In this case, the name of file that contains the grid parameters, generated in previous session of grid_strain, is asked by means of a dialog box. Note that the user, in this case, can to remove another measurement points (with the same interactive procedure shown in fig. 5), as well as to reintroduce one or more points that were excluded in the previous session. Once that the grid configuration is accepted, the program execution is the same for both the first two options.

5. Definition and/or change of the smoothing factor

Once the grid is corrected based on user preferences, the program automatically define the smoothing parameter, or scale factor, for the weighting (Shen, 2000). This default value is assumed as three time the grid spacing, but the user can modify it manually inserting a preferred value using a dialog box (fig. 8). The scale factor characterize the strain calculation. In particular, if data points are widely dense distributed, the local strain can be estimated at each node of the grid (or also in a defined point) using a weighting strategy to automatically lower contribution of distant stations from the node. All available data are involved in computation but errors are rescaled using an appropriate function which increase with distance. Following Shen et al. (2000), the weigh function $e^{-d/d0}$ is used (here *d* is the distance between the node and the grid point and *d0* is the smoothing parameter).

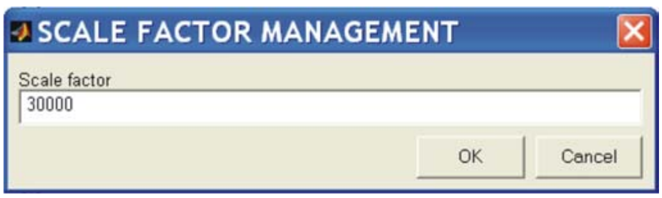

Fig. 8 - Manual definition of scale factor for weighting (default: three times the grid spacing)

6. Strain computation on the grid nodes and significance evaluation

Once the grid is available (generated in the present on in a previous program session) and the scale factor is available, the program starts the strain rate estimation over each node of the grid. The computational function to solve for redundant equation system and estimate the velocity gradient tensor follows the standard linear least square approach but the weight function is introduced to error scaling. The results of the strain computation on all the grid points are presented immediately after the conclusion of the iterations. The eigenvectors are presented with their directions and two colors: if an eigenvalue is positive (extension) the color is blue, whereas the color is red in the case in which the eigenvalue is negative (compression). Figure 9 presents these results.

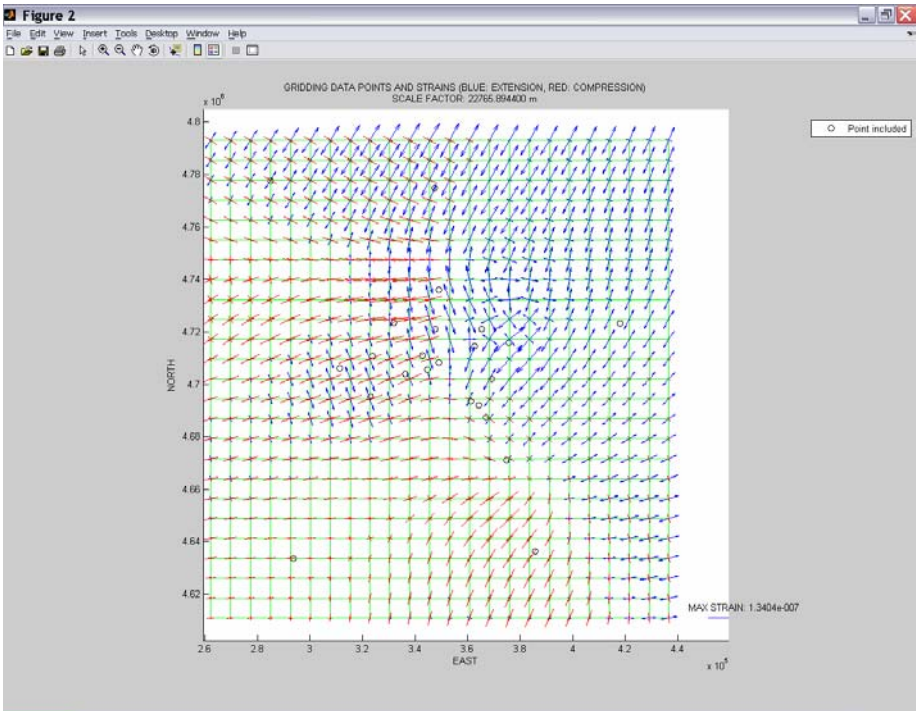

Fig. 9 – Results of the strain computation on all the grid points

In addition, a significance a significance functions detects the points in which the can be considered really representative of a local strain at the selected scale and generates the corresponding flags. In particular, to each grid point is associated a flag that indicates if:

The grid point has high significance. In this case, in all the three 120°-width regions in which the plane can be subdivided, is present at least a data point whose distance between it and the considered data point is lower than (or equal to) the scale factor. The spatial distribution of the data points around the grid point is in this case in good.

The grid point has mid significance. In this case, in two of the three 120°-width regions in which the plane can be subdivided, is present at least a data point whose distance between it and the considered data point is lower than (or equal to) the scale factor. The spatial distribution of the data points around the grid point is in this case near to validity, but the estimate value of the strain is not completely representative of the compressional and/or tensional state of the area (insufficient data points, but situation not completely bad).

The grid point has low significance. In this case, the result cannot be considered.

The menu box represented in fig. 10 appears and enable the user to chose the desired representation

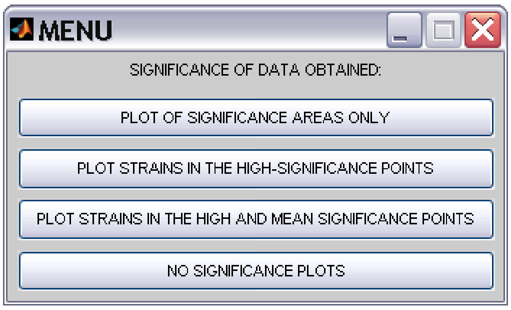

Fig. 10 – Menu box for the representation of the strain vectors on the basis of the significance of results

Figure 11 represents the plots both on the high and high and mean significance points.

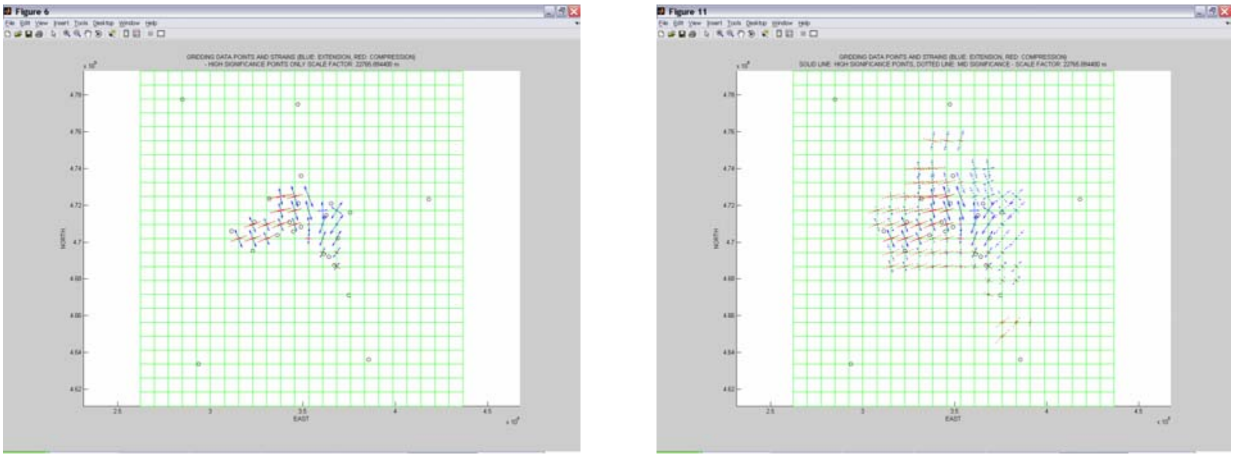

Fig. 11 – Strains on high significance points only (left) and in high and mid significance points (right). High significance eigenvectors are solid, the other dotted

The program allows, in this stage, the interactive selection of one or more points and the visualization of the corresponding strain. The results of this selection are contemporarily shown in a menu box and in the command window (note that no command are typed using this command window after the program initialization; it is used for visualization only). An example of data shown by menu box is presented in fig. 12.

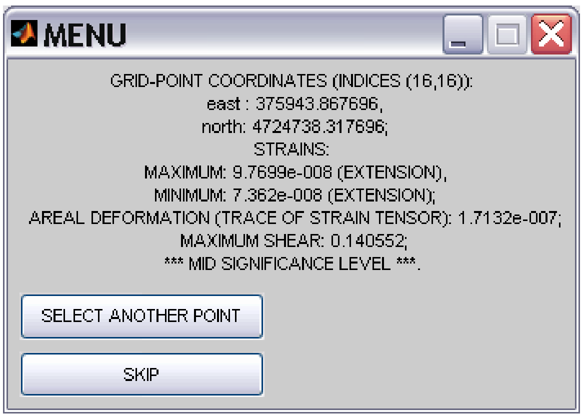

Fig. 12 – An example of the strain data provided by a specified grid point (this point is selected on the figure representing all the data, see fig. 9)

7. Areal deformation and maximum shear patterns

Once the strain are available, the strain tensor trace is immediately computed for each grid point; it is $e_{max} + e_{min}$, in which e_{max} and e_{min} are the maximum and minimum eigenvalue respectively. These traces represent the areal deformations and are shown in a contour plot (fig. 13). Finally, a maximum shear contour map is generated (fig. 14); it presents the values $(e_{max} - e_{min})/(e_{max} + e_{min})$; note that values near zero belong to strain tensors which component are comparable in modulus, while values near +/-1 are characterized by the predominance of one component on the other.

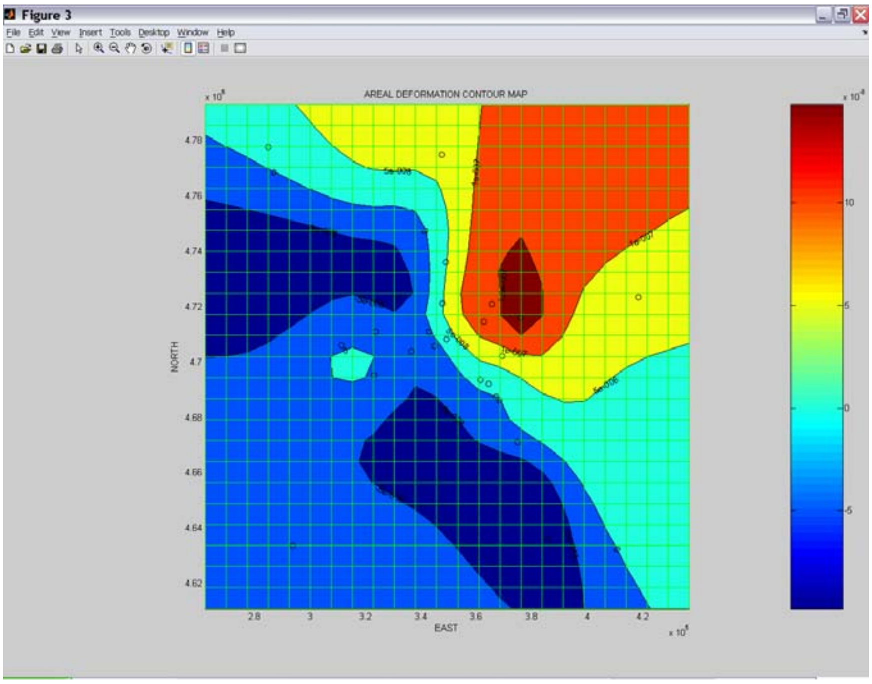

Fig. 13 - Arial deformation contour map

All the calculation of strain can be repeated considering other scale factors; the Matlab figures are maintained in order to comparison the results obtained considering different scales. This fact should be taken in account if the grid presents a large number of points and the strain calculation are repeated many times.

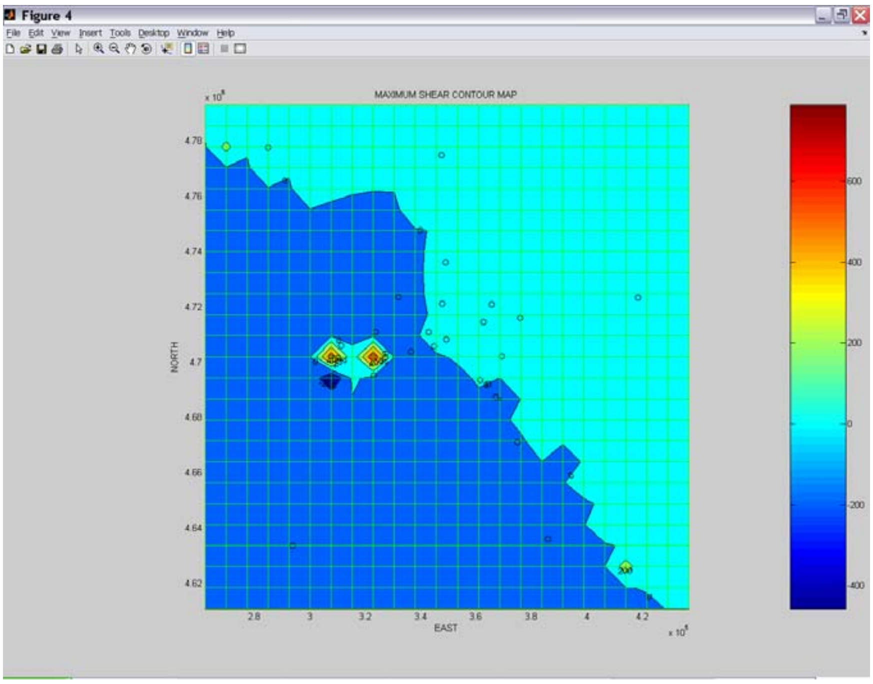

Fig. 14 – Maximum shear contour map

8. The `strain_zero` program

It is a simplified version of `grid_strain`. Assigned a data points file as in the case of `grid_strain`, it calculates the strain in a selected point on the figure. The data provided are as in the case of `grid_strain`, but the generation of grid and the iteration on it are not preformed. In addition, the user can choose a scale factor or exclude it (in the latter case, an infinite scale factor is considered).
This program operates like `grid_strain`; for this reasons, no further information is provided here.