A Tutorial to the Sparse Programs: How to Obtain a Continuous Strain Rate and Velocity Field from Earthquake Data and Geodetic Velocities

version 3.0

**By Corné Kreemer, Bill Holt
and John Haines**

October, 2001, Stony Brook

****
*   *

<u>Disclaimer</u>: None of the programs can be used without proper acknowledgement to Dr. John Haines, the author of the codes, and without accurate reference to *Haines and Holt* (1993), *Holt and Haines* (1995) and *Haines et al.* (1998). References are given in section 8. When extensive help and guidance is needed, an official collaboration is appreciated.

*   *
****

Contents

1. Introduction

This tutorial is written with the purpose of providing a step-by-step explanation of the computer programs that allow you to determine estimates of continuous horizontal velocity and strain rate fields, inferred from seismic, geodetic and geologic data. This procedure was first described by *Haines and Holt* [1993] and later improved by using bi-cubic splines rather than polynomials to interpolate strain rates and velocities [*Holt and Haines*, 1995]. Extensive description and use of this method and its underlying theory can be find in various papers [e.g., *Holt and Haines*, 1995; *Shen-Tu et al.*, 1995,1998; *Haines et al.*, 1998; *Holt et al.*, 2000; *Beavan and Haines*, 2001]. In Appendix F you can find an overview of the methodology as well. This overview is written by Sarah Jenny, and is based on the hard to get publication of *Haines et al.* [1998]. Recently the *fortran* programs have been rewritten, such that they now use sparse matrix implementation. This increased both the speed of the programs and the potential size of the grid you are using tremendously.

Here, we like to discuss step-by-step what needs to be done to go from observations to model results. We will discuss the different kinds of observations and model constraints allowed in the procedure, and illustrate this where possible with real data and results, as they are published in recent papers. This tutorial has benefited a lot by feedback we got from Sarah Jenny, whom we are very grateful. For now, this tutorial will cover in detail;
• Inversion of seismic strain rates
• Inversion of GPS data
• Plate boundary constraints

E-mail questions or comments to Corné Kreemer; kreemer@mantle.geo.sunysb.edu

2. The Programs

The programs can be found on our ftp site at horizon.ess.sunysb.edu. Log in as anonymous and give your e-mail as password. Change directory to /pub/sparse. All the programs referred to in this tutorial can be found in this directory. Example files (input and output) and this tutorial (in postscript and *word* format) can be found in the directory called *EXAMPLES*. This directory has two subdirectories; *seismic* and *GPS*. These contain the same example files that are mentioned in the tutorial In this tutorial most in- and output files are given, but they are often truncated and in some cases they are shown incorrect, because in reality lines in files can be long. Therefore always look at the original files. Examples come from *Kreemer et al.*, [2000].

3. The Inversion of Seismic Strain Rates

We start with an example of taking a data set of earthquake moment tensors from the Eastern Indonesia and Philippines region to obtain estimates of the seismic velocity and strain rate field. Input files are identical as those used in *Kreemer et al.*, [1998] and the results given in this tutorial also come from that same paper.

Step 1; Create ***geometry.dat***.

A description of this file is given below Appendix A. Copy ***geometry.dat*** also to a file with a distinctive file because ***geometry.dat*** will be overwritten in **step 2**.

cp *geometry.dat geometry.old* (be sure to back up!)

Step 2; Run **convert_to_sparse_grid**.

The ***geometry.dat*** file that you have created in **step 1** needs to be converted such that it can be used in the rest of the programs. That is because recently these programs have been changed to use sparse matrix methodology.**.** This will create a file called ***sparse_geometry.dat***. It is also possible to create a ***sparse_geometry.dat*** yourself from the start, but you may find it more illustrative to follow the procedure outlined here.

cp *sparse_geometry.dat geometry.dat*

Step 3; Run **make_sparse_geometry**

Your ***geometry.dat*** file is converted in order to be used in the following programs using the sparse matrix methodology.

Step 4; Run **make_raw_spline_fit_dat**

This creates the raw ***spline_fit.dat*** which at this moment contains zero values for data and (co)variances The final file (later on) must have non-zero values for (co)variances. The file looks as follows:

```
 16          17          120
119
120           0            0            0
209

  1          11            2
```

```
           0           0           0
           0           0           0           0           0           0

           2          12           2
           0           0           0
           0           0           0           0           0           0
.....etc.
```

The first line returns your input from ***geometry.dat.*** The second line shows the number of free (i.e., unconstrained) rotation values (or knotpoints). The third line contains the rotation number of the reference frame followed by 3 zeroes indicating that no rotation is applied to this frame. The next line tells you that there are 209 grid cells in this file. This is followed by these 209 cells, which all have three zeroes for the three data components (i.e. values for the average strain rates in that cell) and six zeroes for the average standard errors and related correlation coefficients. Save this file to ***raw_spline_fit.dat*** for later use;

cp *spline_fit.dat raw_spline_fit.dat*

<u>Step 5</u>; Run **prepare_rectangle_boundaries**

This creates the file ***boundaries.dat***, which contains the boundary locations (33 for each area) around each grid area in the regular grid space (x, y).

cp *boundaries.dat output.dat*

<u>Step 6</u>; Run **make_x_for_output**

This will map the points in ***output.dat*** from (x, y) to (lat, long), and this will be used by the program that searches for events within each area. The following question will be asked:

```
 The time is  0.
 Enter the latitude and longitude (in degrees) of the Euler
  pole for the frame of reference and the rotation to
   be removed
```

The question allows one to rotate the reference frame about a pole of rotation. 0 0 0 it keeps the original frame of reference. This program created two files:

- *x_for_output.log*: contains the latitude and longitude values of the 33 points making up each grid area. The number of lines in this file is thus 33 times the number of grid areas (209).

- *x_for_output.dat*.: binary format of ***x_for_output.log***

cp *x_for_output.log boundaries.log*

It is important to back-up this file (in particular if you want to plot the grid) because **make_x_for_output** *will be used later to for different purposes.*

<u>**Step 7**</u>; Run **make_x_for_fit**

This calculates the area within each grid area, which will be used in the Kostrov summation. The program asks the following question again;

```
The time is  0.
 Enter the latitude and longitude (in degrees) of the Euler
  pole for the frame of reference and the rotation to
   be removed
```

We want to keep the same reference frame, so we give `0 0 0` as input. This program has created a file ***x_for_fit.log***, which contains the areas, normalized by the earth radius squared.

<u>**Step 8**</u>; Run **make_smooth_input_from_earthquakes**

This performs the Kostrov Summation. This program can also smooth the data and the variances. The program reads in ***earthquakes.dat***, which contains the moment tensor. See Appendix B for the format.
The programs gives the following screen output:

```
 The number of earthquakes in earthquakes.dat is  9539
  and the (adjusted) size of the largest earthquake is    1.024352585099486E+20
 Enter (a) the length of time the earthquakes span and
  (b) and (c) the shear modulus and thickness of the
   seismogenic layer
```

This information asked for is directly used in the Kostrov summation.

- Timpesan: a 21.42 years catalog will need to be inputted as 21.42*10^{-9}. *The value is normalized by 10^{9} year, because this is our chosen time unit of strain rate (e.g., 10^{-9} yr^{-1}).*
- Shear modulus: input is in Newton/meters^2. We choose 3.5x1010 N/m^2
- Thickness of the seismogenic layer: input is in meters. We choose `3.0e4 m`

```
Enter the radius of the earth in the units you are using
```

This should be given in meters: 6.371e6

```
 Enter the number of times the rate-of-strain values are to
  be smoothed between adjacent rectangles - each time the
   smoothing is performed 0.5 is added to the square of the
    smoothing kernel's RMS radius, measured in number of
      rectangle sides, so that after smoothing nsmooth
       times this radius is sqrt(0.5*nsmooth)
```

Here you can smooth the data. The value corresponds to a certain grid radius, which is the square root of half of your value. For now we choose the data to be smoothed once.

```
Specify the base level of the rate-of-strain variances, to
  which contributions from the earthquakes are added, by
   entering a quantity equivalent to squaring the result of
    dividing a velocity by the earth's radius. The velocity
     in question is the jump in velocity in any region that
      could be unaccounted for either across a fault of
       length equal to the square root of the area of the
        region or across a series of faults amounting to the
         same thing
```

Here you can input the incompleteness factor or missing earthquake, see Appendix F. We choose `0.02.` In Appendix C we will see that `0.02` will correspond to a velocity jump of 0.9 mm/yr across each area or a Mw=…'missing earthquake for each area;

```
Enter the number of times the variance-covariance data is
  to be smoothed between adjacent rectangles - each time the
   smoothing is performed 0.5 is added to the square of the
    smoothing kernel's RMS radius, measured in number of
      rectangle sides, so that after smoothing nsmooth
       times this radius is sqrt(0.5*nsmooth)
```

The number used here for smoothing of the variance-covariance values is 32. This corresponds to a smoothing kernel RMS radius of 4 rectangles (remember; the square root of half your value).

It will also be instructive to try solutions with zero smoothing of variances and covariances, followed by solutions in which the variances and covariances are smoothed over 1, 2, and 8. Also experiment with the amount of smoothing of the seismic strain rates. Examine for each case both the fit between the model strain rates and the seismic strain rates and the uncertainties in the model velocity field.

The ***spline_fit.dat*** file is created and the has data values are in units of 10^{-9} year^{-1}:

cp *spline_fit.dat spline_fit_1_32.dat*

<u>**Step 9**</u>; Run **sparse_fit**

Here the inversion is performed.

```
Do you want the data to be smoothed (Y/N)?
```

We choose Y. *Be aware that this is a separate procedure from the smoothing option in* **make_smooth_input_from_earthquakes**.

```
  Loading rectangles
  Starting DECOMP
  Starting INV
  Starting INV_T
  Outputting residuals
  Sum of squares is 21.8344325979664
  So you want to save the results (Y/N)?
```

Y ; yes we do.

```
Do you want variances and covariances (Y/N)?
  If not, the output file spline_fit.out can
   be used in all the same programs as before.
    The same applies if you enter "Y" here,
     provided you respond "N" when asked this
      question again in the old programs. As
       a rule, the calculation of variances
        and covariances will be tedious for
         for very large problems.
```

Y; in reality, for most grid sizes, calculation of the variances and covariances will only take a short time; not more than 10 minutes.

The inversion program has produced the file ***spline_fit.out***, which has the fitted velocity gradient tensor parameters in it (binary).

<u>**Step 10**</u>; Run **sparse_average_strain**

Here the the horizontal components of the average strain rate tensor for each grid area are calculated. If you want to calculate reuslts for specific locations, go to <u>**step 12**</u>.

```
Do you want the data to be smoothed (Y/N)?
N

 Number of independent rotations =         120
  Upper bound on RMS velocity =  0.000000000000000E+000
   Number of rectangles =         209
 Do you want to continue (Y/N)?
Y

 Do you want the variance-covariance matrices (Y/N)?
N
```

If you choose Y, be ready for long calculationn time!

```
 The strain values are separated into those for each quarter
```

```
  box. Do you want them combined into a single value for
   each box (Y/N)?
Y
```

This produces an output file called ***average_strain.out***. The first set of values for each grid area give the observed average for each area, ie., the (smoothed) data obtained from Kostrov's summations in **make_smooth_input_from_earthquakes**. The second set of values correspond to the strain rates calculated within the inversion as average for each grid cell.

- First line: no., lat and long (*corresponding to midpoint of grid*) Elong, Elat, Exy (*all observed*), Elong,Elat, Exy (*all calculated*).
- Second line: corresponding standard errors in average strain rate (*a priori estimates for observed values and posteriori estimates from calculated values*).
- Third line: correlation coefficients: ρ(exx,exy), ρ(exx,exy), ρ(eyy,exy) (*observed*), and ρ(exx,exy), ρ(exx,exy), ρ(eyy,exy) (*calculated*).

Step 11; Run **general_spline_principal_average_strains**

As an alternative to **sparse_average_strain** this program calculates the horizontal principal axes of the model strain rate field for each grid area. The output file ***principal_average_strains.out*** contains:

- no., lat, long, X-Azimuth (*azimuth of maximum axis*), Exx, Eyy *(Exx is the maximum axis and Eyy the minimum axis*) (*all for observed*), X-Azimuth, Exx, Eyy (calculated).

Step 12; Run **prepare_midpoints** or **prepare_knotpoints** or **convert_lat_long**

Besides calculating average (principal) strain rate values for each grid area, there is an options to calculate velocities, (principal) strain at specific locations. There are three options; at grid midpoints, at grid knotpoints, and any point within grid. For calculating results at knotpoints you have to **run prepare_knotpoints**, and for calculating results at midpoints you have to **run prepare_midpoints**. Afterwards copy output to ***output.dat***.

cp ***knotpoints.dat/midpoints.dat output.dat***.

If you want to calculate strain rates or velocities for any location (for example on a regular grid), you need to run **convert_lat_long**.

```
 Enter the latitude and longitude (in degrees) of the Euler
  pole for the frame of reference and the rotation rate to
   be removed
0 0 0

 Enter ntest the square root of the total number of test
  points that will be used from each rectangle in the
```

```
   curvilinear grid (N.B. nxtest = ntest and nytest = ntest)
5
```

Don't give a too high number because this takes some significant computer time

```
 Enter the name of the file containing the latitude and
  longitude values in the following format. The first line
   has the number of points. Then each successive line
    contains the number of the point and the latitude value
     and longitude value (degrees)
```

Give here the file with your latitudes and longitudes This file, that can have any name, could look like;

```
48
1 13.56 124.34
2 -4.04 138.95
3 06.97 122.07
4 -3.45 114.75          etc….
```

cp *lat_long.dat output.dat*

__Step 13__; Run **make_x_for_output**

This will map the coordinates properly on the grid

```
The time is  0.
 Enter the latitude and longitude (in degrees) of the Euler
  pole for the frame of reference and the rotation to
   be removed
```

__Step 14__; Run **sparse_velocity**

```
Number of independent rotations =         120
  Upper bound on RMS velocity =  0.000000000000000E+000
   Number of rectangles =         209
 Do you want to continue (Y/N)?
Y
 Do you want the variance-covariance matrices (Y/N)?
Y
 Enter the latidude and longitude (in degrees) of the Euler
  pole for the frame of reference and the rotation rate to
   be removed
0 0 0
```

The program produces a velocity file called ***velocity.out*** (in the example case calculated at the knotpoints). The output should be straight forward. Units of velocity are in the strain units of 10^9 yr^{-1}

(because they are normalized by the earth radius). Thus, the velocity can be converted to velocity in mm/yr by multiplying by earth radius in mm. In other words, multiply velocities with 6.371 to obtain mm/yr.
Velocity components are followed by Vrot, which is the rotation rate ($\dot{\omega}$) of each point (in radians/yr in units of 10^{-9} yr^{-1}. Next in line are the latitude, longitude and rotation rate (same units as other $\dot{\omega}$) of the angular velocity that best describes the velocity in your current reference frame.
The second half of *velocity.out* contains the covariance matrices. Take the square root of the varainces and multiply with 6.371 to obtain model standard errors in mm/yr.

Step 15; Run **sparse_strain**

This programs calculates horizontal components of the model strain rate tensor at the selected location.

```
Number of independent rotations =         120
  Upper bound on RMS velocity =  0.000000000000000E+000
   Number of rectangles =         209
 Do you want to continue (Y/N)?
Y
 Do you want the variance-covariance matrices (Y/N)?
Y
```

Output file is ***strain.out***. Format is equal to ***average_strain.out***, except that observed values are not given.

Step 16; Run **sparse_principal_strains**

This programs calculates horizontal components of the model strain rate tensor at the selected location.
Output file is *princpal_**strains.out***. Format is equal to *principal_**average_strains.out***, except that observed values are not given

Summary Seismic Strain Rate Calculations

The following commands should be executed to do a standard process with a velocity solution, average strain rate solution and corresponding principal strain rate solution;

```
cp geometry.dat geometry.old

convert_to_sparse_geometry
cp sparse_geometry.dat geometry.dat

make_sparse_geometry

make_raw_spline_fit_dat
cp spline_fit.dat raw_spline_fit.dat

prepare_rectangle_boundaries
cp boundaries.dat output.dat

make_x_for_output
cp x_for_output.log boundaries.log

make_x_for_fit

make_smooth_input_from_earthquakes
cp spline_fit.dat spline_fit_1_32.dat

sparse_fit

sparse_average_strain

general_spline_principal_average_strains

prepare_knotpoints/prepare_midpoints
cp knotpoints.dat/midpoints output.dat make_x_for_output

sparse_velocity
```

4. The Inversion of GPS Data

There are several ways to invert for GPS data (or any other observed velocities). Discussed in section 4.1 is a procedure to invert for GPS vectors together with seismic strain rates. Alternatively, you could invert for just the GPS velocities with a variety of possibilities on the, a priori, assumed distribution of ‘strength’ and ‘properties’ of the region covered by the grid. That is, first of all whether it is uniform or non-uniform in ‘strength’, and second whether this involves isotropic deformation or whether you want to constrain the style and direction of the model strain rate field (where in this case constrains on style and direction are inferred from by the seismic strain rate field). These latter cases are discussed in section 4.2.

4.1 Inversion of Seismic Data with GPS Observations Constraints

This procedure involves only two alterations of the inversion procedure described in the previous sections. However, for the actual inversion we do not run **sparse_fit** but **sparse_fit_with_GPS**. However, before we can perform the inversion we need to map the latitude and longitude of the GPS stations on the grid.

run **convert_lat_long**

```
 Enter the latitude and longitude (in degrees) of the Euler
  pole for the frame of reference and the rotation rate to
   be removed
0 0 0

 Enter ntest the square root of the total number of test
  points that will be used from each rectangle in the
   curvilinear grid (N.B. nxtest = ntest and nytest = ntest)
5
```

Don't give a too high number because this takes some significant computer time

```
 Enter the name of the file containing the latitude and
  longitude values in the following format. The first line
   has the number of points. Then each successive line
    contains the number of the point and the latitude value
     and longitude value (degrees)
```

This file, that can have any name, looks like;

```
93
   1 -35.40  148.98
   2 -42.80  147.44
   3 -31.80  115.89
   4 -29.05  115.35.....etc.
```

cp *lat_long.dat output.dat*

run **make_x_for_output**

Next, we need to create a ***GPS.dat*** file that corresponds to the file which contained the site locations. Appendix E contains a description of *GPS.dat*. It is now time to run the inversion.

run **sparse_fit_with_GPS**

```
Do you want the data to be smoothed (Y/N)?
Y
  Loading rectangles
  Loading GPS
  Starting DECOMP
  Starting INV
  Starting INV_T
  Outputting residuals
  Sum of squares is 490.807033507879
  So you want to save the results (Y/N)?
Y
Do you want variances and covariances (Y/N)?
  If not, the output file spline_fit.out can
   be used in all the same programs as before.
    The same applies if you enter "Y" here,
     provided you respond "N" when asked this
      question again in the old programs. As
       a rule, the calculation of variances
        and covariances will be tedious for
         for very large problems.
Y
```

To summarize, in subsequent order you need to do;

convert_lat_long
cp *lat_long.dat output.dat*

make_x_for_output

sparse_fit_with_GPS

That is, of course, besides all the other things you need to do, which are described in chapter 3.

4.2 Inversion of GPS Data

A combined inversion of seismic and geodetic data may not give you a very desirable result, because of the different nature of both datasets. Therefore, you might want to opt to do an inversion of only the geodetic data only, with various degrees of constraints. Such an approach differs in one way from the previous example and that is that now you have to create your own ***spline_fit.dat*** before performing the inversion. This file contains the observed strain rate data and its variances and covariances. Earlier this file was created by make_smooth_input_from_earthquakes which applied Kostrov summation of the focal mechanisms. Now, we have to make it ourselves. However, for the cases discussed here we won't have any observed strain rates and we will only limit ourselves to constraining the a priori variances and covariances. Below, we describe the different options how to create ***spline_fit.dat*** based on different constraints.

First, we need to assign an isotropic variance to ***spline_fit.dat***; a so-called base level.

Run **perform_smoothing_on_input**

```
 Enter the number of times the rate-of-strain values are to
  be smoothed between adjacent rectangles - each time the
   smoothing is performed 0.5 is added to the square of the
    smoothing kernel's RMS radius, measured in number of
      rectangle sides, so that after smoothing nsmooth
       times this radius is sqrt(0.5*nsmooth)
0
```
Right now we don't want to smooth, we can do that later.

```
 Specify the base level of the rate-of-strain variances, to
  which the variance-covariance data are to be added, by
   entering a quantity equivalent to squaring the result of
    dividing a velocity by the earth's radius. The velocity
     in question is the jump in velocity in any region that
      could be accounted for either across a fault of
       length equal to the square root of the area of the
        region or across a series of faults amounting to the
         same thing. (N.B. this base level can be zero,
          provided all the data standard errors are greater
           than zero)
```

Here we input thevalue; 0.02. This base-level value will reflect a jump velocity of 1 mm/yr per grid area, where 1 mm/yr is divided over the Earth's radius and then this ratio is squared; 0.02. It will be clear later why we assign this specific value.

```
 Enter the number of times the variance-covariance data is
 to be smoothed between adjacent rectangles - each time the
  smoothing is performed 0.5 is added to the square of the
   smoothing kernel's RMS radius, measured in number of
     rectangle sides, so that after smoothing nsmooth
      times this radius is sqrt(0.5*nsmooth)
0
```
Also, no smoothing here yet.

This program will have produced a ***spline_fit.dat*** file based on the ***spline_fit.dat*** file that was created by **make_raw_spline_fit_dat** and which only contained zeroes originally.

4.2.1. Uniform and Isotropic

In this case your the ***spline_fit.dat*** needed to run **sparse_fit_with_GPS** will be simply the one that was created **perform_smoothing_on_input**. A priori strain rate variances will be uniform over the area considered and it will be isotropic; var(exx)=var(eyy). However, you will find that a jump velocity of 1 mm/yr (assumed in **perform_smoothing_on_input** for each grid area) may be too small to accommodate motions over some plate boundaries. In that case you need to input a value larger than 0.02, corresponding with the desired jump in velocity.

4.2.2. Non-Uniform and Isotropic

If you want to specify a non-uniform distribution of the a priori magnitude variances (reflecting expected regional variation in strain rate or 'strength') you need to multiply the values the variances in *spline_fit.dat* with an appropriate value for each area. If you have chosen a base-level of 0.02 in **perform_smoothing_on_input** (corresponding to a jump of 1 mm/yr in each grid area) you simply multiply the variances in each grid area with the number of mm/yr that you expect for each grid area. Higher variances and covariances in one area with respct to another one will give this area the propensity to strain more. For instance, suppose you have a plate boundary zone that accommodates 100 mm/yr and which is covered by 10 grid cells (normal to the grid boundary). When it is known that deformation occurs only in 3 of the 10 grid cells you want assign a value of roughly 30 to these three cells, and assigning a value of 1 to the other 7 cells. We can do this only because we have given a base-level variance to each area corresponding to a 1 mm/yr jump. For the described example you may find that it may be necessary to actually assign larger values with the total over the plate boundary exceeding 100 mm/yr.
It may be convenient to have the isotropic constraint to be an option when creating anisotropic constraints in the next session (see also Appendix E).

To apply smoothing:

run perform_smoothing_on_input
do not input any base-level variance, only amount of desired smoothing of variances and (if applicable) strain rates.

The total set of extra procedures (on top of the ones described in the sections for the inversions of seismic data) which are necessary to invert GPS data with constraints as discussed above is;

perform_smoothing_on_input
create a non-uniform distribution
perform_smoothing_on_input
convert_lat_long
cp *lat_long.dat output.dat*
make_x_for_output
sparse_fit_with_GPS

4.2.3. Non-Uniform and Constraints from Deformation Style

One would ultimately like to include the actual style of deformation observed from the earthquakes in order to apply anisotropic constraints on the expected strain rate field. We have to create an ***errors.dat*** file that will be added on to *spline_fit.dat*, where ***errors.dat*** will contain a constructed a priori matrix based on the principal axes of the seismic strain rate field. Appendix E contains a description of ***errors.dat*** and discusses how to implement isotropic versus non-isotropic constraints. Again, you want to include again your assumptions of non-uniformity and therefore have to multiply the appropriate components in ***errors.dat*** with a certain factor. The following steps need to be made between the two instances when you run **perform_smoothing_on_input** in section 4.2.2.

cp spline_fit.dat spline_fit.old
add_errors_to_input *will destroy the old spline_fit.dat*

run **add_errors_to_input**

Note: defining the style of deformation in the way described here does not allow you to differentiate between compression or extension or, for that matter, between left-lateral or right-lateral shear. However, since there is in general a consistency between the strain rate field associated with the GPS observations and the strain rate field that is used as constraint from seismic observations, this way of assigning constraints should not give any problems.

5. Plate Boundary Velocities Constraints

For any of the discussed cases above you could always add the constraint of plate boundary velocities. Therefore you need to adjust *spline_fit.dat* before you run the actual inversion. Let's go back to the *spline_fit.dat* as we had it for the case of seismic observations with the data smoothed once;

```
16          17          120
         119
   120    0.00000    0.00000  0.00000000E+00
         209

     1  11   2 eqs= 70 areas= 0.40264541E-02  0.37104352E-02  0.21507430E-02
 -0.937E+00 -0.107E+02 -0.429E+01
  0.257E+01  0.392E+01  0.242E+01 -0.0010003 -0.0266085  0.2179890

     2  12   2 eqs= 32 areas= 0.30030380E-02  0.29979017E-02  0.21475930E-02
 -0.119E+01 -0.363E+01 -0.463E+01
  0.297E+01  0.456E+01  0.281E+01 -0.0008635 -0.0247269  0.2213013

.....etc.
```

The second line says that there are 119 free rotation values and that the 120th rotation value is the reference frame (‘0 0 0’ because there is no rotation applied to it). If we have assigned the 119th rotation value in ***geometry.dat*** to another plate (i.e., multiple points with the same rotation value) we could assign a plate boundary velocity (e.g., NUVEL-1A motion) to this plate. Therefore, we need the change the second line to 118, since we will have one less rotation value that needs to be solved for. Furthermore, before the line with the reference frame we want to add a line with the rotation pole for the considered plate motion, for instance Pacific-Australia relative motion (latitude, longitude, (rad/yr)*10^9). You could only do this if rotation value 120 was assigned to Australia and 119 to Pacific. ***Spline_fit.dat*** will look like:

```
             16          17          120
         118
   119    -60.2  178.3 18.67
   120    0.00000    0.00000  0.00000000E+00
         209

     1  11   2 eqs= 70 areas= 0.40264541E-02  0.37104352E-02  0.21507430E-02
 -0.937E+00 -0.107E+02 -0.429E+01
  0.257E+01  0.392E+01  0.242E+01 -0.0010003 -0.0266085  0.2179890

     2  12   2 eqs= 32 areas= 0.30030380E-02  0.29979017E-02  0.21475930E-02
 -0.119E+01 -0.363E+01 -0.463E+01
  0.297E+01  0.456E+01  0.281E+01 -0.0008635 -0.0247269  0.2213013.....etc.
```

Of course, one could choose for multiple plate boundary velocities constraints when one has assigned more (rigid) plates, as long as the constrained plates cover the last rotation value(s) before the value of the reference frame.

6. Obtaining Angular Velocities

For any rotation value, either assigned to a rigid plate, GPS study (that wasn't assigned to a reference frame), or individual knotpoint, one could retrieve the predicted angular velocity relative to the chosen reference frame;

Run **sparse_rotation_solution**

The ouput is given in ***rotation_solution.dat***. In this file for every assigned rotation value the angular velocity vector is given, first in x,y and z coordinates, followed by the lat, long and rotation rate in (rad/yr)*10^9.
The angular velocity that corresponds to the rotation value assigned to a GPS study shows the rotation vector with which the geodetic vectors are rotated from their original reference frame to the model reference frame.

7. Plotting Procedures

At this moment no plotting procedures are provided. It is left to the user to find his/hers own convenient way of plotting the solutions.

8. References

Beavan, J., and J. Haines, Contemporary horizontal velocity and strain rate fields of the Pacific-Australian plate boundary zone through New Zealand, *J. Geophys. Res., 106*, 741-770, 2001.

Haines, A. J., and W.E. Holt, A procedure for obtaining the complete horizontal motions within zones of distributed deformation from the inversion of strain rate data, *J. Geophys. Res*., 98, 12,057-12,082, 1993.

Haines, A. J., J. A. Jackson, W. E. Holt, and D. C. Agnew, Representing distributed deformation by continuous velocity fields, Sci. Rept. 98/5, inst. of Geol. and Nucl. Sci., Wellington, New Zealand, 1998.

Holt, W. E., and A. J. Haines, The kinematics of northern South Island New Zealand determined from geologic strain rates, *J. Geophys. Res*., 100, 17,991-18,010, 1995.

Holt, W.E., B. Shen-Tu, A.J. Haines, and J. Jackson, On the determination of self-consistent strain rate fields within zones of distributed continental deformation, *in The History and Dynamics of Global Plate Motions*, eds. M.A. Richards, R.G. Gordon, and R.D. van der Hilst, AGU, Washington, D.C., 2000

Kreemer, C., W.E. Holt, S. Goes, and R. Govers. Active deformation in the eastern Indonesia and Philippines from GPS and seismicity data, *J. Geophys. Res, 105.*, 663-680, 2000.

Shen-Tu, B., Holt, W.E., and Haines, A.J., Intraplate deformation in the Japanese Islands: a kinematic study of intraplate deformation at a convergent plate margin, *J. Geophys. Res*., 100: 24,275-24,293, 1995

Shen-Tu, B., Holt, W.E., and Haines, A.J., Contemporary kinematics of the westrern United States determined from earthquake moment tensors, very long baseline interferometry, and GPS observations, *J. Geophys. Res*., 103, 18,087-18,117, 1998.

Appendix A: *Geometry.dat*

The geometry is set up on a rectangular grid of general coordinates (x,y). The values of x and y go from zero up to maximum values and they should all be given as integers. The knot points of the grid are where x and y have integer values. The file looks like;

```
16          17          120
           0            0 0 3 3
           1            0 0 3 3
           2            0         120           0 3
  -30.88000       101.0800
           3            0         120           0 3
  -33.64000       106.3900
           4            0         120           0 3
  -35.96000       111.6600
           5            0         120           0 3
  -38.13000       116.5600
           6            0         120           0 3
  -39.72000       121.4700
           7            0         120           0 3    etc.
```

The first line contains 1) the maximum number of knotpoints in x direction, 2) in y direction and 3) the number of rotation vectors, which you are either solving for or put in as a constraint. This same number should also be given to the point or points that are considered to be the reference frame, i.e. the last rotation value will correspond to the reference frame which will normally be assigned to several points defining a rigid plate, although it does not necessarily have to be rigid. When other (rigid) plates are considered as well, then these will get the last but one rotation value and so on...

IMPORTANT NOTE: THE PROGRAMS ARE SET-UP TO RUN GRIDS WITH A MAXIMUM SIZE OF 300x160 (x-dir, y-dir). IF YOU WANT TO RUN LARGER GRIDS AND YOUR COMPUTERS ARE CAPABLE OF RUNNING THOSE, YOU NEED TO CHANGE THE PARAMETER SPECIFICATIONS IN ALL PROGRAMS YOU WILL USE (AND DON'T FORGET THE SUBROUTINES)

Second line contains x coordinate number and y coordinate number (note that this starts at zero and therefore the number in either directions is one more than specified on first line). This is followed by three index numbers;

(1) the index of the rotation value/vector for that point. Several points, such as on a rigid plate, can have the same rotation value. If a point is not being used, this is indicated by setting the rotation value to be zero.

(2) an index indicating which derivatives of rotation values are preset to be zero at the point

0 = x derivative and y derivatives of rotation values are preset to be zero (appropriate for a rigid plate),

1 = x derivative free while y derivative is zero,

2 = x derivative zero while y derivative is free,
3 = both derivatives are free. (this is most commonly used when
the point is within a deforming region).

(3) An index indicating which derivatives of x and y as function of latitude and longitude are preset.

0 = both [d(lat)/d(x), d(lat)/d(y)] and [d(long)/d(x), d(long)/d(y)] derivatives are specified. If both are to be specified, then the derivatives follow the latitude and longitude - in the two lines below - as:

```
          lat          long
     [d(lat)/d(x)], [d(long)/d(x)]
     [d(lat)/d(y)], [d(long)/d(y)]
```

1 = only the derivatives of lat and long as function of y (y - coordinate) are specified after the latitude and longitude points:

```
          lat        long
     [d(lat)/d(y)], [d(long)/d(y)]
```

2 = only the derivatives of lat and long as function of x (x - coordinate) are specified after the latitude and longitude points:

```
          lat        long
     [d(lat)/d(x)], [d(long)/d(x)]
```

3 = both sets of derivatives are specified automatically. Unless you specifically want to alter the grid geometry (contrary to how the spline interpolation does it between knotpoints) it is easiest to specify option 3 and have the program calculate the derivatives for you.

On the <u>third line</u> you put the initial <u>latitude and longitude</u> of the point; that is, the position of the point at time t=0. (These are not input if the point is not used). This is followed by the next point.

Appendix B: *Earthquakes.dat*

In the EXAMPLES/seismic directory you will find the following file:

```
 38.22   90.67  0.14E+19     -0.125     -0.642     -0.460

-56.15  217.07  0.41E+18     -0.832      0.821     -0.551

-10.41  118.86  0.31E+19      0.000     -0.809      0.002

 27.25   56.73  0.52E+17     -0.147     -0.494      0.588

 -2.99  144.79  0.17E+20      0.274     -0.017     -0.643

 51.68  184.30  0.59E+18     -0.146     -0.516      0.331

 -2.31   28.39  0.48E+17      0.235      0.765     -0.424      etc.
```

The first line contains the latitude, longitude, moment (in Newton-meters), mxx, myy, mxy.
This is followed by blank line (without this space you will read in only half of the events). The next line is the next event.
The coordinate system is x - east, y - north, and z - positive up. Note that this is different from Aki and Richards what is x- north, y - east, z - down. Thus, if moment tensors are in Aki and Richards coordinates, then conversion is

Mxx = Myy
Myy = Mxx
Mxy = Mxy

Appendix C: Incompleteness Factor

We choose a jump in slip-rate of 0.09 mm/yr for each grid area and we convert this into the asked for value by:

factor = [((0.9 mm/yr)*(1*10^9 year))/(6.371e9 mm)]**2

Remember that we convert to dimensionless time units by multiplying mm/yr times 10^9 yr and then divide that velocity by the radius of the earth in mm.

In terms of what size of earthquake magnitude this jump in velocity would correspond to over the 21.42 year period, we have; 21.24 yr * 0.9 mm/yr = 19.3 mm of slip.
The square root of the average area gives a dimension of about 200 km. Given a fault of this length, and given a fault width of 30 km = assumed seismogenic thickness.
Then the fault area = 2.0e7 cm * 3.0e6 cm = 6e13 cm**2.
Using a shear modulus of 3.5e11 dyn/cm**2, the static seismic moment is
M_0 = (shear modulus)*(Fault Area)*(average Slip)
M_0 = (3.5e11dyn/cm**2)*(6e13 cm**2)*(1.93 cm) = 4.1e25 dyn-cm.
Using the relation of Mw = log(Mo)/1.5 - 10.7 we have;
Mw = log(4.1e25)/1.5 - 10.7 = 6.4

Thus, the size of an incompleteness factor of 0.02 is equivalent to a "missing earthquake" within each area of about Mw = 6.4 over the 21.42 year time interval. Of course you can also start with an estimate for the “missing earthquake” first and determine the incompleteness factor subsequently.

Appendix D: *GPS.dat*

This file may look as follows:

```
93
  1   111    -1 -35.40  148.98
  3.668   8.886
 0.028  0.016  0.000
  2   111    -2 -42.80  147.44
  2.946   8.862
 0.044  0.025  0.000
  3   111    -3 -31.80  115.89
  7.005   8.827
 0.038  0.019  0.000                                   .....etc.
```

- First line: number of GPS stations/observations.
- Second line:
  1) number of observation
  2) rotation number that acts as reference for this GPS observation. This number can come from two different possibilities. It could be the same number as is given to one of the rigid plates or knotpoints. That would mean that the given velocity vector is in a reference frame defined by the assigned plate or knotpoint. Alternatively, you can assign a rotation value that is not assigned to be any knotpoint or plate. In that case the rotation values assigned to the plates should follow the rotation value(s) assigned to the geodetic velocities, such that the reference plate/point in the inversion is still the last rotation value. In the latter case a angular rotation vector will be solved for to rotate the GPS observation(s) into the reference frame you have chosen in the inversion, such that the difference between observations and predicted velocities is minimized. This might be convenient when your grid does not include the actual reference frame in which the GPS observations are given, or when its reference frame is not well defined. Whichever option you choose, be sure that all observed vectors that belong to the same study get assigned the same rotation value.
  3) the rotation value for the knotpoint at which the GPS station is positioned. Most likely you didn't set up your grid this way and the GPS station will not be positioned on any knotpoint of the grid. If that's the case you assign a value equal to the minimum value of the given on the first place of the second line.
  4) the latitude and longitude of station.
- Third line: longitudinal velocity (east) and latitudinal velocity (north). Input should be dimensionless and therefore the values are normalized by the earth's radius (i.e., 6.371, if your velocities were in mm/yr, and if you are going to use data outcome in $1*10^{-9}$).

Fourth line: standard deviations of the longitudinal and latitudinal velocity (also normalized by the earth's radius), and their correlation coefficient.

Followed by the next station.

Appendix E: *Errors.dat*

This file may look like;

```
209
           1           11            2
   112.3431       119.0761       132.2466        34.90000     -0.9221914
           2           12            2
   132.7054       83.26631       188.4221        40.40000     -0.8540727
           3           13            2
   145.6113      0.9650108       4.798995       0.9600000     -9.9716589E-02
           4            1            3
   174.2622       2.830000       2.830000        2.830000      0.0000000E+00
           5            2            3
  0.4245000       3.370000       3.370000        3.370000      0.0000000E+00
           6           11            3
   123.8252       108.5765       148.0106        36.00000     -0.9151013

etc.
```

- <u>First line</u>; number of grid cells
- <u>Second line</u>; number of the grid cell and x and y coordinate.
- <u>Third line</u>; 1) azimuth of largest principal axis (which is ineffective for an isotropic case)
  2) standard error of $\dot{\sigma}$,
  3) standard error of $\dot{\gamma}_1$,
  4) standard error of $\dot{\gamma}_2$,
  5) the correlation coefficient.

Followed by the next grid cell.

The theoretical background is discussed in *Haines et al.* [1998] and a brief overview is given here.

The azimuth may be directly taken from the direction of the modeled principal seismic strain rate in the case of having anisotropic constraints. For isotropic constraints it can be anything..
In the coordinate system of the principal axes of the strain rate tensor we can write the dilatational shear strain rate;

$$\dot{\sigma} = \frac{1}{2}(\dot{e}_{xx} + \dot{e}_{yy})$$

(where $\dot{e}_{xx}$ and $\dot{e}_{yy}$ are the principal axes) and the horizontal shear strain rates;

$$\dot{\gamma}_1 = \frac{1}{2}(\dot{e}_{xx} - \dot{e}_{yy}) \text{ and } \dot{\gamma}_2 = \dot{e}_{xy}$$

Now, the variances of $\dot{\sigma}$, $\dot{\gamma}_1$ and $\dot{\gamma}_2$ can be written as;

$$\mathrm{var}(\dot{\sigma}) = [\sin^2\psi + \varepsilon^2\cos^2\psi]\Sigma$$
$$\mathrm{var}(\dot{\gamma}_1) = [\cos^2\psi + \varepsilon^2\sin^2\psi]\Sigma$$
$$\mathrm{var}(\dot{\gamma}_2) = \varepsilon^2\Sigma$$

and the covariance as; $\mathrm{cov}(\dot{\sigma},\dot{\gamma}_1) = [(1-\varepsilon^2)\cos\psi\sin\psi]\Sigma$

with $\tan\psi = \dot{\sigma} / \dot{\gamma}_1$, $\Sigma = \mathrm{var}(\dot{e}_{xy})$ and the value of ε is chosen according to how much confidence you have in the direction of the principal axes;
ε = 0.2 corresponds to an uncertainty between ~±10° in the directions of the axes,
ε = 0.5 corresponds to ~ ±20°; and
ε = 1.0 corresponds to complete uncertainty of ±45°. This is appropriate for an isotropic case.

The correlation coefficient that is asked for as input is; $\mathrm{cov}(\dot{\sigma}, \dot{\gamma}_1) / [\sqrt{\mathrm{var}(\dot{\sigma})} * \sqrt{\mathrm{var}(\dot{\gamma}_1)}]$

For uniform cases you need to multiply the input values $\sqrt{\mathrm{var}(\dot{\sigma})}$, $\sqrt{\mathrm{var}(\dot{\gamma}_1)}$, and $\sqrt{\mathrm{var}(\dot{\gamma}_1)}$ with a certain factor if the base-level that you have inputted in **perform_smoothing_on_input** does not suffice (alternatively you should have inputted a different value in **perform_smoothing_on_input).** See section 4.2.2 on ideas on how one is able to have the magnitude of variances to be non-uniform within the grid.

The *errors.dat* file shown above is an example of a non-uniform isotropic case.

Appendix F: Some Theoretical Background - by Sarah Jenny

(adapted from *Haines et al.* [1998])

This appendix must be downloaded separately. It is called appendixF.pdf