Contents

1. Acknowledgements

The utility, quality and evolution of the NonLinLoc software package depends entirely on feedback from its users. Many thanks to the following people for contributions, comments and suggestions for the NonLinLoc software:

- Giuseppe Aliberti: formerly University of L'Aquila, Italy
- Alexandre Nercession: IPGP, Paris, France
- Rick Aster: New Mexico Inst Mining and Tech, USA
- Alberto Michelini: OGS, Trieste, Italy
- Stephan Husen: ETH, Zurich, Switzerland
- Mariella Simini: Univ. Naples, Italy

This work is supported in part by funding from IPSN, Fontenay−aux−Roses, France, and by the European Union project TomoVes (ENV4−4980696).

2. Overview

NonLinLoc Version 2.30 (18MAY2001)

The NonLinLoc (Non−Linear Location) package is a set of programs and utilities for velocity model construction, travel−time calculation and probabilistic, non−linear, global−search earthquake location in 3D structures, and for visualisation of 3D volume data and location results. Many of the programs operate with 3D volumes, usually in the form of a **3D Grid** structure which defines a specific, gridded, rectangular volume in space.

See the book chapter **Probabilistic earthquake location in 3D and layered models: Introduction of a Metropolis−Gibbs method and comparison with linear locations** for further information on NonLinLoc and for an examination of synthetic 3D and 1D locations. See the article **Lomax, Zollo, Capuano, and Virieux, 2001** for further information on NonLinLoc and for an examination of 3D locations of micro−seismicity under Vesuvius volcano.

Copyright notice

3. The Programs

The NonLinLoc package consists of the following programs:

NLLoc	**Non−Linear Location** − Determines the location for one or more events within a 3D Grid using a systematic grid−search, a stochastic, Metropolis−Gibbs search, or a hybrid "Oct−Tree" method. (outputs misfit or probability density function (PDF) on a 3D Grid, PDF samples or "scatter cloud", description of best hypocenter, and other results).
Vel2Grid	**Velocity model to 3D Model Grid** − Given a velocity model description, creates a model grid. (outputs a 3D Grid)
Grid2Time	**3D Model Grid to Travel Time Grid** − Given a 3D Model Grid, calculates travel times from a point within a 3D Grid to all other points within the grid. Optionally calculates ray take−off angles within the 3D Grid. (outputs a set of 3D Grids)
Time2EQ	**Travel Time Grid to Synthetic Observations** − Given a hypocenter location and a set of travel time grids, calculated predicted travel times. (outputs a Phase file in NonLinLoc format)
LocSum	**Location Summation** − Utility that combines NLLoc location results and PDF "scatter−cloud" samples from a number of events. (outputs a single set of summary location files)
Grid2GMT	**Grid to GMT commands** − Utility that creates a command script for plotting plotting sections and plan views of 3D Grid files, and misfit, PDF and location "scatter−cloud" results using the GMT plotting package.
Utilities	**Miscellaneous utility programs** − Utilities for conversion between HYPOELLIPSE summary or FPFIT summary formats and NonLinLoc Hypocenter−Phase location format.

In addition, the following analysis and visualisation software supports NonLinLoc event location output formats:

SeismicityViewer	**Java program for viewing seismicity in 3D** − Interactive, animated, local or Internet−based viewing in 3D of hypocenter, PDF "scatter−clouds", error−ellipses, residuals, geographic data and other information.

4. A Typical Location Run

A typical location run involves:

A. One−time preparation of travel−time grids:

1. using Vel2Grid or other software to produce a velocity or slowness model 3D Grid File for the search region for each phase type, (i.e. P or S).
2. using Grid2Time to produce travel−time and ray take−off angle 3D Grid Files for each phase type at each station

B. Locating events:

1. obtaining seismic phase time picks in a supported format (see Phase File Formats)
2. determining a 3D search region and velocity model for this region
3. using NLLoc to locate each event defined by the phase picks using a nested grid−search or Metropolis−Gibbs search over a 3D volume
4. using the utilities LocSum and Grid2GMT or SeismicityViewer to process and plot the location results

5. Getting Started

Download the NonLinLoc Software

Go to the NonLinLoc home page (http://www.alomax.net/nlloc) for links to downloadable files.

NonLinLoc

Version 2.30 (18MAY2001)

− Download the NonLinLoc Software and compile it on your machine,

or

− Download the NonLinLoc Linux binary executables.

NOTES:

1. **With gcc and Sun Solaris 2.6 on a Sun Workstation I have successfully compiled the NonLinLoc software using the command** `make distrib`**. With gcc and Linux(SuSE 6.4) on a Pentium III PC I have successfully compiled the NonLinLoc software using the command** `make -R distrib`**. If you have successfully installed and run the NonLinLoc software on another system, I would appreciate hearing about the details of your hardware, operating system, compiler and compiler flags. Thanks. lomax@geoazur.unice.fr**
2. **Grid2GMT: I have made changes to Grid2GMT to support GMT 3.3.6, though there seems to be bugs or other problems, i.e. the labelling of scale bars is incorrect. If you use an earlier version of GMT, you may need to change the line** `#define GMT_VER_3_3_4 1` **in Grid2GMT to** `#define GMT_VER_3_3_4 0`**.**

SeismicityViewer

Download the SeismicityViewer java code and class files.

Sample Location Problem

Version 3.00 (18APR2001) − Download the NonLinLoc Sample Location control and output files.

Go to the chapter “Installing and Running the Sample Location Tutorial” to create a velocity grid, calculate travel times and locate an event, and visualise the results results.

6. NonLinLoc Release Notes

Date	Type	Description
18MAY2001	New Release v2.30	NonLinLoc software v2.30 released (Updated release of the NonLinLoc software and documentation.) New Features: 1. NLLoc: Added copying of output files from last location to files `last.hdr`, `last.hyp`, etc. 2. NLLoc: Added new magnitude type MD_FMAG (duration magnitude) to control statement LOCMAG. 3. NLLoc: Added SIMULPS format for phase pick files. 4. NLLoc: Added N Calif Seismic Network (NCSN)format. 5. NLLoc: Added grid memory management; NLLoc program keeps and re−uses travel−time grids in memory while there is space. 6. NLLoc: Added Oct−tree importance sampling search method. 7. Grid2GMT: Added plotting of mechanisms and residuals. 8. NLLoc: Added new `STAT_GEOG` line to Hypocenter−Phase file output. 9. Generic control statement MAPLINE: Added support for GMT grd file plotting in Grid2GMT. 10. Generic: Added new, optional control statement MAPTRANS: Specifies geographic transformation for plotting in Grid2GMT. 11. Generic: Added new, optional control statement MAPGRID: Specifies grid region for plotting in Grid2GMT.

Date	Type	Description
		Updates/Changes: 1. NLLoc: Added new station parameter sta_corr_fd_fmag (duration magnitude station correction) to control statement LOCCMP. 2. Seismicity Viewer: Updated version, runs as stand−alone and applet. 3. NonLinLoc: Binary executable files provided for Linux, no longer for Solaris. 4. NLLoc: Added maximum difference of S−P to Hypocenter−Phase file VpVs line output. 5. NLLoc: Added quick fix for HYPO71 phase data Y2K problem: if year < 20, year = year + 100. 6. NLLoc: Added quick fix for HYPO71 phase data Y2K problem: if year < 20, year = year + 100. Bug Fixes: 1. NLLoc: Fixed bug: hypo−inverse format: W long and S lat were not correctly written to summary file. 2. NLLoc: Fixed bug: Events with maximum−likelihood hypocenters on y−max boundary of location search grid were not identified as being on the grid boundary, and consequently were not "rejected". 3. NLLoc: Fixed bug in Hutton Boore magnitude calculation: epicentral and not hypocentral distance was used for **r** in **K(r−100)** term; changed to hypocentral distance. 4. NLLoc: Fixed bug in identifications of maximum likelihood location on search grid boundary: locations on Y−max boundary were not correctly identified.

Date	Type	Description
17FEB2000	New Version v2.00	NonLinLoc software v2.00 released (New version of the NonLinLoc software and documentation. Significant changes are indicated in the documentation by **(ver 2.0)**) New Features: 1. NLLoc: Control statement LOCMETH: Added *minNumberSphases*, minimum number of S phases that must be accepted before event will be located **(Important: You must add a *minNumberSphases* value to the** LOCMETH **statement in any existing NLLoc control files to update them for NLLoc v2.0)** 2. NLLoc: Added new control statement LOCEXCLUDE to explicitly exclude selected station/phase observations from being used for location 3. NLLoc: output Hypocenter−Phase file: Added Vp/Vs estimation (in VPVSRATIO line) following methodology of Lahr (1989) 4. Grid2GMT: Added optional override of default GMT−cpt colours. If a GMT−cpt file named Grid2GMT.cpt is present in the directory in which the GMT script is run, it will be used for the plot and scalebar contours levels and colours for all plot types that use contouring. 5. Grid2GMT: Added optional additional GMT script to be run for each view type. If a GMT script file named Grid2GMT.?.gmt, where ? = H, X, Y or V is present in the directory in which the main GMT script is run, it will be run for the corresponding view (H, X, Y or V)after all standard GMT script is run for that view. 6. Time2EQ: Added new control statement EQVPVS to specify constant Vp/Vs ratio to allow use of P phase travel−time grids to calculate S phase travel times. Updates/Changes: 1. Grid2GMT: Added support for oblique vertical cross sections in V mode and for specification of cross section end points in latitude and longitude (new VL mode). 2. Time2EQ: Control statement EQSTA: Added NONE error type

<table>
<tr><th>Date</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td>Bug Fixes:

NLLoc: Residual calculation in Hypocenter−Phase file: Fixed MAJOR BUG in calculating phase residuals for some travel−time file types if Vp/Vs ratio was used (residual written to output did not correspond to maximum likelihood location)
NLLoc: Phase observation time in Hypocenter−Phase file: Fixed MAJOR BUG that observed times were written to output with delays subtracted for phases with LOCDELAY corrections
NLLoc: Control statement LOCPHASEID: Fixed bug in identifying "S" phases to use Vp/Vs (S phases not identical to "S" were ignored, regardless of LOCPHASEID settings)
NLLoc: Fixed bug that not all observation file types had phases checked for valid P or S phase code before accepting observation for location
Vel2Grid: Control statement POLYGON2: Fixed bug that velocity gradients in POLYGON2 were reversed (positive gradients were interpreted as decreasing downwards)
Grid2GMT: Fixed bug that grid files were not closed when multiple events were processed.
</td></tr>
<tr><td>25JUN1999</td><td>New Release v1.30</td><td>NonLinLoc software v1.30 released (Updated release of the NonLinLoc software and documentation.)

Updates/Changes:

Generic control statement TRANS: Implemented rotAngle in LAMBERT transformation.
</td></tr>
<tr><td>08APR1999</td><td>New Release v1.21</td><td>NonLinLoc software v1.21 released (Fourth "Beta" release of the NonLinLoc software and updated documentation.)

Updates/Changes:

NLLoc: LOCMETH Added parameter to limit maximum number of 3D travel time grids read into memory for Metropolis−Gibbs search. This helps to avoid time consuming memory swapping.
</td></tr>
</table>

<table>
<tr><th>Date</th><th>Type</th><th>Description</th></tr>
<tr><td>10MAR1999</td><td>New Release v1.2</td><td>NonLinLoc software v1.2 released (Third "Beta" release of the NonLinLoc software and updated documentation.)

New Features:

1. NLLoc: LOCMETH Added option for constant Vp/Vs ratio to allow use of P phase travel−time grids to calculate S phase travel times.

Updates/Changes:

1. Generic control statement CONTROL: Added random number seed.

Bug Fixes:

1. NLLoc: Fixed bug that automatic x grid positioning could not be switched off (see Control statement LOCGRID).</td></tr>
</table>

Date	Type	Description
25JAN1999	New Release v1.1	NonLinLoc software v1.1 released (Second "Beta" release of the NonLinLoc software and updated documentation.) New Features: 1. NLLoc: Control statement LOCPHSTAT: Added RMS_Max, NRdgs_Min and Gap_Max cut−offs for selection of residuals to be used in calculation of average residuals. 2. Grid2Time: Control statement GTSRCE: Added LATLONDS (degrees + minutes + decimal seconds latitude/longitude) format for source location. 3. Time2EQ: Added control statement EQMECH to allow specification of a double couple or explosion mechanism for synthetic first motion calculation. 4. Time2EQ: Added control statement EQSRCE to allow specification of multiple sources. 5. Time2EQ: Control statement EQSTA: Added FIX option to allow a fixed noise or static shift to travel time. 6. Time2EQ: Control statement EQSTA: Added parameters errorReportType and errorReport to specify how phase error is written to phase file. 7. Time2EQ: Added control statement EQMODE to select calculation of times from single source to multiple stations or from multiple sources to single station. 8. Time2EQ: Added control statement EQQUAL2ERR to allow mapping of errorReport to HYPO71 format quality levels. 9. LocSum: Added additional command arguments Len3Max, ProbMin, RMSMax, NRdgsMin and GapMax to filter events that are summed. Updates/Changes: 1. Generic control statement TRANS: Implemented rotAngle in SIMPLE transformation. 2. NLLoc: Grid−Search and Control statement LOCGRID: Automatic grid shifting added to attempt to get nested grids inside original grid. 3. NLLoc: Metropolis−Gibbs search: Initial walk location now set under the station with earliest arrival time. Algorithm for automatic setting of initial step size changed. 4. NLLoc: Metropolis−Gibbs search: Walk algorithm modified to handle the case that new walk locations are rejected for a large number of consecutive tries. This case may indicate that the search has found a deep, narrow minima in misfit. 5. Grid2GMT: 2D error ellipses replaced by projections of 3D confidence ellipsoid. 6.

Date	Type	Description
		Bug Fixes: 1. Generic control statement INCLUDE: Implemented for programs Vel2Grid, Grid2Time, Time2EQ and NLLoc.
20JAN1999	Known Bug	Grid registration not specified NonLinLoc software 3D Grid File format does not distinguish between Grid line registration (used for travel times, angles, etc) and Pixel registration (required for velocity models input to Grid2Time/Podvin−Lecomte) as defined by the GMT plotting package.
11AUG1998	New Release v1.0	NonLinLoc software v1.0 released (Initial "Beta" version of the NonLinLoc software and documentation.)

7. Installing and Running the Sample Location Tutorial

Tutorial for the installation, running and visualisation of the sample location problem.

Overview − Installing the sample location − Running the sample location tutorial

Overview

The following steps will lead you through the installation of files, the creation of a model grid and calculation of travel time grids, and the location and visualisation of the sample location.

Before doing the sample location, you must download and install the NonLinLoc software, and, for visualisation, the Seismogram Viewer software (see the NonLinLoc Home page). For postscript plotting, you need to have the the GMT plotting package available on your system. (Note that the sample location tar include a hidden ".gmtdefaults" file that will be installed in your sample location working directory. You may want to modify this file, but for proper functioning of Grid2GMT leave the units as "MEASURE_UNIT = cm")

Copies of some the output files you should obtain will be in the `original_output/` sub-directory of the directory where you install the sample location files.

Installing the sample location

To install on a UNIX system:

Create a working directory, i.e.:

```
mkdir $HOME/NLLoc
```

Copy the samples tar file to this directory, i.e.:

```
cp $HOME/download/NLLVersion_samples.tar.gz $HOME/NLLoc
```

where: *`Version`* is the NNloc version you are using (i.e. `NLLoc1.30_samples.tar.gz`).

Go to the working directory, i.e.:

```
cd $HOME/NLLoc
```

Unzip and unpack the files:

```
gunzip NLLVersion_samples.tar.gz
tar -xvf NLLVersion_samples.tar
```

This will create the directory `nlloc_sample/` with sub−directories:
`data_geog/` `loc/` `obs/` `run/` `viewer/` `gmt/` `model/` `original_output/` `time/`
and a hidden `.gmtdefaults` file.

Examine the NonLinLoc Control file: `run/nlloc_sample.in`, this file contains the statements used by each of the NonLinLoc programs when the sample is run.

Running the sample location tutorial

To run the sample location tutorial:

Go to the samples directory, i.e.:

```
cd $HOME/NLLoc/nlloc_sample
```

Generate and view the model grid

Create the model grid:

```
Vel2Grid run/nlloc_sample.in
```

Create a GMT script and postscript image of the model:

```
Grid2GMT run/nlloc_sample.in model/layer.P.mod gmt/ V G 1 0 1 301
```

Look at the postscript with your favourite viewer, i.e.:

```
gs gmt/layer.P.mod.VG.ps
```

Generate and view the travel−time and take−off angle grids

Create the travel−time and take−off angle grids:

```
Grid2Time run/nlloc_sample.in
```

Create a GMT script and postscript image of a travel−time grid:

```
Grid2GMT run/nlloc_sample.in time/layer.P.AURF.time gmt/ V G 0 0 0 301
```

Look at the postscript with your favourite viewer, i.e.:

```
gs gmt/layer.P.AURF.time.VG.ps
```

Create a GMT script and postscript image of a take−off angle grid:

```
Grid2GMT run/nlloc_sample.in time/layer.P.AURF.angle gmt/ V
```

```
G 0 0 0 301
```

Look at the postscript with your favourite viewer, i.e.:

```
gs gmt/layer.P.AURF.angle.VG.ps
```

Generate some synthetic arrival times

Create the synthetic times:

```
Time2EQ run/nlloc_sample.in
```

Look at the synthetic times:

```
more obs/synth.obs
```

Do the event Location

Locate the event using the Oct−Tree search:

```
NLLoc run/nlloc_sample.in
```

Look at the NonLinLoc Hypocenter−Phase location file:

```
more loc/vinti.19950421.080259.grid0.loc.hyp
```

Plot the first event location with GMT

Create a GMT script and postscript image of the location PDF:

```
Grid2GMT run/nlloc_sample.in
loc/vinti.19950421.080259.grid0.loc gmt/ L S
```

Look at the postscript with your favourite viewer, i.e.:

```
gs gmt/vinti.19950421.080259.grid0.loc.LS.ps
```

Combine all event locations with LocSum

Combine the Hypocenter−Phase and scatter files into a location summary file:

```
LocSum ./run/vinti 1 loc/vinti "loc/vinti.*.*.grid0.loc"
```

Plot the combined locations with GMT

Create a GMT script and postscript image of the location Expectations and Ellipsoids:

```
Grid2GMT run/nlloc_sample.in loc/vinti gmt/ L E101
```

Look at the postscript with your favourite viewer, i.e.:

```
gs gmt/vinti.LE_101.ps
```

Visualise the location with Seismicity Viewer (you must have installed Seismicity Viewer, see Seismicity Viewer software guide)

Go to the viewer directory, i.e.:

```
cd $HOME/NLLoc/nlloc_sample/viewer
```

Run Seismicity Viewer and visualise interactively the combined locations:

```
java seismicity.Seismicity ../loc/vinti.hyp
```

8. Vel2Grid Program − velocity model description to 3D model grid

Given a velocity model description, **Vel2Grid** generates a 3D model Grid header and buffer files containing velocities, slownesses or other model specification.

Overview − Running the program−Input − Output − Processing and Display of results

Overview

The Vel2Grid program converts analytic or other velocity model specifications into a 3D Grid file containing velocity or slowness values.

The Vel2Grid program uses a "flat earth", rectangular, left−handed, *x*,*y*,*z*,*t* co−ordinate system (positive X = East, positive Y = North, positive Z = down). Distance units are kilometres.

Running the program

Input

Synopsis: `Vel2Grid` *`InputControlFile`*

The Vel2Grid program takes a single argument *`InputControlFile`* which specifies the complete path and filename for an Input Control File with certain required and optional statements specifying program parameters and input/output file names and locations. See the Vel2Grid Statements section of the Input Control File for more details. Note that to run Vel2Grid the Generic Statements section of the Input Control File must contain the `CONTROL` and `TRANS` (Geographic Transformation) statements.

In addition, the Vel2Grid program requires a set of Vel2Grid Statements in the Input Control File that specify a layered model or a 3D velocity model. The velocity model can be specified in the control file by:

1. A set of LAYER statements defining a horizontally layered model with constant or constant−gradient velocity and density in each layer.
2. A set of VERTEX, EDGE, and POLYGON2 statements defining a 2D polygon model and a 2DTO3DTRANS statement to convert this 2D model into a 3D model. Optionally, there may be a set of LAYER statements defining a horizontally layered background model. This

background model must be defined if the transformed 2D polygon model does not completely fill the requested 3D grid.

Output

The velocity or slowness values throughout the requested grid are written to a new 3D Grid File. For a description of the naming convention for these grid files, see the VGOUT statement in the Vel2Grid Statements section of the Input Control File.

Processing and Display of results

The 3D model grids can be post−processed with the program Grid2GMT to produce a GMT command script for plotting with the GMT plotting package.

9. Grid2Time Program − 3D model grid to travel−time and angles grids

Given a velocity model grid, **Grid2Time** calculates the travel−time from a source point in a 3D grid to all other points in the grid. Optionally, the program also estimates the take−off angles for rays from each point in the grid to the source.

Overview − Podvin and Lecomte Algorithm − Take−Off Angles Algorithm − Running the program−Input − Output − Processing and Display of results − [NonLinLoc Home]

Overview

The Grid2Time program calculates the travel−times between a station and all nodes of an *x,y,z* spatial grid using the Eikonal finite−difference scheme of Podvin and Lecomte (1991). The results are stored on disk in a **travel−time** 3D Grid files.

Optionally, the Grid2Time program also estimates the take−off angles for rays from each point in the grid to the source by examining the gradients of the travel−time field. These results are stored in an **angles** grid.

The 3D travel−time computation and the size of the output time−grid files grow rapidly with grid dimension. However, for location in horizontally layered models the travel−times can be stored on compact 2D grids. A layered model / 2D grid can also be used for"regional" stations far from the local search volume in combination with 3D models and 3D grids for stations within the search volume. This option may introduce some error if strong heterogeneity in the local 3D velocity structure intersects the (usually downgoing) ray paths to the regional stations.

The Grid2Time program uses a "flat earth", rectangular, left−handed, *x,y,z,t* co−ordinate system (positive X = East, positive Y = North, positive Z = down). Distance units are kilometres, and many input/output distance quantities can be expressed in rectangular or geographic (latitude and longitude) co−ordinates.

Podvin and Lecomte, Eikonal, Finite−difference Algorithm

The travel times between a station and all nodes of a 3D grid are calculated using the Eikonal finite−difference scheme of Podvin and Lecomte (1991). The algorithm is implemented in the Grid2Time program using a C function time_3d() due to P. Podvin (last revision 2 January 1992). The abstract of Podvin and Lecomte (1991) describes the algorithm as:

This method relies on a systematic application of Huygen's principle in the finite difference approximation. Such approximation explicitly takes into account the existence of different propagation modes (transmitted and diffracted body waves, head waves). Local discontinuities of the time gradient in the first arrival time field (e.g. caustics) are built as intersections of locally independent wavefronts. As a consequence, the proposed method provides accurate first travel times in the presence of extremely severe, arbitrarily shaped velocity contrasts.

Associated to a simple procedure which accurately traces rays in the obtained time field, this method provides a very fast tool for a large spectrum of seismic and seismological problems.

Take−Off Angles Algorithm

The take−off angles at a node for the first−arrival ray to the source are estimated from the gradients of travel−time at the node. Two gradients are estimated for each axis direction x, y, z − one G_{low} between the node and its preceding neighbour along the axis, and a second G_{high} between the following neighbor and the node. The total gradient G_{axis} along an axis is the mean of these two gradients; the total gradient along the three axes determine the vector gradient of travel−time. The ray take−off angles R_{dip} (dip, range of 0 (down) to 180 deg (up)) and R_{az} (azimuth, range of 0 to 360 deg CW from North) specify the direction opposite to the vector gradient of travel−time.

A crude quality factor Q_{axis} between 0 and 10 is determined from the ratio

$$Q_{axis} = (20\, G_{low}\, G_{high}) / (G_{low}^2 + G_{high}^2)$$

If $Q_{axis} < 0$ (i.e. the two gradients have opposite sign), Q_{axis} is set equal to 0. If $Q_{axis} = 10$ then the two gradients have the same magnitude and sign. A final quality for the take−off angles is determined from the weighted average of the qualities along each axis, where the weighting is given by the magnitude of the mean gradient along each axis,

$$Q = (|G_x|\, Q_x + |G_y|\, Q_y + |G_z|\, Q_z) / (|G_x| + |G_y| + |G_z|).$$

Running the program

Input

Synopsis: `Grid2Time` *`InputControlFile`*

The Grid2Time program takes a single argument *`InputControlFile`* which specifies the complete path and filename for an Input Control File with certain required and optional statements specifying program parameters and input/output file names and locations. See the Grid2Time Statements section of the Input Control File for more details. Note that to run Grid2Time the

Generic Statements section of the Input Control File must contain the `CONTROL` and `TRANS` (Geographic Transformation) statements.

In addition, the Grid2Time program requires:

1. A 2D or a 3D velocity model 3D Grid file created using Vel2Grid or other software. One velocity model grid is required for each wave type (i.e. P or S). Note that a 3D Grid file may specify a 2D model.

The names, locations and other information for these files is specified in the Grid2Time Statements section of the Input Control File.

Output

The travel−times and take−off angles throughout a grid are written to a separate 3D Grid File for each phase at each station. For a description of the naming convention for these grid files, see the GTFILES statement in the Grid2Time Statements section of the Input Control File.

Processing and Display of results

The travel−time and angles grid results for a single source can be post−processed with the program Grid2GMT to produce a GMT command script for plotting with the GMT plotting package.

10. Time2EQ Program − travel−time grid to synthetic observations

Given a hypocenter location and a set of travel−time grids, Time2EQ calculates predicted travel−times.

Running the program−Input − Output − Processing and Display of results − [NonLinLoc Home]

Overview

The Time2EQ program calculates predicted travel−times between one or more synthetic events and one or more stations. Predicted take−off angles at the source are also calculated if an event mechanism is given and the corresponding take−off angles grids are available.

Running the program

Input

Synopsis: `Time2EQ` *`InputControlFile`*

The Time2EQ program takes a single argument *`InputControlFile`* which specifies the complete path and filename for an Input Control File with certain required and optional statements specifying program parameters and input/output file names and locations. See the Time2EQ Statements section of the Input Control File for more details. Note that to run Time2EQ the Generic Statements section of the Input Control File must contain the `CONTROL` and `TRANS` (Geographic Transformation) statements.

In addition, the Time2EQ program requires:

1. Files containing a 2D or a 3D **Travel−time grids** (and optionally **Angles grids**) created by the program Grid2Time for each phase type at each station.

The names, locations and other information for these files is specified in the Time2EQ Statements section of the Input Control File.

Output

The predicted travel−times are written to an observation file in NonLinLoc phase file format.

Processing and Display of results

The predicted travel−time files can be used as input phase/observation files for location with the program NLLoc.

11. NLLoc Program − non−linear, earthquake location program

NLLoc performs earthquake locations in 3D models using non−linear search techniques.

Overview − Inversion Approach − Grid−Search − Metropolis−Gibbs Sampling − Oct−Tree Sampling − Running the program−Input − Output − Processing and Display of results − [NonLinLoc Home]

Overview

The NLLoc program produces a **misfit function**, **"optimal" hypocenters**, an estimate of the **posterior probability density function** (PDF) for the spatial, *x,y,z* hypocenter location, and other results using either a systematic **Grid−Search**, a stochastic, **Metropolis−Gibbs sampling** approach, or an "oct−tree" importance sampling algorithm.

The location algorithm used in NLLoc (Lomax, et al., 2000) follows the inversion approach of Tarantola and Valette (1982), and the earthquake location methods of Tarantola and Valette (1982), Moser, van Eck and Nolet (1992) and Wittlinger et al. (1993). The errors in the observations (phase time picks) and in the forward problem (travel−time calculation) are assumed to be Gaussian. This assumption allows the direct, analytic calculation of a maximum likelihood origin time given the observed arrival times and the calculated travel times between the observing stations and a point in x,y,z space. Thus the 4D problem of hypocenter location reduces to a 3D search over *x,y,z* space.

To make the location program efficient for complicated, 3D models, the travel−times between each station and all nodes of an *x,y,z* spatial grid are calculated once using a 3D version (Le Meur, 1994; Le Meur, Virieux and Podvin, 1997) of the Eikonal finite−difference scheme of Podvin and Lecomte (1991) and then stored on disk as travel−time grid files. This storage technique has been used by Wittlinger et al. (1993), and in related approaches by Nelson and Vidale (1990) and Shearer (1997). The forward calculation during location reduces to retrieving the travel−times from the grid files and forming the misfit function $g(\mathbf{x})$ in, equation (3).

In addition, to save disk space and for faster calculation, a constant Vp/Vs ratio can be specified, and then only P travel−time grids are required for each station.

The Podvin and Lecomte (1991) algorithm and related methods use a finite−differences approximation of Huygen's principle to find the first arriving, infinite frequency travel times at all nodes of the grid. The algorithm of Podvin and Lecomte (1991) gives stable recovery of diffracted waves near surfaces of strong velocity contrast and thus it accurately produces travel times for diffracted and head waves. A limitation of the current 3D version of the method is a restriction to cubic grids. This may lead to excessively large travel−time grids if a relatively fine cell spacing is required along one dimension since the same spacing must be used for the other dimensions. This can be a problem for regional studies where a fine node spacing in depth is necessary, but the horizontal extent of the study volume can be much greater than the depth extent. Thus a modification of the travel times calculation to allow use of an irregular grid would be very useful.

After the travel times are calculated throughout the grid, the NonLinLoc program uses the gradients of travel−time at the node to estimate the take−off angles at each node. Two gradients are estimated for each axis direction *x, y,* and *z* − one G_{low} between the node and its preceding neighbour along the axis, and a second G_{high} between the following neighbour and the node. The total gradient G_{axis} along an axis is the mean of these two gradients; the total gradient along the three axes determines the vector gradient of travel−time. The direction opposite to the vector gradient of travel−time gives the ray take−off angles for dip and azimuth. An estimate of the quality of the angle determination is given by a comparison of the magnitudes and signs of G_{low} and G_{high}. If these two values are not similar, then there may be two rays which arrive nearly simultaneously at the station, and the take−off angle determination at the node may be unstable.

The *x,y,z* volume used for location must be fully contained within the 3D travel−time grids. This limits the largest station distance that can be used for location since the 3D travel−time computation and the size of the output time−grid files grow rapidly with grid dimension. However, for location in flat−layered media, the travel times can be stored on very compact 2D grids, and readings for "regional" stations far from the search volume can be used.

The NLLoc program uses a "flat earth", rectangular, left−handed, *x,y,z* co−ordinate system (positive X = East, positive Y = North, positive Z = down). Distance units are kilometres, and many input/output distance quantities can be expressed in rectangular or geographic (latitude and longitude) co−ordinates.

See the book chapter **Probabilistic earthquake location in 3D and layered models: Introduction of a Metropolis−Gibbs method and comparison with linear locations** (Lomax, et al., 2000) for further information on the NonLinLoc location algorithms.

Inversion Approach

The earthquake location algorithm implemented in the program NLLoc (Lomax, et al., 2000) follows the probabilistic formulation of inversion presented in Tarantola and Valette (1982) and Tarantola (1987). This formulation relies on the use of normalised and non−nnormalised probability density functions to express our knowledge about the values of parameters. Thus, given the normalised density function *f*(*x*) for value of a parameter *x*, the probability that *x* has a value between *X* and *X*+Δ*X* is

$$P(X \le x \le X + \Delta X) = \int_X^{X+\Delta X} f(x)dx \cdot \qquad (1)$$

In geophysical inversion we wish to constrain the values of a vector of unknown parameters **p**, given a vector of observed data **d** and a theoretical relationship $\theta(\mathbf{d},\mathbf{p})\mu_p(\mathbf{p})$ relating **d** and **p**. When the density functions giving the prior information on the model parameters $\rho_p(\mathbf{p})$ and on the observations $\rho_d(\mathbf{d})$ are independent, and the theoretical relationship can be expressed as a conditional density function $\theta(\mathbf{d}|\mathbf{p})$, a complete, probabilistic solution can be expressed as a

posterior probability density function (PDF) $\sigma_p(\mathbf{p})$ (Tarantola and Valette, 1982; Tarantola, 1987)

$$\sigma_p(\mathbf{p}) = \rho_p(\mathbf{p}) \int \frac{\rho_d(\mathbf{d})\theta(\mathbf{d}\,|\,\mathbf{p})}{\mu_d(\mathbf{d})} d\mathbf{d}, \qquad (2)$$

where $\mu_p(\mathbf{p})$ and $\mu_d(\mathbf{d})$ are null information density functions specifying the state of total ignorance.

Gaussian Error Assumption

For the case of earthquake location, the unknown parameters are the hypocentral co−ordinates x=(x,y,z) and the origin time T, the observed data is a set of arrival times t, and the theoretical relation gives predicted travel times h. Tarantola and Valette (1982) show that, if the theoretical relationship and the observed arrival times are assumed to have Gaussian uncertainties with covariance matrices $\mathbf{C}_T$ and $\mathbf{C}_t$, respectively, and if the prior information on T is taken as uniform, then it is possible to evaluate analytically the integral over **d** in (2) and an integral over origin time T to obtain the marginal PDF for the spatial location, $\sigma(\mathbf{x})$. This marginal PDF reduces to (Tarantola and Valette, 1982; Moser, van Eck and Nolet, 1992)

$$\begin{aligned} \sigma(\mathbf{x}) &= K\,\rho(\mathbf{x}) \cdot \exp[-\tfrac{1}{2} g(\mathbf{x})] \\ g(\mathbf{x}) &= [\hat{\mathbf{t}}_0 - \hat{\mathbf{h}}(\mathbf{x})]^T (\mathbf{C}_t + \mathbf{C}_T)^{-1} [\hat{\mathbf{t}}_0 - \hat{\mathbf{h}}(\mathbf{x})]. \end{aligned} \qquad (3)$$

In this expression K is a normalisation factor, $\rho(\mathbf{x})$ is a density function of prior information on the model parameters, and $g(\mathbf{x})$ is an L2 misfit function. $\hat{\mathbf{t}}_0$ is the vector of observed arrival times **t** minus their weighted mean, $\hat{\mathbf{h}}$ is the vector of theoretical travel times **h** minus their weighted mean, where the weights wi are given by

$$w_i = \sum_j w_{ij}; \quad w_{ij} = [(\mathbf{C}_t + \mathbf{C}_T)^{-1}]_{ij}. \qquad (4)$$

Furthermore, Moser, van Eck and Nolet, 1992 show that the maximum likelihood origin time corresponding to a hypocenter at (x, y, z) is given by

$$T_{ml}(\mathbf{x}) = \frac{\sum_i \sum_j w_{ij}[t_i - h_i(\mathbf{x})]}{\sum_i \sum_j w_{ij}}. \qquad (5)$$

The posterior density function (PDF) $\sigma(\mathbf{x})$ given by equation (3) represents a complete, probabilistic solution to the location problem, including information on uncertainty and resolution. This solution does not require a linearised theory, and the resulting PDF may be irregular and multi−modal because the forward calculation involves a non−linear relationship between hypocenter location and travel−times.

This solution includes location uncertainties due to the spatial relation between the network and the event, measurement uncertainty in the observed arrival times, and errors in the calculation of theoretical travel times. However, realistic estimates of uncertainties in the observed and theoretical times must be available and specified in a Gaussian form through $\mathbf{C}_t$ and $\mathbf{C}_T$, respectively. Absolute location errors due to incorrect velocity structure could be included through $\mathbf{C}_T$ if the resulting travel time errors can be estimated and described with a Gaussian structure. Estimating these travel time errors is difficult and often not attempted. When the model used for location is a poor approximation to the "true" structure (as is often the case with layered model approximations), the absolute location uncertainties can be very large.

Complete, Non−linear Location − PDF

The NLLoc grid−search algorithm systematically determines the posterior probability density function $\sigma(\mathbf{x})$ or the "misfit" function $g(\mathbf{x})$ over a 3D, x,y,z spacial grid. The NLLoc Metropolis−Gibbs and oct−tree sampling algorithms obtain a set of samples distributed according to the posterior probability density function $\sigma(\mathbf{x})$.

The grid−search $\sigma(\mathbf{x})$ grid, samples drawn from this function, or the samples obtained by the Metropolis−Gibbs sampling, form the full, non−linear spatial solution to the earthquake location problem. This solution indicates the uncertainty in the spatial location due to picking errors, a simple estimate of travel−time calculation errors, the geometry of the observing stations and the incompatibility of the picks. The location uncertainty will in general be non−ellipsoidal (non−Gaussian) because the forward calculation involves a non−linear relationship between hypocenter location and travel−times.

Because it is difficult or impossible to obtain, a more complete estimate of the travel−time errors (or, equivalently, a robust estimate of the errors in the velocity model) is not used. This is a serious limitation of this and most location algorithms, particularly for the study of absolute event locations.

The PDF may be output to a 3D Grid and a binary Scatter file (see Output below). PDF values are also used for the determination of weighted average phase residuals (output to a Phase Statistics file), and for calculating location confidence contour levels (see Output below), and "Traditional" Gaussian estimators (see below).

Maximum likelihood hypocenter

The maximum likelihood (or minimum misfit) point of the complete, non−linear location PDF is selected as an "optimal" hypocenter. The significance and uncertainty of this maximum likelihood hypocenter cannot be assessed independently of the complete solution PDF. The maximum likelihood hypocenter parameters are output to the NNLoc, ASCII Hypocenter−Phase File (`HYPOCENTER`, `GEOGRAPHIC` and `QUALITY` lines), and to the quasi−HYPOELLIPSE format and HYPO71 format files. The maximum likelihood hypocenter is also used for the determination of ray take−off angles (output to a HypoInverse Archive file), for the determination of average phase residuals (output to a Phase Statistics file), and for magnitude calculation. The ray take−off angles can be used for a first−motion fault plane determination.

Gaussian estimators

"Traditional" Gaussian or normal estimators, such as the expectation $E(\mathbf{x})$ and covariance matrix **C** may be obtained from the gridded values of the normalised location PDF or from samples of this function (e.g. Tarantola and Valette, 1982; Sen and Stoffa,1995). For the grid case with nodes at $\mathbf{x}_{i,j,k}$,

$$E(\mathbf{x}) = \Delta V \sum_{i,j,k} \mathbf{x}_{i,j,k} \sigma(\mathbf{x}_{i,j,k}), \qquad (6)$$

where ΔV is the volume of a grid cell. For N samples drawn from the PDF with locations $\mathbf{x}_n$,

$$E(\mathbf{x}) = \frac{1}{N} \sum_{n} \mathbf{x}_n, \qquad (7)$$

where the PDF values $\sigma(\mathbf{x}_n)$ are not required since the samples are assumed distributed according to the PDF. For both cases, the covariance matrix is then given by

$$\mathbf{C} = E[(\mathbf{x} - E(\mathbf{x})) \cdot (\mathbf{x} - E(\mathbf{x}))^T]. \qquad (8)$$

The Gaussian estimators are output to the NNLoc, ASCII Hypocenter−Phase File (`STATISTICS` line).

Confidence Ellipsoid

The 68% confidence ellipsoid can be obtained from singular value decomposition (SVD) of the

covariance matrix **C**, following Press *et al*. (1992; their sec. 15.6 and eqs. 2.6.1 and 15.6.10). The SVD gives:

$$\mathbf{C} = \mathbf{U}[\mathit{diag}\, w_i]\mathbf{V}^{\mathbf{T}}, \qquad (9)$$

where **U** = **V** are square, symmetric matrices and w_i are singular values. The columns **V***i* of **V** give the principle axes of the confidence ellipsoid. The corresponding semi−diameters for a 68% confidence ellipsoid are $(3.53wi)^{½}$, where 3.53 is the $\Delta\chi^2$ value for 68.3% confidence and 3 degrees of freedom.

The Gaussian estimators and resulting confidence ellipsoid will be good indicators of the uncertainties in the location only in the case where the complete, non−linear PDF has a single maximum and has an ellipsoidal form.

Grid−Search Algorithm

The grid−search algorithm performs successively finer, systematic grid−searches within a spatial, *x,y,z* volume to obtain a misfit function, an optimal hypocenter and an estimate of the posterior probability density function (PDF) for hypocenter location.

Advantages:

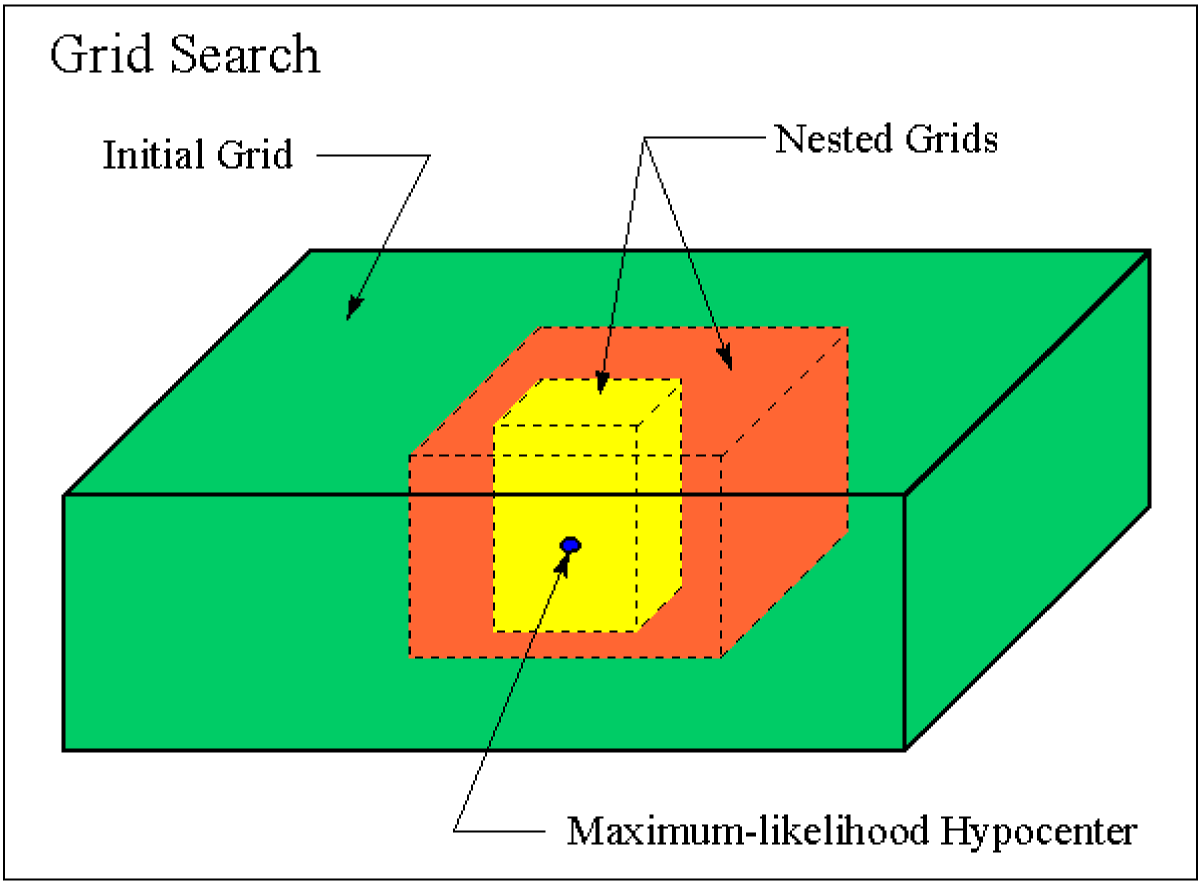
1. Does not require partial derivatives, thus can be used with complicated, 3D velocity structures
2. Systematic, deterministic coverage of search region
3. Accurate recovery of very irregular (non−ellipsoidal) PDF's with multiple minima
4. Efficiently reads into memory 2D planes of 3D travel−time grid files, thus can be used with large number of observations and large 3D travel−time grids
5. Results can be used to obtain confidence contours

Drawbacks:

1. Very time consuming relative to stochastic and linear location techniques
2. Relative to the size of the most significant region of the PDF, the final search grids may be too large (giving low resolution) or too small (giving truncation of the PDF)
3. Requires careful selection of grid size and node spacing

Procedure

The Grid−Search location is based on a nested grid search using one or more location grids as specified by LOCGRID statements in the Input Control File. The first LOCGRID statement specifies a specific initial search grid with fixed size, number of nodes and location. Subsequent LOCGRID statements specify the size and number of nodes for subsequent, nested grids; the location of these nested grids is usually set automatically in one or more of the *x,y,z* directions.



For each location grid, the location quality (misfit or PDF value) at every node is obtained. For each node, the travel−times for each observation are obtained from the corresponding travel−time grid file and the PDF σ(**x**), or misfit value *g*(**x**) is calculated using the equations given above in the Inversion Approach section. These location quality values are saved to a 3D grid file if requested. If there is a subsequent nested grid, its position (for the directions with automatic positioning) is set so that it is centred on the maximum PDF node (or, equivalently, the minimum misfit node) of the current grid.

The initial location grid must be fully contained within the travel−time grid files corresponding to a given observation for that observation to be used in the location. Subsequent location grids, even if their position is set automatically, must be fully contained within the initial grid. The NLLoc program will attempt to translate a nested grid that intersects a boundary of the initial grid so that it is contained inside of the initial grid; if this is not possible the location will be terminated prematurely.

For every node of each location grid, the grid−search algorithm must obtain travel−times for every observation. These times are stored on disk in 3D travel−time grid files which may be very large. It would be extremely time consuming to read these times one by one directly from the disk files, but there is also not enough space in general to fully read all the relevant 3D grid files into memory. However, the grid search is performed systematically throughout each location grid with the *x* index varying last. Thus, it is adequate to have 2D planes or "sheets" corresponding to the current *x* index available in memory at any one time. This approach is used by the grid−search algorithm. Sheets of data with a given *x* index are read from the 3D travel−time grid files as large blocks of bytes, which is very fast in comparison to reading the same number of data values individually.

Metropolis−Gibbs Sampling Algorithm

The Metropolis−Gibbs algorithm performs a directed random walk within a spatial, *x*,*y*,*z* volume to obtain a set of samples that follow the 3D PDF for the earthquake location. The samples give and estimate of the optimal hypocenter and an image of the posterior probability density function (PDF) for hypocenter location.

Advantages:

1. Does not require partial derivatives, thus can be used with complicated, 3D velocity structures
2. Accurate recovery of moderately irregular (non−ellipsoidal) PDF’s with a single minimum
3. Only only moderately slower (about 10 times slower) than linearised, iterative location techniques, and is much faster (about 100 times faster) than the grid−search
4. Results can be used to obtain confidence contours

Drawbacks:

1. Stochastic coverage of search region − may miss important features
2. Inconsistent recovery of very irregular (non−ellipsoidal) PDF’s with multiple minima
3. Requires careful selection of sampling parameters
4. Attempts to read full 3D travel−time grid files into memory, thus may run very slowly with large number of observations and large 3D travel−time grids

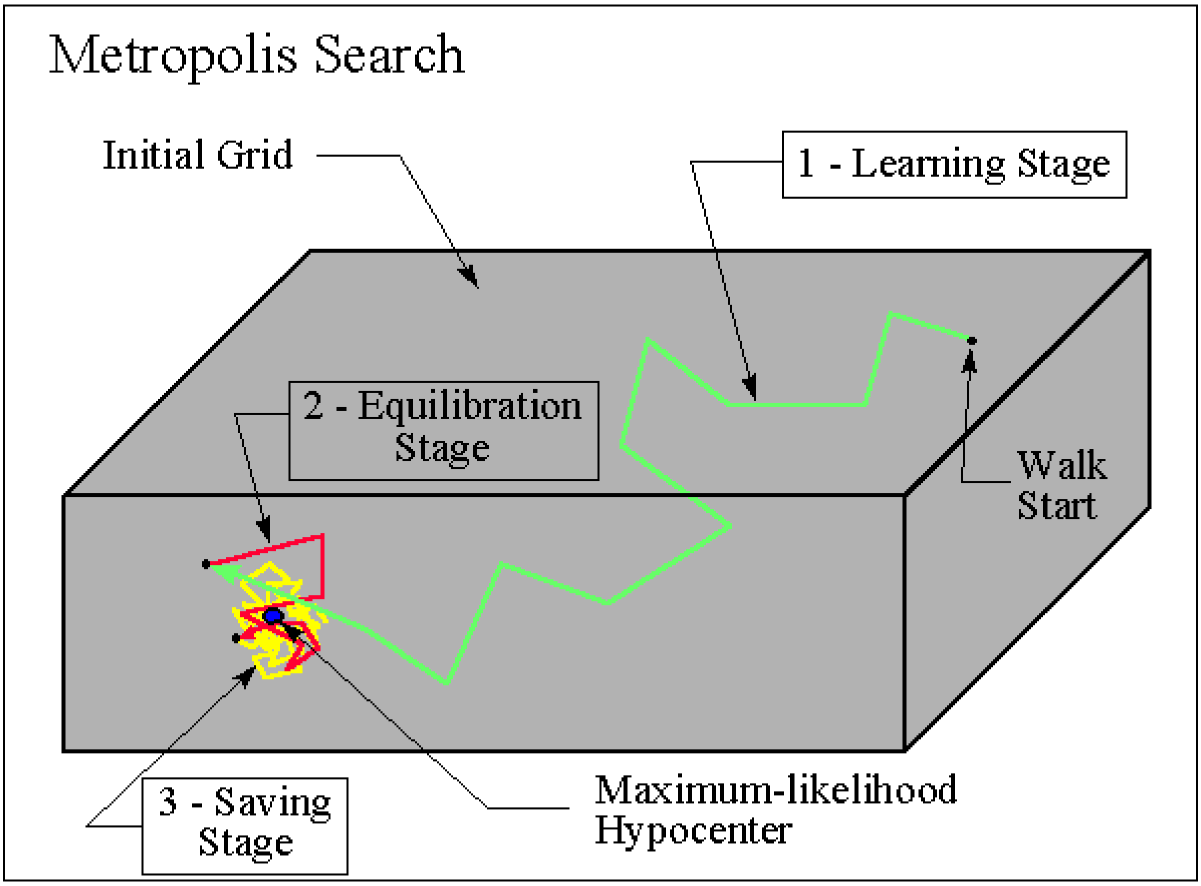
Procedure

The Metropolis−Gibbs search procedure to obtain samples of a PDF is based on the algorithm of Metropolis et al. (1953) for the simulation of the distribution of a set of atoms at a given temperature. The Metropolis−Gibbs algorithm used here is similar to the "Metropolis" algorithm described in Mosegaard and Tarantola (1995) and the "Gibbs sampler" with temperature T=1 described in Sen and Stoffa (1995; sec 7.2). It may be considered as a version of Metropolis simulated annealing Kirkpatrick et al. (1983) where the temperature parameter is a constant determined by the covariance matrix for the observational and forward problem uncertainties. Thus the algorithm does not "anneal" or converge to an optimal solution, but instead produces a set of samples which follow the posterior PDF for the inverse problem.

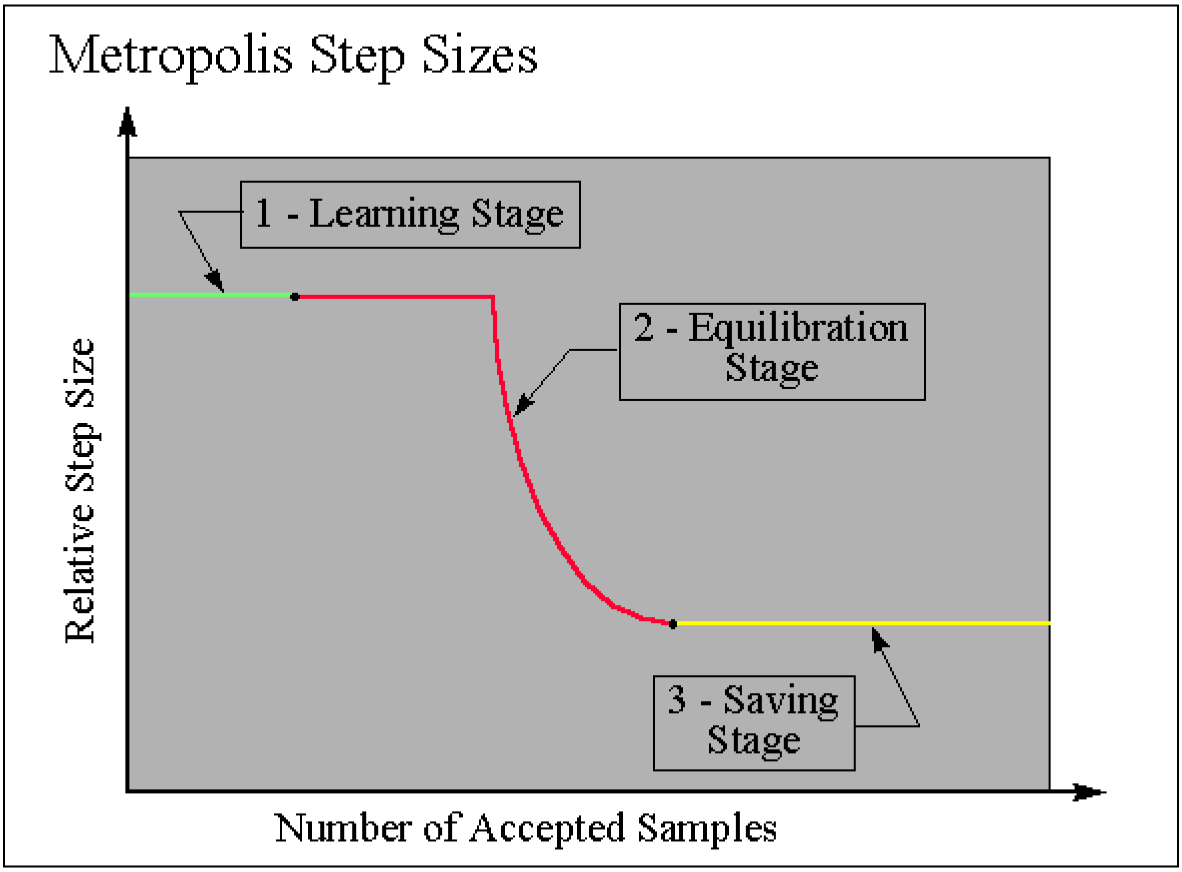
The Metropolis−Gibbs sampler used in the program NonLinLoc for earthquake location consists of a directed walk in the solution space (*x*, *y*, *z*) which tends towards regions of high likelihood for the location PDF, $\sigma(\mathbf{x})$ given by equation (3). At each step, the current walk location $\mathbf{x}_{curr}$ is perturbed by a vector $d\mathbf{x}$ of arbitrary direction and given length l to give a new location $\mathbf{x}_{new}$. The likelihood $\sigma(\mathbf{x}_{new})$ is calculated for the new location and compared to the likelihood $\sigma(\mathbf{x}_{curr})$ at the current location. If $\sigma(\mathbf{x}_{new}) \geq \sigma(\mathbf{x}_{curr})$, then the new location is accepted. If $\sigma(\mathbf{x}_{new}) < \sigma(\mathbf{x}_{curr})$, then the new location is accepted with probability $P = \sigma(\mathbf{x}_{new}) / \sigma(\mathbf{x}_{curr})$. When a new location is accepted it becomes the current location and may be saved as a sample of the location PDF.

In earthquake location, the dimensions of the significant regions of the location PDF can vary enormously and are not known a priori. It is important to choose an initial step size large enough to allow global exploration of the search volume, and to obtain a final step size that gives good coverage of the location PDF while resolving details and irregular structure of the PDF. The NonLinLoc Metropolis−Gibbs sampler uses three distinct sampling stages to determine adaptively an optimal step size l for the walk:

1. A **learning** stage where the step size is fixed and relatively large. The walk can explore globally the search volume and migrate towards regions of high likelihood. "Accepted" samples are not saved.
2. An **equilibration** stage where the step size l is adjusted in proportion to the standard deviations (s_x, s_y, s_z) of the spatial distribution of all previously "accepted" samples obtained after the middle of the learning stage. After each new accepted sample, the standard deviations are updated and the step size l is set equal to $f_s\ (s_x s_y s_z / N_s)^{1/3}$, where N_s is the number of previously "accepted" samples, and f_s=8 is a step size scaling factor. This formula sets l in proportion to the cell size required to tile with Ns cells the rectangular volume with sides s_x, s_y and s_z. The walk can continue to migrate towards or may begin to explore regions of high likelihood. "Accepted" samples are not saved.
3. A **saving** stage where the step size l is fixed at its final value from the equilibration stage. The walk can continue to explore regions of high likelihood. "Accepted" samples are assumed to follow the location PDF and can be saved, but there may be a waiting time of several samples between saves to insure the independence of saved samples.



It is important to set the parameters for the directed walk so that (1) during the **learning** and **equilibration** stages the walk approaches and reaches the high likelihood regions of the location PDF, and so that (2) by the **saving** stage a suitable, relatively small, fixed step size has been obtained to accurately explore and image the PDF.



The NonLinLoc Metropolis−Gibbs sampling algorithm is initialised as follows:

1. The walk location is set at the *x*,*y* position of the station with the earliest arrival time and non−zero weight, at the mean depth of the search region.
2. If the initial step *l* size is not specified, it is set to the cell size required to tile with N_s cells the plane formed by the two longest sides of the initial search region. N_s is the total number of samples to be accepted during the saving stage, including samples that are skipped between saves.

The rejection by the algorithm of new walk locations for a large number of consecutive tries (the order of 1000 tries) may indicate that the last "accepted" sample falls on a sharp likelihood maxima that is narrower than the current step size. To allow the search to continue in this case, the new location is accepted unconditionally and the step size is reduced by a factor of two.

In the case that the size of the location PDF is very small relative to the search region, the algorithm may fail to locate the region of high likelihood or obtain an optimal step size. In this case the size of the search region must be reduced or the size of the initial step size adjusted. A more robust solution to this problem may be to add a temperature parameter to the likelihood function, as with simulated annealing. This variable parameter could be set to increase the effective size of the PDF during the learning and equilibration stages so that the region of high likelihood is located efficiently, and then set to 1 during the saving stage so that the true PDF is imaged.

Oct−tree Importance Sampling Algorithm

Developed in collaboration with Andrew Curtis; Schlumberger Cambridge Research, Cambridge CB3 0EL, England; curtis@cambridge.scr.slb.com

The oct−tree importance sampling algorithm gives accurate, efficient and complete mapping of earthquake location *PDF*s in 3D space (x−y−z).

Advantages:

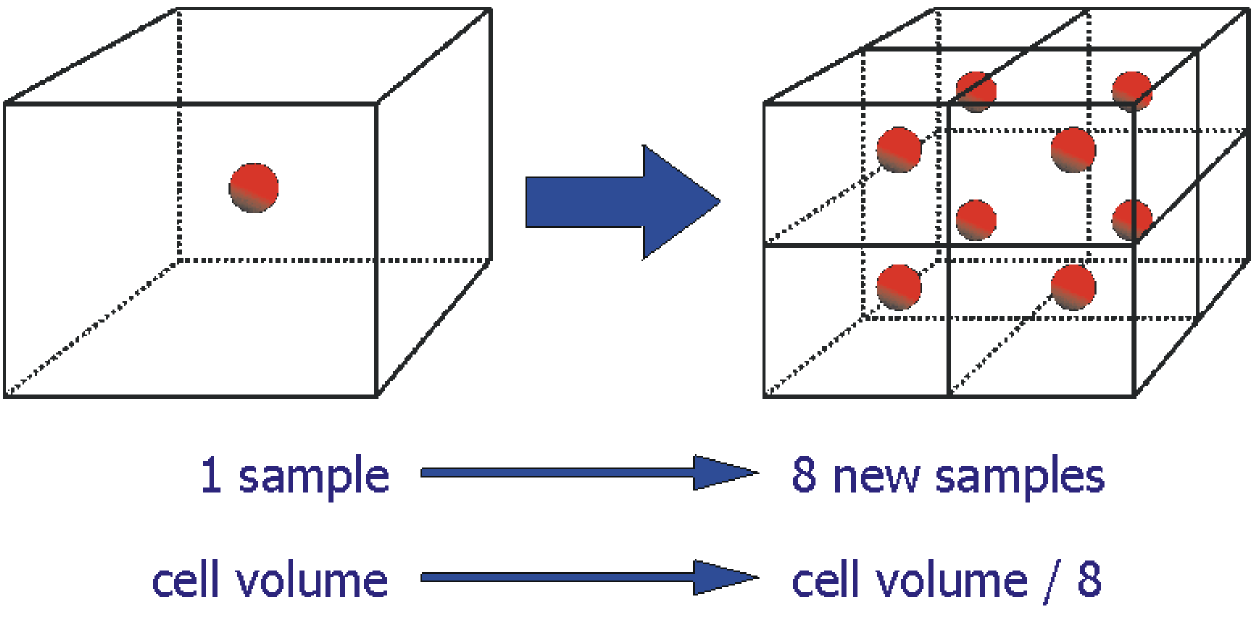
1. Much faster than grid−search (factor 1/100)
2. More global and complete than Metropolis−simulated annealing
3. Simple, with very few parameters (initial grid size, number of samples)

Drawbacks:

1. Results are weakly dependant on initial grid size − the method may not identify narrow, local maxima in the PDF.
2. Attempts to read full 3D travel−time grid files into memory, thus may run very slowly with large number of observations and large 3D travel−time grids

Procedure

The oct−tree method uses recursive subdivision and sampling of cells in 3D space (*below*) to generate a cascade of sampled cells, where the density of sampled cells follows the *PDF* values of the cell centre.

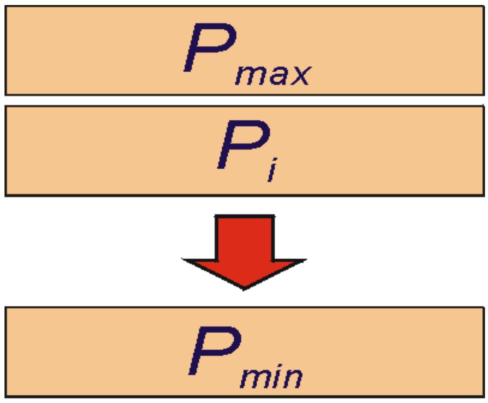


The probability that the earthquake location is in a given cell *i* is approximately,

$P_i = V_i \, PDF(\mathbf{x}_i)$

where V_i is the cell volume and $\mathbf{x}_i$ are the co−ordinates of the cell centre.

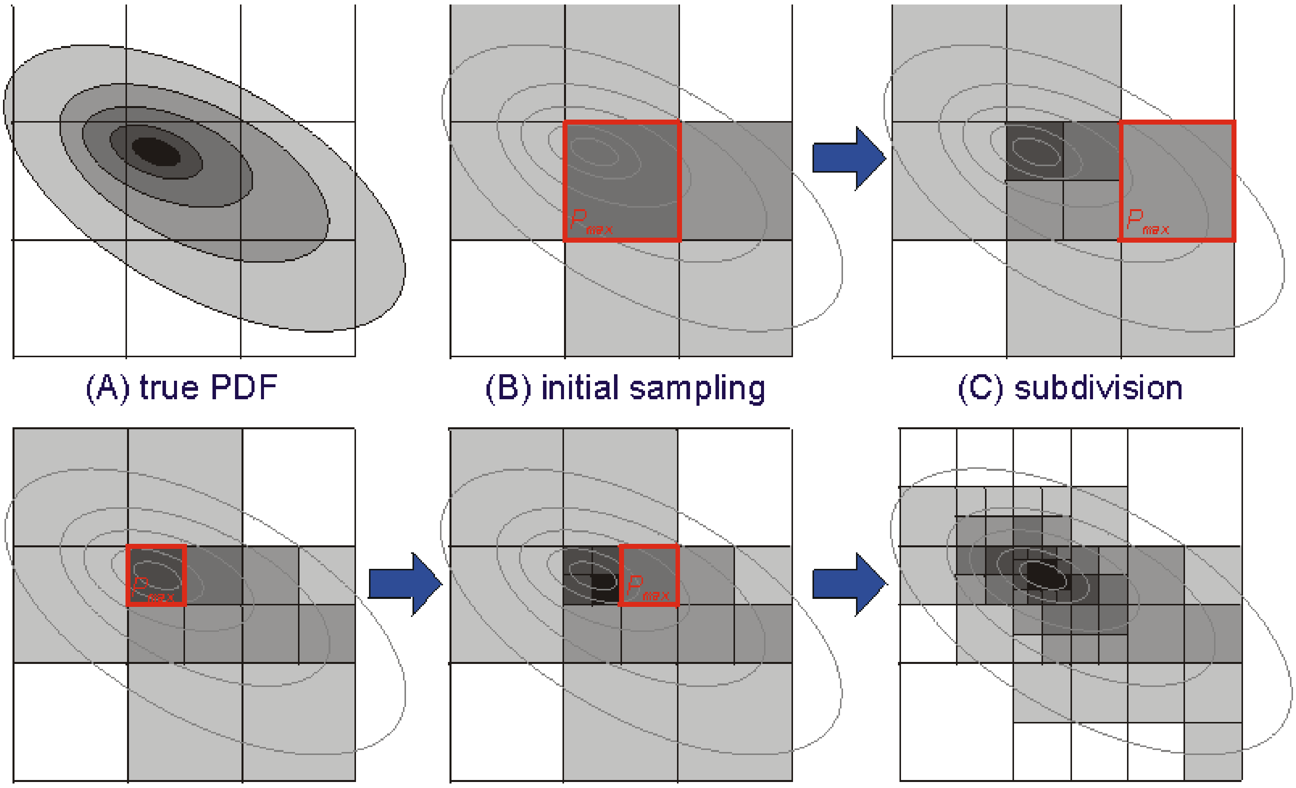
The core of the method is an ordered list L_P of probability values P_i for all previously sampled cells:



The oct−tree sampling procedure is initialised by a global sampling of the full search space on a coarse, regular grid (**B** *below*). The misfit value $g_i(\mathbf{x})$ at the centre of each grid cell is determined, the probability P_i is calculated, and the cell is inserted in the probability list L_P at the position corresponding to its probability P_i.

Next the following steps are repeated (**C−E** *below*) until a predetermined number of evaluations of the forward problem or other termination criterion has been reached:

1. The cell C_{max} with the largest probability P_{max} (red squares *below*) is obtained from the ordered list L_P
2. *Cmax* is divided into 8 child−cells
3. The misfit and probability *Pi* are calculated for each of the 8 child−cells
4. The 8 new cells are inserted into the ordered list L_P according to their P_i
5. Repeat

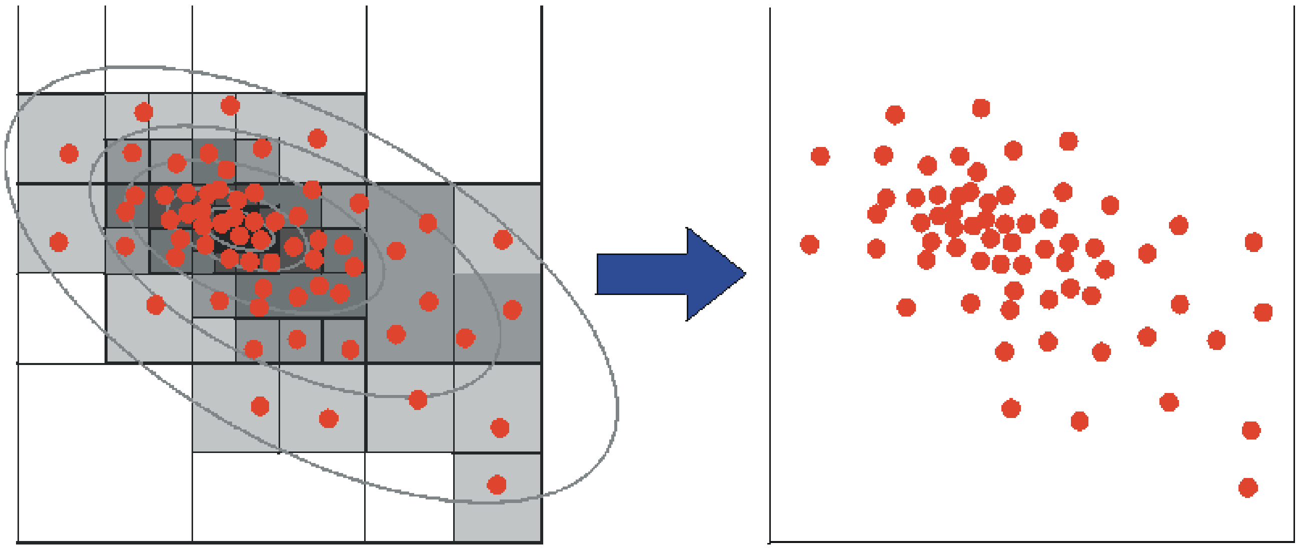


(A) true PDF (B) initial sampling (C) subdivision

(D) subdivision (E) subdivision (F) many subdivisions

This recursive procedure converges rapidly, producing an oct−tree structure of cells specifying location PDF values in 3D space (**F** *above*). This oct−tree structure will have a larger number of cells in the regions of higher *PDF* (lower misfit) and thus gives approximate importance sampling of the *PDF* (**A** above).

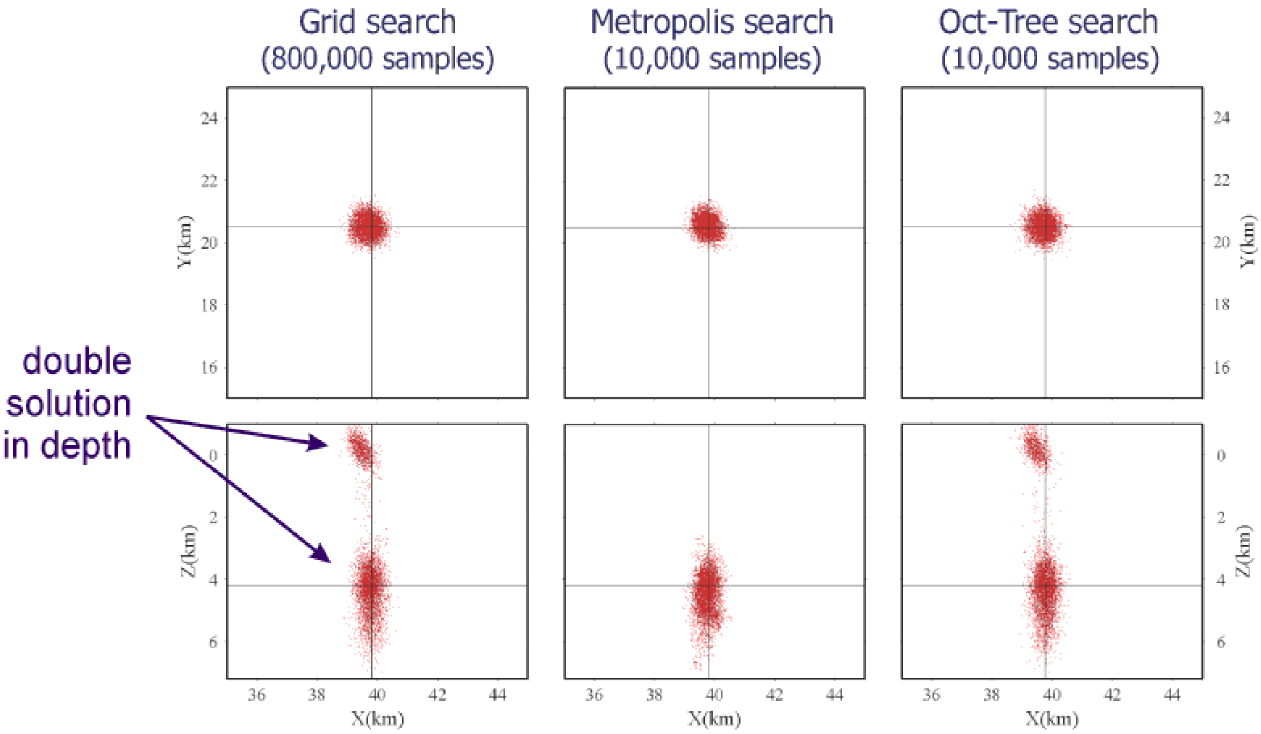
Finally, samples in 3D drawn from the oct−tree structure (*below*) give a useful and compact representation of the *PDF*.



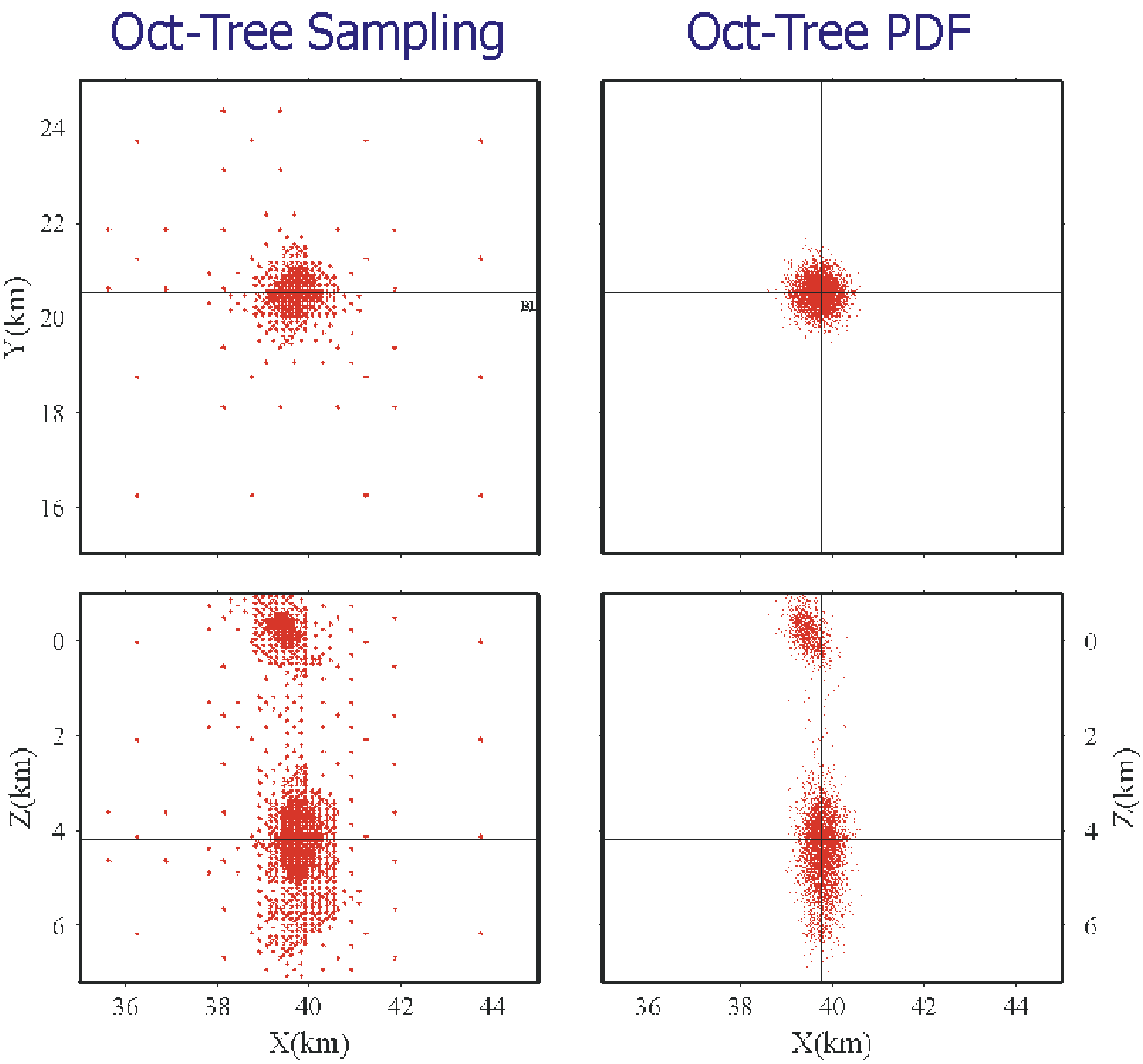
The example below is illustrated by 2D projections of such 3D samples.

Example − An earthquake location with a double solution

An exhaustive grid−search (*below, left*) shows the complete *PDF* for the location; this *PDF* shows two distinct regions of high probability at different depths. The oct−tree method (*right*) identifies both solution volumes and is thus more complete than a Metropolis−simulated annealing approach (*centre*), which identifies only the deeper solution. Both of these methods are about 100 times faster than the grid search, but only the oct−tree method produces an image of the solution *PDF* that is nearly identical to that of the exhaustive grid−search. The oct−tree and Metropolis−simulated annealing methods are only about 10 times slower than standard linearised location algorithms, which are difficult or impossible to apply with 3D models.



An image of all cell centres visited by the oct−tree method (*blow, left*) shows that this method samples globally while producing efficient importance sampling: *i.e.* The distribution of cell centres follows closely the distribution of samples of the final *PDF* (*right*).



Discussion

The oct−tree method performs well for the earthquake location problem because it is 3D. This allows the use of the simple geometry of oct−tree division of rectangular cells: the volume of each cell is always known and it can be determined easily which cell contains a given point. The oct−tree method should be applicable in 4D, allowing a search over origin time. But in higher dimensional problems the determination of the volume of a cell and whether or not a cell contains a given point may become difficult or impossible.

The oct−tree approach can be applied to teleseismic location in a spherical earth by a) performing the location in a cubic region containing the spherical earth, or b) dividing the spherical earth into organised curvilinear cells which can be further sub−divided into 8 child cells.

Running the program

Input

Synopsis: `NLLoc` *`InputControlFile`*

The NLLoc program takes a single argument *`InputControlFile`* which specifies the complete path and filename for an Input Control File with certain required and optional statements specifying program parameters and input/output file names and locations. See the NLLoc Statements section of the Input Control File for more details. Note that to run NLLoc the Generic Statements section of the Input Control File must contain the `CONTROL` and `TRANS` (Geographic Transformation) statements.

In addition, the NLLoc program requires:

1. A file or files containing sets of seismic phase arrival times for each event. These arrival times can be can be specified in a number of Phase formats, including those of the HYPO71/HYPOELLIPSE and SEISAN software, and the RéNaSS DEP format.
2. Files containing a 2D or a 3D **Travel−time grid** created by the program Grid2Time for each phase type at each station. If a constant Vp/Vs ratio is used, then only P travel−time grids are required for each station.

The names, locations and other information for these files is specified in the NLLoc Statements section of the Input Control File.

Output

The location results can be output for **single event** and **summary** (all events) as:

1. A 3D Grid containing **misfit values** or **PDF*** **(probability density function)** values throughout the search volume (Grid−search only).
2. An ASCII Hypocenter−Phase File containing **hypocentral co−ordinates and origin time** for the best **(minimum misfit / maximum likelihood)** point in the the search volume and an associated **phase list**! containing station and phase identifiers, phase times, residuals, take−off angles and other station/phase information. This file contains other information, including the **hypocentral co−ordinates and uncertainty*** given by the traditional (Gaussian/Normal) **expectation** and **covariance matrix** measures of the PDF.
3. A binary Scatter file containing samples drawn from the PDF
4. An ASCII Confidence Levels giving the value of the PDF corresponding to confidence levels from 0.1 to 1.0

* these output types are only generated for grids where the PDF is calculated.
! these output types are only written to single event files

The location results can also be output as **summary** (all events) files containing:

1. A 3D Grid header file describing the search volume
2. ASCII Phase Statistics giving the mean residuals for P and S phases at each station
3. An expanded, quasi−HYPOELLIPSE format
4. The HypoInverse Archive format which serves as input to the program FPFIT (Reasenberg *et al.*, 1985) for grid−search determination of focal mechanism solutions.

Single event and summary files are only saved for specific nested search−grids as specified in the LOCGRID statement in the Input Control File.

Processing and Display of results

The location results for one or more events can be combined with the program LocSum to produce output such as a comprehensive, summary Hypocenter−Phase File, a binary Scatter File, and a set of simple ASCII format Scatter samples files.

The comprehensive, summary Hypocenter−Phase File forms the input for the Java program SeismicityViewer for interactive, 3D display of event locations.

The location results for a single event or the output files produced by the program LocSum can be post−processed with the program Grid2GMT to produce a GMT command script for plotting misfit, PDF and location "cloud" results using the GMT plotting package.

12. LocSum Program − combine location results

LocSum combines NLLoc location results and PDF "scatter−cloud" samples from a number of events.

Running the program−Input − Output

Overview

The LocSum utility combines single event NLLoc location files (Hypocenter−Phase files and binary Scatter files) into a single set of summary location files.

For flexibility, the LocSum utility takes most of its parameters from the command line.

Running the program

Input

The LocSum utility takes several of command line arguments.

Synopsis: `LocSum` *`SizeGridRoot decimFactor OutRoot LocRoot [Len3Max [ProbMin [RMSMax [NRdgsMin [GapMax]]]]]`*

Parameters:

`SizeGridRoot` (*chars*)
: full or relative path and *root* name (no extension) for a 3D Grid Header file. The grid dimensions in this header file are used to create an empty grid buffer and new grid header file with root name *`OutRoot`*.

`decimFactor` (*integer*)
: decimation factor (*`decimFactor`* `> 0`) for decimating the number of PDF Scatter samples. Every *`decimFactor`*−th sample is saved to the output files.

`OutRoot` (*chars*)
: full or relative path and root name for output files.

`LocRoot` (*chars*)
: full or relative path and root name (no extension) for one or more NLLoc single Event Location files. Multiple root names may be specified using standard UNIX "wild−card" characters (* and ?); however, if any "wild−card" characters are used then the path and root name must be enclosed in double quotes (") to prevent the

shell from evaluating the "wild−card" characters.

`Len3Max` (*float*)
: maximum length in kilometres of the longest ellipsoid semi−axis at maximum likelihood hypocenter.

`ProbMin` (*float*)
: minimum value of probability at maximum likelihood hypocenter.

`RMSMax` (*float*)
: maximum RMS in seconds at maximum likelihood hypocenter.

`NRdgsMin` (*integer*)
: minimum number of readings used for location.

`GapMax` (*float*)
: maximum azimuth gap in degrees at maximum likelihood hypocenter.

Notes:

1. See the Definitions section of the NonLinLoc Control File documentation for more information on datatypes.

Example:

1. `LocSum dursum0 1 dursum "dur.*.*.grid0.loc"`

   Using an existing 3D Grid Header file `dursum0.hdr` to determine the grid size, creates a dummy grid buffer file, a grid header file, a set of summary Hypocenter−Phase files, binary Scatter files, and a set of ASCII Scatter files for each location in the current directory with root name `"dur.*.*.grid0.loc"`. The output files are written to the root name `dursum`. The scatter samples are not decimated since *`decimFactor`* = `1`.

Output

The LocSum utility creates the following files:

1. A summary Hypocenter−Phase file named *`OutRoot`*`.hyp`. This file includes `SCATTER` blocks.
2. A summary binary Scatter file named *`OutRoot`*`.scat`.
3. A set of summary ASCII Scatter files for *x−y*, *x−y* and *z−y* projections, named *`OutRoot`*`.scat.`*`ext`*, where *`ext`* = `XY, XZ, ZY` for sample locations in kilometres and *`ext`* = `longlat.XY, longlat.XZ, longlat.ZY` for sample locations in degrees of latitude and longitude and depth in kilometres. These ASCII formats are compatible with the GMT plotting package.
4. A 3D Grid Header file named *`OutRoot`*`.hdr` and an empty 3D Grid Buffer file named *`OutRoot`*`.buf`. These file are created to insure compatibility with post−processing programs and utilities.

13. Grid2GMT Program − location results or 3D grid data to GMT command script

Grid2GMT writes ASCII, GMT command scripts which will generate postscript plots of event location results and sections of 3D grid files.

Running the program−Input − Output

Overview

The Grid2GMT utility generates and runs an ASCII, GMT command script for plotting sections and plan views of 3D grid files and location misfit, PDF and "scatter−cloud" results. When run, the GMT command script creates a postscript file which can be viewed on screen or printed. Running the GMT command script requires the installation of the GMT plotting package.

For flexibility, the Grid2GMT utility takes most of its parameters from the command line.

The default GMT script plots the views in the transformed X−Y co−ordinates. Horizontal (plan) view only plots can also be plotted in geographic lat−long co−ordinates. See the notice and instructions in the *.gmt script file generated by Grid2GMT for more details.

Grid2GMT can be somewhat cumbersome and confusing to use because it combines several functions (i.e. plotting sections and location results), it requires several paths and filenames, and it does not have as sophisticated error checking as other NonLinLoc programs.

Running the program

Input

The Grid2GMT utility takes a number of command line arguments and has three basic modes − (1) Vertical cross section (**`V`**), (2) Horizontal section (**`H`**), and (3) Location (**`L`**) which generates a horizontal section and two vertical sections through the maximum likelihood point of an event location.

If a GMT−cpt file named `Grid2GMT.cpt` is present in the directory in which the GMT script is run, it will be used for the plot and scalebar contours levels and colours for all plot types that use contouring. **(ver 2.0)**

If a GMT script file named `Grid2GMT.?.gmt`, where `? = H, X, Y` or `V` is present in the directory in which the main GMT script is run, it will be run for the corresponding view (`H, X, Y` or `V`)after all standard GMT script is run for that view. **(ver 2.0)**

Synopsis:

Vertical cross section (**`V`**):

`Grid2GMT` *`InputControlFile GridRoot Outroot`* **`V`** *`PlotType iX1 iY1 iX2 iY2`*

or, Vertical cross section lat/long (**`VL`**): **(ver 2.0)**

`Grid2GMT` *`InputControlFile GridRoot Outroot`* **`VL`** *`PlotType lat1 long1 lat2 long2`*

or, Horizontal section (**`H`**):

`Grid2GMT` *`InputControlFile GridRoot Outroot`* **`H`** *`PlotType iZ`*

or, Location (**`L`**, horizontal section and 2 vertical sections):

`Grid2GMT` *`InputControlFile GridRoot Outroot`* **`L`** *`PlotType`*

Parameters: (general)

`InputControlFile` (*chars*)
: specifies the complete path and filename for an Input Control File with certain required and optional statements specifying program parameters and file names. To run Grid2GMT the Generic Statements section of the Input Control File must contain the `CONTROL` and `TRANS` (Geographic Transformation) statements; it may optionally contain one or more `MAPLINE` (Geographic Mapline) statements; and may contain a `MAPTRANS` (Geographic Transformation for Grid2GMT plotting), or `MAPGRID` (Grid region for Grid2GMT plotting), statements **(ver 2.3)**. If maplines are specified, they will be transformed if necessary and plotted on horizontal views. If a `MAPTRANS` statement is specified, it will be used for plotting, regardless if a `TRANS` statement is present in the control file. If a `MAPGRID` statement is specified, it will be used for plotting, regardless if any `LOCGRID` specification is given in any hypocenter. files.

`GridRoot` (*chars*)
: full or relative path and *root* name (no extension) for a 3D Grid (modes **`V`** or **`H`**), or for an event location (mode **`L`**). Multiple root names may be specified using standard UNIX "wild−card" characters (`*` and `?`); however, if any "wild−card" characters are used then the path and root name must be enclosed in double quotes (") to prevent the shell from evaluating the "wild−card" characters.

`Outroot` (*chars*)
: full or relative path and optional root name for output and temporary files.

`PlotType`
: type of data to plot, options are:

`G` − Plot gridded data as a contoured, colour grid. For locations (**`L`**), also plots location statistics.

`S` − Plot location "scatter−cloud" results. For locations (**`L`**), also plots location statistics.

`E`*`nnndx`* − Plot location statistics (Maximum Likelihood location and/or traditional, Gaussian/Normal Expectation and Confidence Ellipsoid). *`nnndx`* is a code of: three *integers* with values *`n`* = `0` (no) or `1` (yes) specifying which statistics to plot (Maximum Likelihood point, Expectation point and Confidence Ellipsoid, respectively), and an optional *float* *`dx`* which specifies a maximum distance in km between the Expectation and Maximum Likelihood points allowed for each event to be plotted.

`M` **(ver 2.3)** − Plot focal mechanisms.

`R`*`phases/scale`* **(ver 2.3)** − Plot phase residuals as symbols with a size proportional to residual magnitude and with type and color dependent on phase. *`phases`* is a set of phase codes, i.e. `P` or `SnSg` specifying which phases to plot. *`scale`* (*float*) specifies the relative size of the residual symbols.

Parameters: (Vertical cross section − `V`)

`iX1 iY1 iX2 iY2` (*integers*)
the minimum and maximum grid indices (`0` to *Num*−`1`, where *Num* is the number of grid nodes) for the *x* and *y* directions. This option plots sections parallel to the co−ordinate axes and oblique sections **(ver 2.0)**. Set *`iX1=iX2`* and *`iY1<iY2`* for a section in the *y*,*z* plane, or *`iY1=iY2`* and *`iX1<iX2`* for a section in the *x*,*z* plane. The full grid range in the *z* direction is always plotted.

Parameters: (Vertical cross section lat/long − `VL`) (ver 2.0)

`lat1 long1 lat2 long2` (*floats*)
the latitude and longitude of the left (1) and right (2) end of the cross section (all lat/long locations must be within the 3D grid) . This option plots sections parallel to the co−ordinate axes and oblique sections. The lat/long locations are converted to grid locations using the transformation specified in the `TRANS` statement of the *`InputControlFile`*. The full grid range in the *z* direction is always plotted.

Parameters: (Horizontal cross section − `H`)

`iZ` (*integer*)
the grid index (`0` to *zNum*−`1`, where `zNum` is the number of grid nodes) for the section level in the *z* direction. Currently this option only plots sections parallel to the *x*,*y* co−ordinate axes. The full grid range in the *x* and *y* directions is always plotted.

Parameters: (Location − `L`)

(*no additional parameters*)
The full grid range in the *x*, *y* and *z* directions is always plotted.

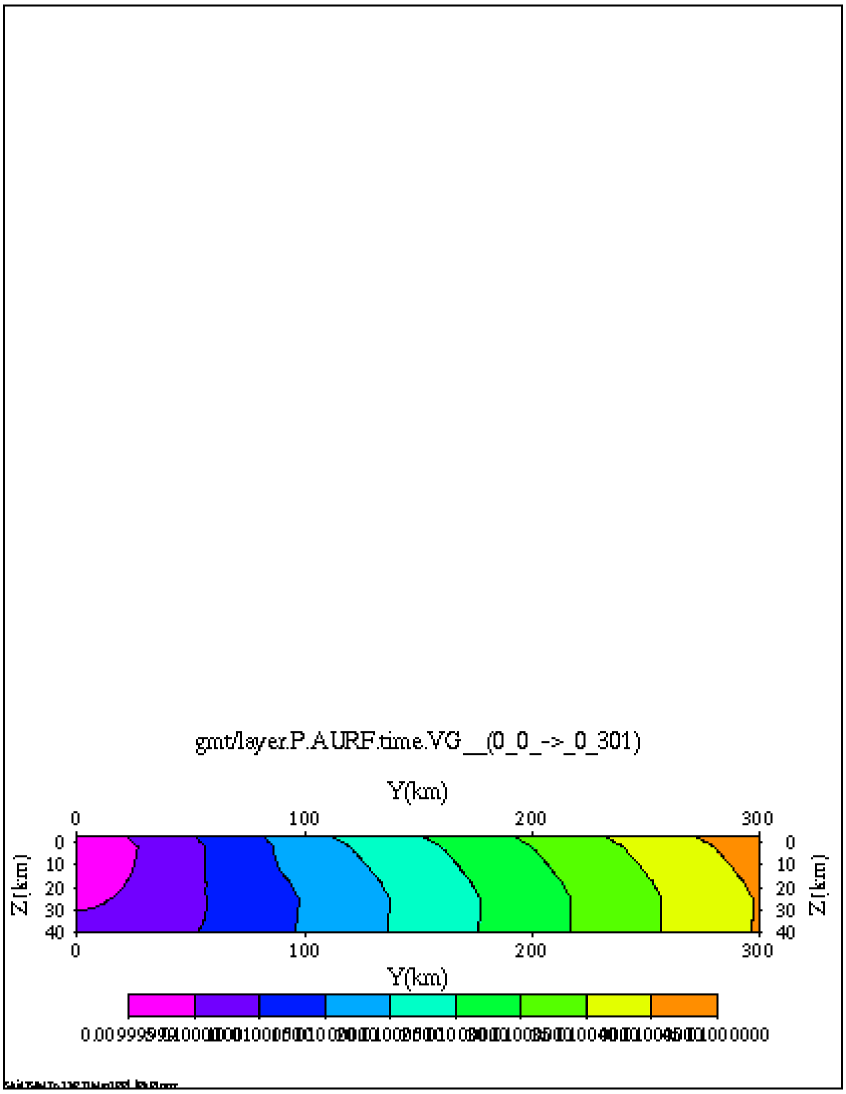
Notes:

1. See the Definitions section of the NonLinLoc Control File documentation for more information on datatypes.
2. Grid2GMT uses cm units, thus for proper execution of the GMT command script, the MEASURE_UNIT entry in the .gmtdefaults file must be set to cm (i.e. `MEASURE_UNIT = cm`)
3. The *x*, *y* and *z* co−ordinate limits and the number of grid nodes is determined by the grid origin and dimensions in the grid header file corresponding to the *`GridRoot`* root name. For plotting event locations which have no corresponding grid files, create a 3D Grid header file named (*`GridRoot`*`.hdr`) with the desired grid origin and dimensions.
4. The plotted ellipsoids are 2D projections of the Confidence Ellipsoids as calculated by the NLLoc location program (see Inversion section of the NLLoc program documentation); the 2D projections DO NOT represent the 2D, marginal distributions of the 3D Confidence Ellipsoids.

Examples:

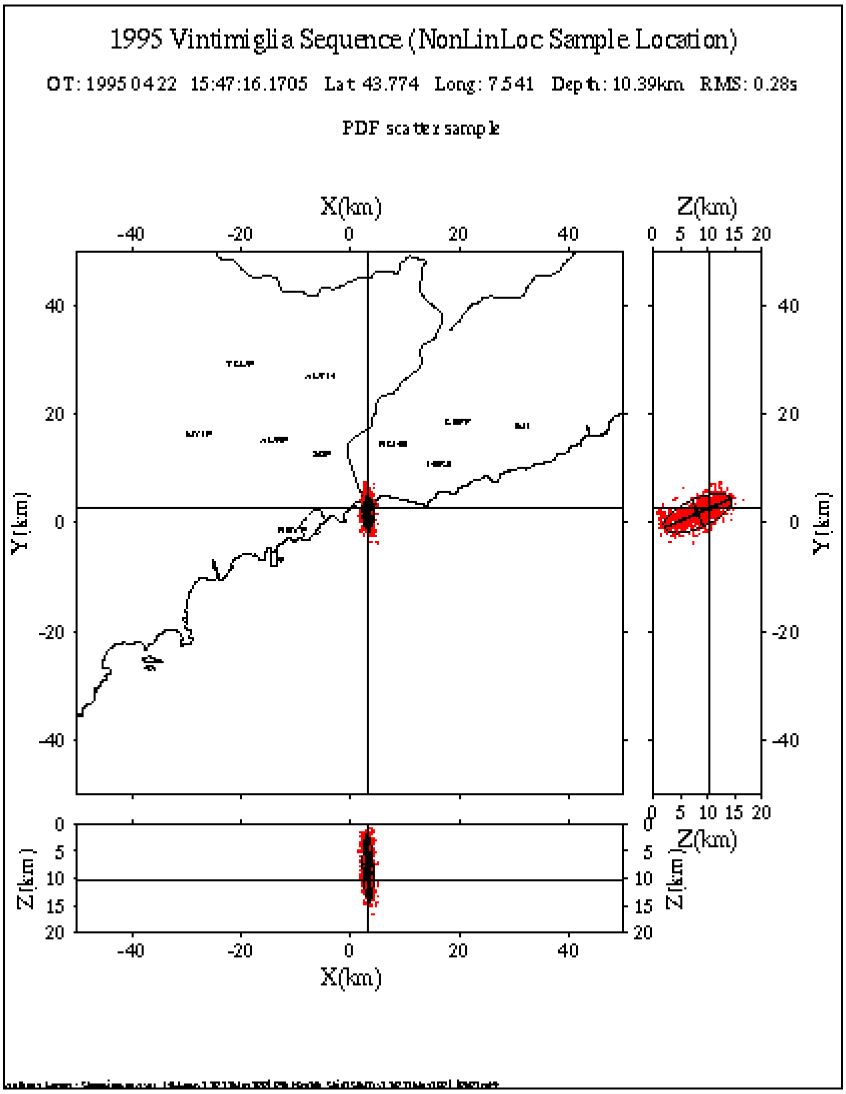
1. `Grid2GMT run/nlloc_sample.in time/layer.P.AURF.time gmt/ V G 0 0 0 301`

   Using Input Control File `run/nlloc_sample.in`, creates a GMT script for the 3D grid travel−time file with root name `time/layer.P.AURF.time`. The GMT script and temporary files are written to directory `./gmt/`. The script will plot a Vertical cross section (`V`), showing the travel−time grid (`G`) for grid nodes = `0` in the *x* direction and between grid nodes `0` and `301` in the *y* direction. (HYPERLINK "file:///zip/nlloc/soft2.30/Grid2GMT_ex1.gif"View resulting plot)



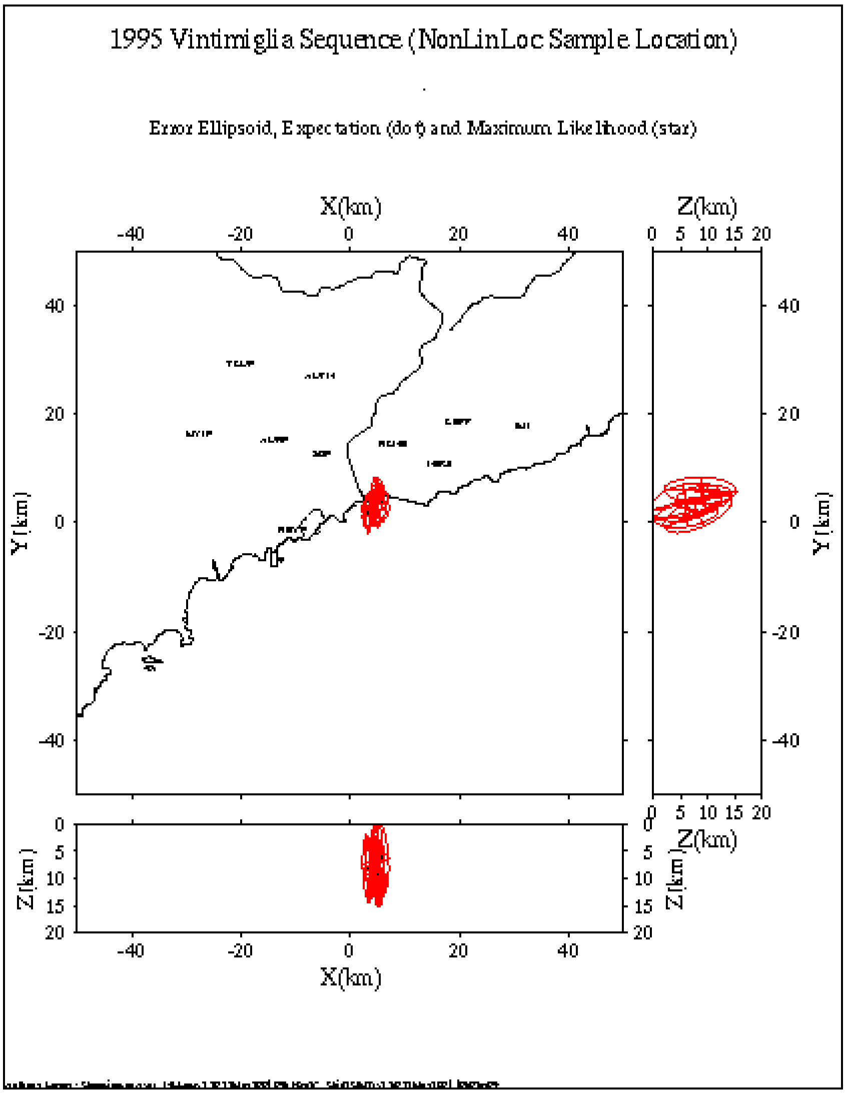
2. `Grid2GMT run/nlloc_sample.in "loc/vinti.*.*.grid0.loc" gmt/ L S`

   Using Input Control File `run/nlloc_sample.in`, creates a GMT script for each location with root name `"loc/dur.*.*.grid0.loc"`. The GMT script and temporary files are written to directory `./gmt/`. For each location, the script will plot a Location (`L`, horizontal section and 2 vertical sections), showing the location PDF "scatter−cloud" samples (`S`). (HYPERLINK "file:///zip/nlloc/soft2.30/Grid2GMT_ex2.gif"[View one of the resulting plots](file:///zip/nlloc/soft2.30/Grid2GMT_ex2.gif))



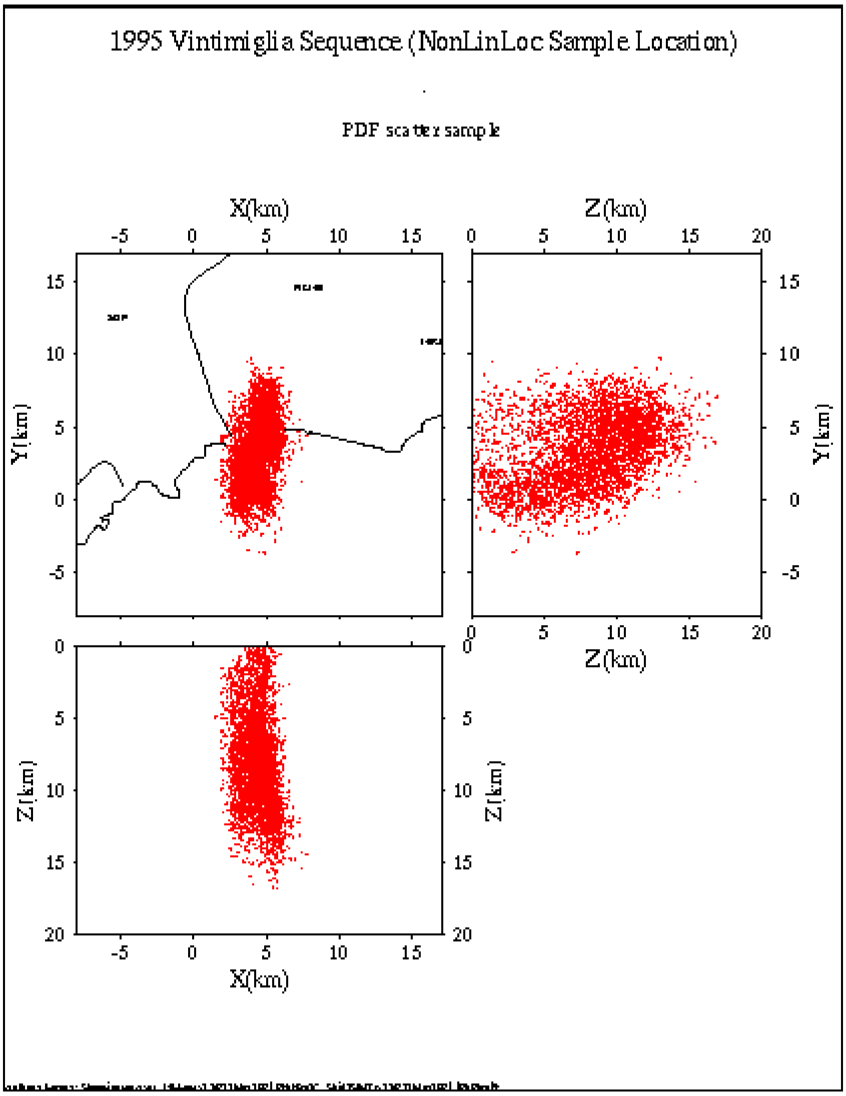
3. `Grid2GMT run/nlloc_sample.in loc/vinti.sum.grid0.loc gmt/ L E111`

   Using Input Control File `run/nlloc_sample.in`, creates a GMT script for the summary hypocenter file with root name `loc/vinti.sum.grid0.loc`. The GMT script and temporary files are written to directory `./gmt/`. The script will plot a Location (`L`, horizontal section and 2 vertical sections), showing all location statistics (`E111`, Maximum Likelihood location and traditional, Gaussian/Normal Expectation and projection of Confidence Ellipsoid), with no restriction on the maximum distance between the Expectation and Maximum Likelihood points (*dx* omitted). Note that for the summary hypocenter files there are no corresponding grid files, thus a [3D Grid] header file named `dur.sum.grid0.loc.hdr` with the desired grid origin and dimensions had to be created. (HYPERLINK "file:///zip/nlloc/soft2.30/Grid2GMT_ex3.gif"[View resulting plot])



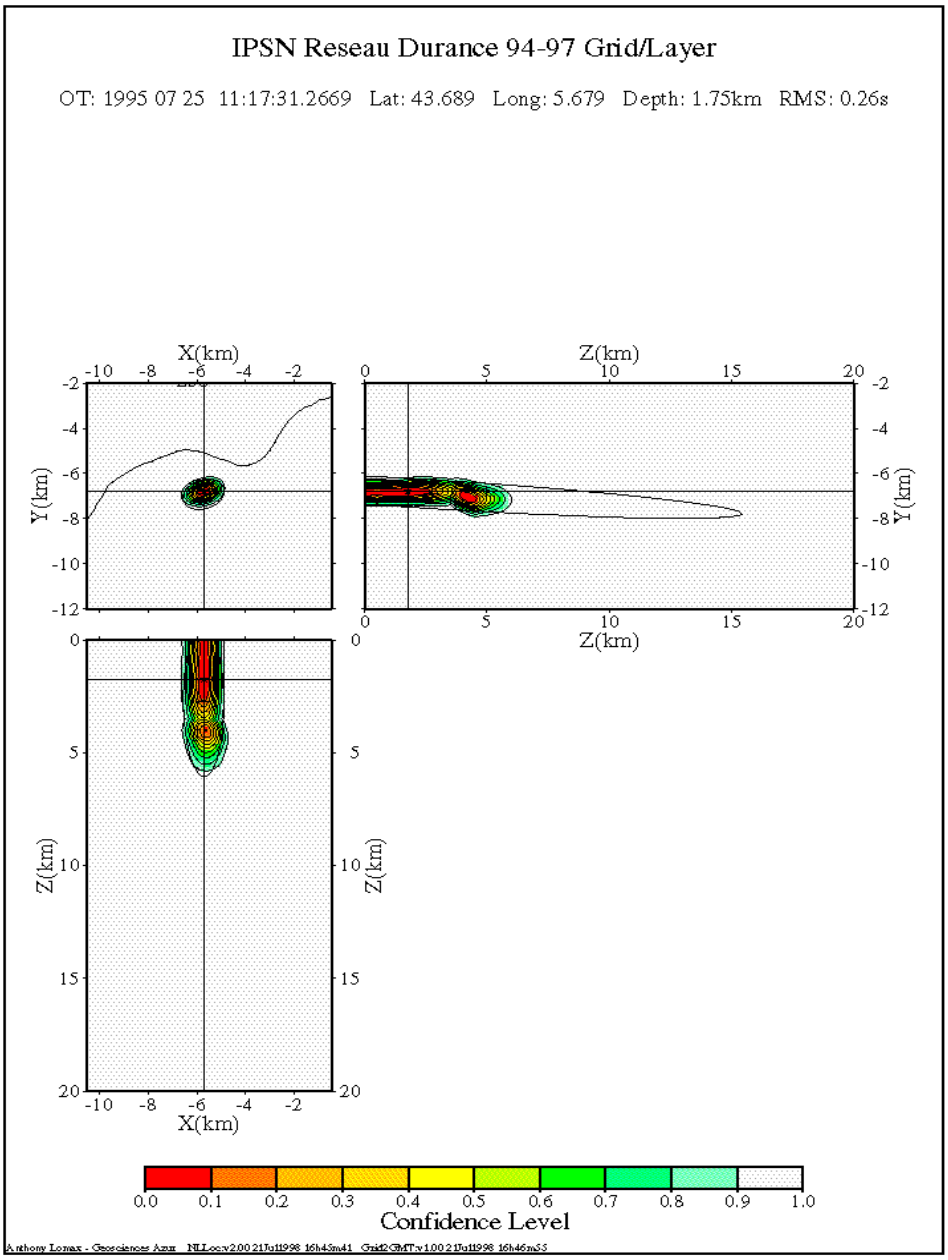
4. `Grid2GMT run/nlloc_sample.in loc/vinti gmt/ L S`

   Using Input Control File `run/nlloc_sample.in`, creates a GMT script for the summary hypocenter/scatter file with root name `loc/vinti` created by program [LocSum](). The GMT script and temporary files are written to directory `./gmt/`. The script will plot a Location (`L`, horizontal section and 2 vertical sections), showing the "scatter−cloud" samples (`S` for all events. (HYPERLINK "file:///zip/nlloc/soft2.30/Grid2GMT_ex4.gif"[View resulting plot]())



5. `Grid2GMT run/dur.in dur.19950725.111732.grid2.loc gmt/ L G`

   Using Input Control File `run/dur.in`, creates a GMT script for the 3D grid grid−search PDF file in the current directory with root name `dur.19950725.111732.grid2.loc`. The GMT script and temporary files are written to directory `./gmt/`. The script will plot a Location (`L`, horizontal section and 2 vertical sections), showing the location PDF as contoured, color confidence levels (`S`), along with location statistics (Maximum Likelihood location and traditional, Gaussian/Normal Expectation and projection of Confidence Ellipsoid).



Output

The Grid2GMT utility generates and runs an ASCII, GMT command script with the name *`Outroot`*`/`*`GridRoot`*`.gmt`. When run, this GMT command script creates a postscript file which can be viewed on screen with a postscript viewer, or printed on a postscript printer. Running the GMT command script requires the installation of the GMT plotting package.

The GMT command script can be edited and elements of the script extracted to produce custom GMT plots.

14. Utilitie Programs − Miscellaneous utility programs

Utilites for format conversion between HYPOELLIPSE summary or FPFIT summary formats and NonLinLoc hypocenter−Phase location format.

hypoe2hyp − fpfit2hyp

hypoe2hyp

The hypoe2hyp utility converts a file containing HYPOELLIPSE hypocenter summary records to a file in NonLinLoc hypocenter−Phase summary format. Optionally, the events written to output can be selected based on cut−off values for the maximum length of the confidence ellipsoid major axis, the maximum RMS misfit, the minimum number of observation readings, or the maximum azimuth gap.

Synopsis:

```
hypoe2hyp inputHypoellipseFile outputNonLinLocHypFile
[EllLenMax [RMSMax [NRdgsMin [GapMax]]]]
```

Parameters:

`inputHypoellipseFile` (*chars*)
: specifies the complete path and filename for an existing HYPOELLIPSE hypocenter summary file.

`outputNonLinLocHypFile` (*chars*)
: specifies the complete path, filename and file extension for the output NonLinLoc hypocenter−Phase summary file. The extension `.hyp` is recommended.

`EllLenMax` (*float*, default=1.0e6)
: maximum length of the confidence ellipsoid major axis for event selection.

`RMSMax` (*float*, default=1.0e6)
: maximum RMS misfit for event selection.

`NRdgsMin` (*int*, default=0)
: minimum number of observation readings for event selection.

`GapMax` (*int*, default=360)
: maximum RMS misfit for event selection.

Notes:

1. If a given event selection criteria is used, then values for all preceding selection criteria on the command line must also be included.
2. Only a subset of the NonLinLoc hypocenter−Phase file format lines are included in

the output.

fpfit2hyp

The fpfit2hyp utility converts a file containing FPFIT (Reasenberg *et al.*, 1985) focal mechanism summary records to a file in NonLinLoc hypocenter−Phase summary format. Optionally, the events written to output can be selected based on cut−off values for the the maximum RMS misfit for the mechanism determination, the maximum RMS misfit for the event location, the minimum number of observation readings, or the maximum azimuth gap.

Synopsis:

```
hypoe2hyp inputFPFITSumFile outputNonLinLocHypFile
[MechMisfitMax [RMSMax [NRdgsMin [GapMax]]]]
```

Parameters:

`inputFPFITSumFile` (*chars*)
: specifies the complete path and filename for an existing FPFIT summary file. (The extension for this file is usually `.sum`).

`outputNonLinLocHypFile` (*chars*)
: specifies the complete path, filename and file extension for the output NonLinLoc hypocenter−Phase summary file. The extension `.hyp` is reccommended.

`MechMisfitMax` (*float*, default=1.0e6)
: maximum RMS misfit of the mechanism determination for event selection.

`RMSMax` (*float*, default=1.0e6)
: maximum RMS misfit of the event location for event selection.

`NRdgsMin` (*int*, default=0)
: minimum number of observation readings for event selection.

`GapMax` (*int*, default=360)
: maximum RMS misfit for event selection.

Notes:

1. If a given event slection criteria is used, then values for all preceeding selection criteria on the comand line must also be included.
2. Only a subset of the NonLinLoc hypocenter−Phase file format lines are included in the output.

15. NonLinLoc programs file formats

Overview − Definitions − 3D Grid Files − Phase Files − Location Files

Overview

This chapter describes the input and output file formats used by the various NonLinLoc programs. Some formats (such as the 3D Grid format) are used by many of the programs, while others are specific to a particular program.

Many files have an easy to read ascii format with spaces between fields. To minimize disk space usage and to speed input/output some files are binary (such as the 3D Grid "buffer" file containing the grid data.

Definitions

General − ASCII Datatypes − Binary Datatypes

- **General:**

  FileExtension=
  : required extension for file name

  required
  : item must be present in file

  optional
  : item is optional in file

  repeatable
  : item may be present multiple times file

  ignored
  : may be present in file but is not used

- **ASCII Datatypes:**

  integer
  : ascii decimal integer (*i.e.* `0, 5, 285`)

  float
  : ascii decimal floating point number (*i.e.* `1.0, 34.68, -4.5`)

  expFloat

ascii exponential floating point number (*i.e.* `-5.44e+06`)

chars

sequence of ascii characters without spaces (*i.e.* `NO_SAVE, abcdef, /data/bigevent.dat`)

string

sequence of ascii characters continuing the end of line, spaces are allowed (*i.e.* `The biggest earthquake sequence in history`)

- **Binary Datatypes:**

  `char`

  1 byte character

  `int`

  signed integer (usually 4 bytes)

  `short`

  short integer (usually 2 bytes)

  `long`

  long integer (usually 4? bytes)

  `float`

  4 byte floating point number

  `double`

  8 byte floating point number

3D Grid Files Format

3D Grid Header file – 3D Grid Data buffer file

Each 3D Grid is stored on disk with a small, simple, ASCII header file and a (possibly very large) binary data buffer file. Both files have identical names except for the extension.

- **3D Grid Header file** (*ASCII, FileExtension=*`*.hdr`)

  Line 1: (*required*) Specifies the size and type of the 3D velocity grid.

  `xNum yNum zNum xOrig yOrig zOrig dx dy dz gridType`

  Fields:

  `xNum yNum zNum` (*integer*)

  number of grid nodes in the x, y and z directions

  `xOrig yOrig zOrig` (*float*)

  x, y and z location of the grid origin in km relative to the geographic origin.

  `dx dy dz` (*float*)

grid node spacing in kilometers along the x, y and z axes

`gridType` (*chars*)

specifies type of grid and physical quantity stored on grid:
`VELOCITY` = velocity (km/sec);
`VELOCITY_METERS` = velocity (m/sec);
`SLOWNESS` = slowness (sec/km);
`VEL2` = velocity**2 ((km/sec)**2);
`SLOW2` = slowness**2 ((sec/km)**2);
`SLOW2_METERS` = slowness**2 (sec/m)**2;
`SLOW_LEN` = slowness*length (sec);
`TIME` = time (sec) 3D grid;
`TIME2D` = time (sec) on 2D grid / 1D model;
`PROB_DENSITY` = probability density;
`MISFIT` = misfit (sec);
`ANGLE` = take−off angles 3D grid;
`ANGLE2D` = take−off angles on 2D grid / 1D model;

Notes:

1. The 3D velocity grid dimensions are in kilometers with Z positive down (left−handed co−ordinate system).
2. The length of the grid in the x direction is `(xNum−1)*dx`, and similar for y and z.
3. The location of the grid maximum in the x direction is `xOrig+(xNum−1)*dx`, and similar for y and z.

Example:

```
81 81 81 22.000000 42.500000 0.000000 0.200000
0.200000 0.500000 PROB_DENSITY
```

Line 2: (*required* for `gridType = TIME, TIME2D, ANGLE, ANGLE2D`, *ignored* otherwise) Specifies the label and x, y, z location of a source/station for the grid data.

```
label xSrce ySrce zSrce
```

Fields:

`label` (*chars*)

source/station label (*i.e.* a station code: `ABC`)

`xSrce ySrce` (*float*)

x and y positions relative to geographic origin in kilometers for source

`zSrce` (*float*)

z grid position (depth) in kilometers for source

Example:

```
CALN −8.882080 −27.537037 −1.430000
```

- **3D Grid Data buffer file** (*Binary*, *FileExtension*=`*.buf`)

Buffer: (*required*) Sequence of one `float` for each node on the grid specifying a physical value at the node (*i.e.* travel−time between the node and the source), or a coded set of values (*i.e.* take−off angles at the node for rays to the source)

```
d(N) (N = 0, xNum*yNum*zNum - 1)
```

Fields:

`d(N)` (`float`)
: value at node `N = ((Nx-1)*yNum + (Ny-1))*zNum + Nz`, where `xNum yNum zNum` are the number of grid nodes in the x, y and z directions and `Nx Ny Nz` are the node indexes (starting at 0) along the x, y and z axes, respectively.

Notes:

1. The values are stored as a succession of planes in x, each plane consisting of a succession of vectors in y, each vector consisting of a succession of points in z. Thus, to read/write the buffer requires a loop over z, contained within a loop over y, contained within a loop over x. If the file is read directly into a memory buffer, it might be accessed in C with an array `d[Nx][Ny][Nz]`, and in FORTRAN with an array `d(Nz,Ny,Nx)`.

2. For take−off angle data (`gridType = ANGLE, ANGLE2D`) each `float` value contains a coded set of 3 values − the ray take−off azimuth (16 bit integer, 0 to 3600 in tenths deg clockwise from North), the ray take−off dip (12 bit integer, 0 (down) to 1800 (up) in tenths deg) and a quality value (4 bit integer, 0 (low) to 10 (high)). The 3 values are related to the `float` with a C union:

```
typedef union
{
    float fval;                 /* float value (dummy) */
    unsigned short ival[2];     /* unsigned short values:
                                      ival[0] bits 0-3 = quality
                                              bits 4-15 = dip
                                      ival[1] = azimuth */
} TakeOffAngles;
```

The dummy variable `float fval` allows accessing, reading and writing of the buffer with the same functions that operate on the other data buffer types. An analogue to this union can be formed in FORTRAN using an `EQUIVALENCE` statement. The `unsigned short ival[2]` can be coded and decoded with the folowing C functions:

```
/*** function to set angle values in take-off angles union */
TakeOffAngles SetTakeOffAngles(double azim, double dip, int
iqual)
{
        TakeOffAngles angles;

        angles.ival[1] = (unsigned short) (0.5 + 10.0 * azim);
        angles.ival[0] = (unsigned short) iqual
                + (unsigned short) 16
                * (unsigned short) (0.5 + 10.0 * dip);

        return(angles);
```

```
}

/*** function to get values in take-off angles union */
int GetTakeOffAngles(TakeOffAngles *pangles,
                        double *pazim, double *pdip, int
*piqual)
{
        *pazim = ((double) pangles->ival[1]) / 10.0;
        *pdip = ((double) (pangles->ival[0] / (int) 16)) / 10.0;
        *piqual = (int) pangles->ival[0] % (int) 16;

        return(*piqual);
}
```

Phase File Formats

NonLinLoc Format − HYPO71 Format − NCSN Format − SIMULPS Format − SEISAN Format − RéNaSS DEP Format

The seismic phase time−pick observations for the program NLLoc may be specified in a number of different file formats. These files may contain picks for a single event or a number of events. NLLoc can read multiple observation files using a "wild−card" specification. (see the LOCFILES statement of the NLLoc control file).

Most phase file types may contain non phase pick data before or after the pick data for each event.

- **NonLinLoc Phase file format** (*ASCII*, NLLoc *obsFileType* = `NLLOC_OBS`)

  The NonLinLoc Phase file format is intended to give a comprehensive phase time−pick description that is easy to write and read.

  For each event to be located, this file contains one set of records. In each set there is one "arrival−time" record for each phase at each seismic station. The final record of each set is a blank. As many events as desired can be included in one file.

  Each record has a fixed format, with a blank space between fields. A field should never be left blank − use a "?" for unused characther fields and a zero or invalid numeric value for numeric fields.

  The NonLinLoc Phase file record is identical to the first part of each phase record in the NLLoc hypocenter−Phase file output by the program NLLoc. Thus the phase list output by NLLoc can be used without modification as time pick observations for other runs of NLLoc.

  NonLinLoc phase record:

  Fields:

  `Station name` (*char*6*)
  station name or code

`Instrument` (*char*4*)
: instument identification for the trace for which the time pick corresponds (*i.e.* `SP, BRB, VBB`)

`Component` (*char*4*)
: component identification for the trace for which the time pick corresponds (*i.e.* `Z, N, E, H`)

`P phase onset` (*char*1*)
: description of P phase arrival onset; `i`, `e`

`Phase descriptor` (*char*6*)
: Phase identification (*i.e.* `P, S, PmP`)

`First Motion` (*char*1*)
: first motion direction of P arrival; `c, C, u, U` = compression; `d, D` = dilatation; `+, -, Z, N`; `.` or `?` = not readable.

`Date` (*yyyymmdd*) (*int*6*)
: year (with century), month, day

`Hour/minute` (*hhmm*) (*int*4*)
: Hour, min

`Seconds` (*float*7.4*)
: seconds of phase arrival

`Err` (*char*3*)
: Error/uncertainty type; `GAU`

`ErrMag` (*expFloat*9.2*)
: Error/uncertainty magnitude in seconds

`Coda duration` (*expFloat*9.2*)
: coda duration reading

`Amplitude` (*expFloat*9.2*)
: Maxumim peak−to−peak amplitude

`Period` (*expFloat*9.2*)
: Period of amplitude reading

Example:

```
GRX    ?    ?    ? P      U 19940217 2216   44.9200 GAU  2.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
GRX    ?    ?    ? S      ? 19940217 2216   48.6900 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
CAD    ?    ?    ? P      D 19940217 2216   46.3500 GAU  2.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
CAD    ?    ?    ? S      ? 19940217 2216   50.4000 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
BMT    ?    ?    ? P      U 19940217 2216   47.3500 GAU  2.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
BMT    ?    ?    ? S      ? 19940217 2216   52.8700 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
ESC    ?    ?    ? P      D 19940217 2216   47.4700 GAU  2.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
ESC    ?    ?    ? S      ? 19940217 2216   52.8100 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
BST    ?    ?    ? P      D 19940217 2216   48.0000 GAU  1.00e+05
-1.00e+00 -1.00e+00 -1.00e+00
BST    ?    ?    ? S      ? 19940217 2216   54.6600 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00
```

- **HYPO71 Phase file format** (*ASCII*, NLLoc *obsFileType* = `HYPO71`)

The HYPO71 Phase file format allows direct reading by NLLoc of HYPO71 (Lee and Lahr,

1972) and of HYPOELLIPSE (Lahr, 1989) phase data files.

The following description of the **HYPO71 Phase file format** is modified from Lahr, 1989. Some HYPO71 fields are not used by NLLoc.

For each event to be located, this file contains one set of records. In each set there is one "arrival−time" record for each seismic station with the arrival times of P and S phases, the maximum amplitude and the period for amplitude determination, and a code (0 through 4) that indicates the precision of the arrival times. The final record of each set is a blank. As many events as desired can be included in one file.

HYPO71 phase record:

Fields:

`Station name` (Columns 1−4, *char*4*)
: station name or code

`P phase onset` (Columns 5, *char*1*)
: description of P phase arrival onset; `i`, `e`

`Phase descriptor` (Columns 6, *char*1*)
: P phase type `P`, `N` or `E`

`First Motion` (Columns 7, *char*1*)
: first motion direction of P arrival; `c, C, u, U` = compression; `d, D` = dilatation; `+, −, Z, N`; `.` or blank = not readable.

`P weight` (Columns 8, *int*1*)
: P phase weight code based on estimated standard deviation, `0, 1, 2, 3, 4`

`Date/time` (yymmddhhmm) (Columns 10−19, *int*10*)
: year (without century), month, day, hour, min

`P seconds` (Columns 20−24, *float*5.2*)
: seconds of P arrival

`S seconds` (Columns 32−36, *float*5.2*)
: seconds of S arrival

`S remark` (Columns 37−39, *char*3*)
: S phase type

`S weight` (Columns 40, *int*1*)
: S phase weight code based on estimated standard deviation, `0, 1, 2, 3, 4`

`Amplitude` (Columns 44−47, *float*4*)
: Maxumim peak−to−peak amplitude

`Period` (Columns 48−50, *float*3*)
: period of maxumim amplitude in 1/100 sec

`Time correction` (Columns 66−70, *float*5*)
: time correction sec (*not used by NLLoc*)

`F-P time` (Columns 71−75, *float*5*)
: F−P time interval in sec for FMAG calculation

Example:

```
ANNMiPc0 961217114029.07
BRUMeP 1 961217114035.97
CANM P 0 961217114029.57
```

```
STV iPd0 961217114027.40       45.64 S 1
ROB  P 0 961217114031.59       52.80 S 2
FIN  P 1 961217114032.23       53.65 S 2
```

- **Northern California Seismic Netowrk (NCSN) "Raw phase data (Y2K compliant)" file format** (*ASCII*, NLLoc *obsFileType* = `NCSN_Y2K_5`)

  The Northern California Seismic Netowrk (NCSN) "Raw phase data (Y2K compliant)" file format allows direct reading by NLLoc of Northern California Earthquake Catalog Search phase data files (available from the Northern California Earthquake Data Center (http://quake.geo.berkeley.edu/ncedc)).

  The documentation for the **NCSN "Raw phase data (Y2K compliant)" file format** is at ftp://quake.geo.berkeley.edu/pub/doc/cat5/ncsn.phase.y2k.5

  Note: NLLoc ignores all NCSN phases for which the P phase onset is "X", "Y", or "Z". This filter should remove all RTP picks.

  Example:

```
198910180009294237  910121 5738  196340  5240 19   334579 73214810
244 00LOM   59 -  1 240 719   8   13    0   5   0LOMT2 2D   9X
10090510X342  13
JLX  NC VVHZ EPD3198910180009 3047 -49   0     0    0    0      00 0   0
-8    0  62 6504   0     0334 00 00    0    02
JBC  NC VVHZ EPU3198910180009 3041 -68   0     0    0    0      00 0   0
2    0  67 8104   0     0283 00 00    0    02
JBL  NC EVLE EPU3198910180009 3274 -40   0 3604ES 3    0      00 0  52
-37 -66 191 6902   0    17262 00 00    0 9992
JSF  NC ZVLZ IPD1198910180009 3638   1258    0    0    0  15200 0   0
33    0 341 5634 25M    0326 00340 959    02 X
CAL  NC ZVLZ EPU3198910180009 3705    3 52    0    0    0   4700 0   0
66    0 359 5524 32M    0 22 00330 997    02 X
JMP  NC FVFZ EPD2198910180009 3694  -7103    0    0    0   1200 0   0
21    0 385 5424 14P    0332 00350 512    02
CAO  NC ZVLZ EPD3198910180009 3924 109   0     0    0    0   4900 0   0
55    0 433 5224 35M    0 59 00350    0    02 X
HMO  NC VVHZ EPD4198910180009 3871-176   0     0    0    0      00 0   0
-30    0 613 5504   0     0176 00 00    0    02
CMC  NC VVHZ EPU3198910180009 4281  -4 35     0    0    0      00 0   0
58    0 725 4904   0     0345 00 00 530    02

10090510
```

- **SIMULPS file format** (*ASCII*, NLLoc *obsFileType* = `SIMULPS`)

  The SIMULPS file format allows direct reading by NLLoc of SIMULPS "Traveltime Data for Earthquakes" format. See ftp://orfeus.knmi.nl/pub/software/mirror/thurber/SIMULPS12/simulps.ms.ps).

  Example:

```
84 3 6   6 7 58.58 46N19.78   7E26.85   3.77   2.00
SIE_IP-1  1.45DIX2IP-1  4.94DIX_IP-1  4.95EMS_IP-1  8.14EMV_IP-1
8.54MMK2IP-1  8.58
MMK_IP-1  8.67STG_IP-1 16.48SLE_IP-1 27.54WIL_IP-1 28.14SAX_IP-1
30.78
```

```
84 3 7   4 7 56.92 46N20.89   7E26.59   6.09   1.50
ZZB_IP-1  1.03ZZE_IP-1  1.06ZZA_IP-1  1.07ZZC_IP-1  1.11ZZD_IP-1
1.11ZZF_IP-1  1.11
```

- **SEISAN Phase file format** (*ASCII*, NLLoc *obsFileType* = `SEISAN`)

  The SEISAN Phase file format allows direct reading by NLLoc of SEISAN/HYPOCENTER (ftp://ftp.ifjf.uib.no/pub/seismo/SEISAN/) phase data files.

The description of the **SEISAN/HYPOCENTER Phase file format** below is available from the HYPOCENTER 3.2 reference manual (`hypocent.ps`, at ftp://ftp.ifjf.uib.no/pub/seismo/SEISAN/) (Lienert, 1994, and HYPOCENTER download at ftp://elepaio.soest.hawaii.edu/pub/lienert/). Some SEISAN/HYPOCENTER fields are not used by NLLoc.

For each event to be located, this file contains one set of records. In each set there is one "phase header line" with general date and hyocenter information followed by optional information lines followed by a "phase data line" for each arrival times of a single P, S or other phase. The final record of each set is a blank. As many events as desired can be included in one file.

Example:

```
 1997 1014 2347 54.5 L  43.069  12.817  8.0  ITA  5 0.1 0.4CITA
                1.23       7.2     6.5  9.5 -0.2402E+01 -0.7041E+01
-0.1050E+02E
 ACTION:UPD 98-05-07 09:49 OP:fran STATUS:
ID:19971014234754     I
 9710-14-2347-11S.ITA_018
6
 STAT SP IPHASW D HRMM SECON CODA AMPLIT PERI AZIMU VELO SNR AR TRES
W  DIS CAZ7
 ANNI    IP      D 2347 56.36
-0.110    3 110
 ANNI    IS   4    2347 57.80                                    -0.2
0    3 110
 COLF    IP      D 2347 56.97   18
0.110    8 135
 COLF    IS   1    2347 58.96                                     0.2
7    8 135
 FRAN    IP   1    2347 57.09                                     0.1
7    9 173
 FRAN    IS   1    2347 58.85                                    -0.1
7    9 173
 SVIT    IP      C 2347 57.18   22
0.110   10 216
 SVIT    IS   1    2347 59.15                                     0.0
7   10 216
 FEMA    IP      D 2347 59.06                                    -0.1
9   21 135
 FEMA    IS   1    2348  2.70                                     0.0
7   21 135
```

- **RéNaSS DEP Phase file format** (*ASCII*, NLLoc *obsFileType* = `RENASS_DEP`)

  The RéNaSS DEP Phase file format allows direct reading by NLLoc of the DEP ("depouillement") format for phase picks available on the internet from the Réseau National de Surveillance Sismique, Ecole et Observatoire des Sciences de la Terre de Strasbourg (http://renass.u−strasbg.fr/ren.fr.html).

Some RéNaSS DEP fields are not used by NLLoc.

This file contains one set of records for a single event. The file name must follow the RéNaSS DEP specification as this name is used by NLLoc to determine the date of the event.

description of the **RéNaSS DEP Phase file format** (http://renass.u−strasbg.fr/Reseaux_Sismiques/Nouveau_format/dathed.fr.html , see section III − Les fichier DEP ("depouillement"))

Event Location Files Format

NLLoc Hypocenter−Phase file − Phase Statistics file − Scatter file − Confidence Level file − Quasi−HYPOELLIPSE file − HYPO71 Hypocenter/Station file − HypoInverse Archive file

The location results from a run of the program NLLoc are stored in individual event files and in summary files for the run. A number of different file types are output, some of these are created and saved only if specifically requested in the NLLoc control file.

The individual event files have names of the form:

path−Name.date.time.`grid`*N*.`loc`.*FileExtension*

and the summary files have names of the form:

path−Name.`sum`.`grid`*N*.`loc`.*FileExtension*

where *path−Name* is the output file path and root name specified in the LOCFILES statement of the NLLoc control file, *date.time* is an automatically generated data and time for the event of the form *yyyymmdd.hhmmss*, *N* is the grid index starting from 0 for the initial search grid, and *FileExtension* is the required file extension for each output file type as specified below.

- **NLLoc Hypocenter−Phase file** (*ASCII*, *FileExtension*=`*.hyp`)

  The NLLoc Hypocenter−Phase file is an easy to read ASCII file output by the program NLLoc. This file contains a description of the search results and the optimal/maximum−likelihood hypocenter, it may also include phase information for this hypocenter. The summary NLLoc Hypocenter−Phase file contains hypocenter information for all events from a single run of the program NLLoc but does not contain phase information. The event NLLoc Hypocenter−Phase files contain hypocenter and phase information for a single events. The NLLoc Hypocenter−Phase file output by program LocSum also include a Scatter block (`SCATTER . . . END_SCATTER`) containing samples of the location PDF.

NLLOC Line: (*required*) Specifies the beginning of a NLLoc Hypocenter−Phase description block and gives the event file name. The end of the block is denoted by an `END_NLLOC` line (see below).

`NLLOC "`*eventFileName*`"`

Fields:

eventFileName (*string*)
: path and root name of individual event files corresponding to this hypocenter

SIGNATURE Line: (*required*) Signature text and program run stamp.

`SIGNATURE "`*signature verRunTime*`"`

Fields:

signature (*string*)
: is the signature text specified in the LOCSIG statement of the NLLoc control file

verRunTime (*string*)
: is an automatically generated text giving the NLLoc program version and the date/time of the run

COMMENT Line: (*required*) Comment text.

`COMMENT "`*commentText*`"`

Fields:

commentText (*string*)
: is the comment text specified in the LOCCOM statement of the NLLoc control file

GRID Line: (*required*) 3D Grid description.

`GRID` *gridDescription*

Fields:

gridDescription (*string*)
: is a standard grid description − see Line 1 of 3D Grid Header file above.

SEARCH Line: (*required*) Search type description.

`SEARCH GRID` *numSamplesDraw*
or
`SEARCH METROPOLIS nSamp` *numSamples* `nAcc` *numAccepted* `nSave` *numSaved* `nClip` *numClipped* `Dstep0` *stepInit* `Dstep` *stepFinal*

Fields: (`GRID`**)**

numSamplesDraw (*integer*)
specifies the number of samples drawn from a PDF grid

Fields: (`METROPOLIS`**)**

numSamples (*integer*)
total number of accepted samples obtained
numSaved (*integer*)
number of accepted samples saved
numClipped (*integer*)
number of samples rejected because they were outside of the search grid (a high value indicates that event maximum likelihood location may be outside search grid)
stepInit (*integer*)
initial step size in km used for the learning stage
stepFinal (*integer*)
step size in km used for the saving stage

HYPOCENTER Line: (*required*) Maximum likelihood hypocenter − xyz co−ordinates.

`HYPOCENTER x` *xLoc* `y` *yLoc* `z` *zLoc* `OT` *originSec* `ix` *iX* `iy` *iY* `iz` *iZ*

Fields:

xLoc yLoc (*float*)
x and y positions of hypocenter relative to geographic origin in kilometers
zLoc (*float*)
z position (depth) of hypocenter in kilometers
originSec (*float*)
seconds of origin time of hypocenter
iX iY iZ (*integer*)
x, y and z index of grid node location of hypocenter

GEOGRAPHIC Line: (*required*) Maximum likelihood hypocenter − Geographic co−ordinates.

`GEOGRAPHIC OT` *year month day hour minute second* `Lat` *latitude* `Long` *longitude* `Depth` *depth*

Fields:

year month day (*integer*)
date of origin time
hour minute (*float*)
hour and minute of origin time
second (*float*)
seconds of origin time
latitude longitude depth (*float*)
latitude and longitude in decimal degrees and depth in kilometers of

hypocenter

STAT_GEOG Line: (*optional*) Gaussian Expectation (Statistical) hypocenter − Geographic co−ordinates.

`STAT_GEOG ExpectLat` *latitude* `Long` *longitude* `Depth` *depth*

Fields:

latitude longitude depth (*float*)
: latitude and longitude in decimal degrees and depth in kilometers of expectation hypocenter

QUALITY Line: (*required*) Maximum likelihood hypocenter − Geographic co−ordinates.

`QUALITY Pmax` *probMax* `MFmin` *misfitMin* `MFmax` *misfitMax* `RMS` *rms* `Nphs` *nPhases* `Gap` *maxGap* `Dist` *minStaDist* `Mamp` *magAmp nMagAmp* `Mdur` *magDur nMagDur*

Fields:

probMax (*float*)
: maximum probability on grid

misfitMin misfitMax rms (*float*)
: maximum and minimum weighted misfit on grid

rms (*float*)
: root−mean−square of residuals at maximum likelihood hypocenter

magAmp (*float*)
: amplitude (i.e. ML) magnitude for maximum likelihood hypocenter

nMagAmp (*integer*)
: number of readings used for amplitude magnitude

magDur (*float*)
: duration magnitude for maximum likelihood hypocenter

nMagDur (*integer*)
: number of readings used for duration magnitude

VPVSRATIO Line: (*required*) Estimated Vp/Vs ratio. **(ver 2.0)**

`VPVSRATIO VpVsRatio` *ratio* `Npair` *numPair*

Fields:

ratio (*float*)
: Vp/Vs ratio estimated following the iterative minimization technique of Lahr (1989; ch. 5)

Npair (*integer*)
: number of P−S pairs used to calculate Vp/Vs ratio

STATISTICS Line: (*required*) "Traditional" Gaussian (normal) statistics of PDF (evaluated for `PROB_DENSITY` grids only).

`STATISTICS ExpectX` *expX* `Y` *expY* `Z` *expZ* `CovXX` *covXX* `XY` *covXY* `XZ` *covXZ* `YY covYY YZ` *covYZ* `ZZ` *covZZ* `EllAz1` *Azimuth1* `Dip1` *Dip1* `Len1` *StdErr1* `Az2` *Azimuth2* `Dip2` *Dip2* `Len2` *StdErr2* `Len3` *StdErr3*

Fields:

expX expY expZ (*expFloat*)
: expectation values of PDF for x, y and z

covXX covXY ... (*expFloat*)
: covariance values of PDF

Azimuth1 (*float*)
: azimuth of axis 1 of confidence ellipsoid

Dip1 (*float*)
: dip of axis 1 of confidence ellipsoid

StdErr1 (*expFloat*)
: length of semi−axis 1 of confidence ellipsoid

Azimuth2 (*float*)
: azimuth of axis 2 of confidence ellipsoid

Dip2 (*float*)
: dip of axis 2 of confidence ellipsoid

StdErr2 (*expFloat*)
: length of semi−axis 2 of confidence ellipsoid

StdErr3 (*expFloat*)
: length of semi−axis 3 of confidence ellipsoid

Notes:

1. See NLLoc Program Inversion Approach for more information on the calculation of the expectation, covariance and confidence ellipsoid "traditional" Gaussian statistics.
2. The confidence ellipsoid represents the 68% confidence region for 3 degrees of freedom (x, y and z).

TRANS Line: (*required*) Geographic to rectangular transformation parameters.

`TRANS SIMPLE` *latOrig longOrig rotAngle*
or `TRANS LAMBERT` *refEllipsoid latOrig longOrig firstStdParal secondStdParal*

Sets geographic to rectangular transformation parameters.

Fields: (`SIMPLE`)

latOrig (*float*)
: latitude in decimal degrees of the rectangular co−ordinates origin

longOrig (*float*)
: longitude in decimal degrees of the rectangular co−ordinates origin

rotAngle (*float*)
: rotation angle of the rectangular co−ordinates system Y−axis in degrees counter−clockwise (?) from geographic north (not implemented)

Fields: (`LAMBERT`**)**

refEllipsoid (*chars*)
: reference ellipsoid (one of: `WGS-84, GRS-80, WGS-72, Australian, Krasovsky, International, Hayford-1909, Clarke-1880, Clarke-1866, Airy, Bessel, Hayford-1830, Sphere`)

latOrig (*float*)
: latitude in decimal degrees of the rectangular co−ordinates origin

longOrig (*float*)
: longitude in decimal degrees of the rectangular co−ordinates origin

firstStdParal secondStdParal (*float*)
: first and second standard parallels (meridians) in decimal degrees

FOCALMECH Line: (*optional*) Focal mechanism description line. Specifies the hypocenter, focal mechanism, and mechanisn statistics

`FOCALMECH` *dlat dlong depth* `Mech` *dipDir dipAng rake* `mf` *misfit* `nObs` *nObs*

Fields:

dlat dlong (*floats*)
: latitude and logitude in decimal degrees of the hypocenter

depth (*float*)
: depth in kilometers of the hypocenter

dipDir dipAng rake (*floats*)
: dip strike and rake of double couple mechanism

mf (*float*)
: misfit of mechanism determination

nObs (*float*)
: number of first motion observations used in mechanism determination

PHASE Line: (*optional*) Phase format description line. Specifies the beginning of a block with a list of phase records. The end of the block is denoted by an `END_PHASE` line (see below)

```
PHASE ID Ins Cmp On Pha FM Date HrMn Sec Err ErrMag Coda
Amp Per > TTpred Res Weight StaLoc(X Y Z) SDist SAzim RAz
RDip RQual
```

Fields:

None

next *N* Lines: (*optional*) Phase data for *N* phases.

ID Ins Cmp On Pha FM Q Date HrMn Sec Coda Amp Per > Err ErrMag TTpred Res Weight StaLoc(X Y Z) SDist SAzim RAz RDip RQual

Fields:

ID (*char*6*)
station name or code

Ins (*char*4*)
instrument identification for the trace for which the time pick corresponds (*i.e.* `SP, BRB, VBB`)

Cmp (*char*4*)
component identification for the trace for which the time pick corresponds (*i.e.* `Z, N, E, H`)

On (*char*1*)
description of P phase arrival onset; `i`, `e`

Pha (*char*6*)
Phase identification (*i.e.* `P, S, PmP`)

FM (*char*1*)
first motion direction of P arrival; `c, C, u, U` = compression; `d, D` = dilatation; `+, -, Z, N`; `.` or `?` = not readable.

Date (*yyyymmdd*) (*int*6*)
year (with century), month, day

HrMn (*hhmm*) (*int*4*)
Hour, min

Sec (*float*7.4*)
seconds of phase arrival

Err (*char*3*)
Error/uncertainty type; `GAU`

ErrMag (*expFloat*9.2*)
Error/uncertainty magnitude in seconds

Coda (*expFloat*9.2*)
coda duration reading

Amp (*expFloat*9.2*)
Maxumim peak−to−peak amplitude

Per (*expFloat*9.2*)
Period of amplitude reading

> (*char*1*)
Required separator between first part (observations) and second part (calculated values) of phase record.

TTpred (*float*9.4*)
Predicted travel time

Res (*float*9.4*)
Residual (observed − predicted arrival time)

Weight (*float*9.4*)
Phase weight

StaLoc(X Y Z) (*3 * float*9.4*)
x, y, z location of station in transformed, rectangular co−ordinates

SDist (*float*9.4*)
Maximum likelihood hypocenter to station epicentral distance in kilometers

SAzim (*float*6.2*)
Maximum likelihood hypocenter to station epicentral azimuth in degrees CCW from North

RAz (*float*5.1*)
Ray take−off azimuth at maximum likelihood hypocenter in degrees CCW from North

RDip (*float*5.1*)
Ray take−off dip at maximum likelihood hypocenter in degrees upwards from vertical down (0 = down, 180 = up)

RQual (*float*5.1*)
Quality of take−off angle estimation (0 = unreliable, 10 = best)

END_PHASE Line: (*required* if Phase format description line is present) Specifies the end of a block with a list of phase records.

```
END_PHASE
```

Fields:

None

END_NLLOC Line: (*required*) Specifies the end of a NLLoc Hypocenter−Phase description block.

```
END_NLLOC
```

Fields:

None

SCATTER Line: (*optional*) Specifies the begining of a NLLoc Scatter block containing samples of the location PDF.

`SCATTER Nsamples` *nSamples*

Fields:

Nsamples (*integer*)
number of scatter samples

next *Nsamples* Lines: (*optional*) PDF samples.

x y z prob

Fields:

x y z (*float*)
x, y and z location of the sample in kilometers relative to the geographic origin.

prob (*float*)
relative probability density value at sample (for Grid Search, the probability density values are normalized so that the volume integral over the corresponding search grid of the PDF = 1.0)

END_SCATTER Line: (*required* if `SCATTER` line is present) Specifies the end of a NLLoc Scatter block.

```
END_SCATTER
```

Fields:

None

Example:

```
NLLOC "./loc/dur.19940217.221644.grid0" "LOCATED" "Location
completed."
SIGNATURE "Anthony Lomax - Geosciences Azur    NLLoc:v2.00 04Aug1998
16h40m19"
COMMENT "IPSN Reseau Durance 94-97 Grid/Layer"
GRID  51 51 43  -50.000000 -50.000000 -1.500000  2.000000 2.000000
1.000000 PROB_DENSITY
SEARCH METROPOLIS nSamp 13829 nAcc 10000 nSave 1000 nClip 0 Dstep0
1.774978 Dstep 0.789846
HYPOCENTER  x 24.705017 y -8.665524 z 1.592826  OT 40.998183  ix -1
iy -1 iz -1
GEOGRAPHIC  OT 1994 02 17  22 16 40.998183  Lat 43.671585 Long
6.056350 Depth 1.592826
QUALITY  Pmax 2.370e-02 MFmin 0.865120 MFmax 5.701586 RMS 0.157467
Nphs 10 Gap 329 Dist 18.346664 Mamp -9.90 0 Mdur -9.90 0
STATISTICS  ExpectX 2.6546e+01 Y -7.5794e+00 Z 5.1371e+00  CovXX
1.69e+00 XY 3.78e-01 XZ 9.87e-01 YY 2.08e+00 YZ 1.08e+00 ZZ 9.78e+00
EllAz1  296.7 Dip1  2.7 Len1  2.26e+00 Az2  206.2 Dip2  10.3 Len2
2.69e+00 Len3  5.96e+00
TRANSFORM  LAMBERT RefEllipsoid Clarke-1880  LatOrig 43.750000
LongOrig 5.750000  FirstStdParal 43.199300  SecondStdParal 44.996100
PHASE ID Ins Cmp On Pha  FM Date     HrMn   Sec      Err  ErrMag
Coda      Amp       Per       >   TTpred    Res       Weight
StaLoc(X  Y         Z)         SDist    SAzim  RAz   RDip  RQual
GRX    ?    ?    ? P      U 19940217 2216   44.9200 GAU  2.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >    4.1083   -0.1865    1.3136
9.1262    1.0246   -0.3350   18.3467 301.88 301.9  90.9  7
GRX    ?    ?    ? S      ? 19940217 2216   48.6900 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >    7.6074    0.0844    1.2757
9.1262    1.0246   -0.3350   18.3467 301.88 301.9  90.5  6
CAD    ?    ?    ? P      D 19940217 2216   46.3500 GAU  2.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >    5.1083    0.2435    1.3136
1.3977   -8.1653   -0.3700   23.3127 271.23 271.2  59.3  8
CAD    ?    ?    ? S      ? 19940217 2216   50.4000 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >    9.4494   -0.0476    1.2757
1.3977   -8.1653   -0.3700   23.3127 271.23 271.2  58.4  8
BMT    ?    ?    ? P      U 19940217 2216   47.3500 GAU  2.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >    6.3464    0.0054    0.8953
-5.0083   -2.0904   -0.4300   30.4321 282.48 282.5  50.9 10
BMT    ?    ?    ? S      ? 19940217 2216   52.8700 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >   11.7356    0.1362    0.8776
-5.0083   -2.0904   -0.4300   30.4321 282.48 282.5  50.7 10
ESC    ?    ?    ? P      D 19940217 2216   47.4700 GAU  2.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >    6.5539   -0.0821    0.8953
-6.2166   -1.8857   -0.4050   31.6561 282.37 282.4  50.9 10
ESC    ?    ?    ? S      ? 19940217 2216   52.8100 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >   12.1189   -0.3071    0.8776
-6.2166   -1.8857   -0.4050   31.6561 282.37 282.4  50.7 10
BST    ?    ?    ? P      D 19940217 2216   48.0000 GAU  1.00e+05
-1.00e+00 -1.00e+00 -1.00e+00 >    7.3481   -0.3463    0.0000
-8.4624    5.8008   -0.5000   36.1850 293.56 293.6  50.9 10
BST    ?    ?    ? S      ? 19940217 2216   54.6600 GAU  4.00e-02
-1.00e+00 -1.00e+00 -1.00e+00 >   13.5860    0.0758    1.2757
-8.4624    5.8008   -0.5000   36.1850 293.56 293.6  50.7 10
END_PHASE
END_NLLOC
```

- **Phase Statistics file** (*ASCII, FileExtension*=`*.stat`)

The Phase Statistics file contains the average residuals (denoted by `Average Phase Residuals`) for P and S phases at each station. The average residuals are calculated for all station/phase combinations that are present in the observations files and for which the corresponding time grids exist, regardless of the weight of the station/phase reading used for location.

In the Phase Statistics file there are 2 sets of phase statistics. One is based on the residuals for the maximum likelihood hypocenter. The other is based on a PDF−weighted average of the residuals for locations throughout the search region, where the weight is given by the PDF function likelihood values (Grid Search) or the density of samples (Metroplolis).

The Phase Statistics file also lists a cumulative delay (denoted by `Total Phase Corrections`) for P and S phases at each station. The cumulative delay is given by the sum of the average residuals and the input delay specified in LOCDELAY statements in the NLLoc Statements section of the Input Control File. The cumulative delay is listed for the maximum likelihood hypocenter residuals and for the PDF−weighted residuals.

The body of a Phase Statistics file can be used directly as a set of time delay LOCDELAY statements in the Input Control File. Thus the average phase residuals from a run of NLLoc can be used as the station corrections for later runs of NLLoc.

Line 1: (*required*) Title and maximum values for P and S average residual calculation.

> `Average Phase Residuals (CalcResidual) P_Residual_Max:` *P_residualMax* `S_Residual_Max:` *S_residualMax*
> or
> `Total Phase Corrections (CalcResidual + InputDelay)`
> `P_Residual_Max:` *P_residualMax* `S_Residual_Max:` *S_residualMax*
> or
> `Average Phase Residuals PDF (CalcPDFResidual)`
> `P_Residual_Max:` *P_residualMax* `S_Residual_Max:` *S_residualMax*
> or
> `Total Phase Corrections PDF (CalcPDFResidual + InputDelay)`
> `P_Residual_Max:` *P_residualMax* `S_Residual_Max:` *S_residualMax*

Fields:

P_residualMax S_residualMax (*float*)
: maximum (cut−off) value for P and S average residual calculation (set in Input Control File statement LOCPHSTAT). Residuals larger than the cut−off are not used in the calculation.

Line 2: (*required*) Column headings.

> `ID Phase Nres AveRes`

Lines 3−N: (*required*) Average residuals.

> `LOCDELAY` *label phaseType nResiduals residual*

Fields:

label (*chars*)
source/station label
phaseType (*chars*)
phase type `P` or `S`
nResiduals (*int*)
number of residuals used to calculate average residual
residual (*float*)
average residual

Example:

```
Average Phase Residuals (CalcResidual)  P_Residual_Max: 1.000000
S_Residual_Max: 1.000000
          ID         Phase     Nres    AveRes
LOCDELAY  BST      P       19       0.033835
LOCDELAY  BST      S       17       0.032845
LOCDELAY  CAD      P       22       0.155235
LOCDELAY  CAD      S       24       0.070664

Total Phase Corrections (CalcResidual + InputDelay)  P_Residual_Max:
1.000000  S_Residual_Max: 1.000000
          ID         Phase     Nres    TotCorr
LOCDELAY  BST      P       19       0.033835
LOCDELAY  BST      S       17       0.032845
LOCDELAY  CAD      P       22       0.155235
LOCDELAY  CAD      S       24       0.070664

Average PDF Phase Residuals (CalcPDFResidual)  P_Residual_Max:
1.000000  S_Residual_Max: 1.000000
          ID         Phase     Nres    AveRes
LOCDELAY  BST      P       19       0.051917
LOCDELAY  BST      S       17       0.026063
LOCDELAY  CAD      P       22       0.151105
LOCDELAY  CAD      S       24       0.058676

Total PDF Phase Corrections (CalcPDFResidual + InputDelay)
P_Residual_Max: 1.000000  S_Residual_Max: 1.000000
          ID         Phase     Nres    TotCorr
LOCDELAY  BST      P       19       0.051917
LOCDELAY  BST      S       17       0.026063
LOCDELAY  CAD      P       22       0.151105
LOCDELAY  CAD      S       24       0.058676
```

- **Scatter file** (*Binary*, *FileExtension*=`*.scat`)

The Scatter file contains the *x*,*y*,*z* locations and PDF value of each sample of the location PDF. The number of samples to save is specified in the LOCSEARCH statement in the NLLoc Statements section of the Input Control File.

Header: (*required*) one `float` value

nSamples

Fields:

nSamples (*float*)
number of PDF samples in the following buffer

Buffer: (*required*) Sequence of four `float` values for each PDF sample

x(N), y(N), z(N), pdf(N) (N = 0, nSamples − 1)

Fields:

x(N), y(N), z(N) (`float`)
: x, y and z location of the sample in kilometers relative to the geographic origin.

pdf(N) (`float`)
: PDF value of the sample, normalized so that the volume integral over the corresponding search grid of the PDF = 1.0

- **Confidence Level file** (*ASCII, FileExtension*=`*.conf`)

The Confidence Level file lists the values of the location PDF corresponding to confidence levels from 0.1 to 1.0 in increments of 0.1. Each PDF value denotes a constant−PDF surface in the 3D, *x,y,z*, PDF function; the integral of the PDF over the volume within this surface will evaluate to the corresponding confidence level. The algorithm to determine the PDF value is described in Moser, van Eck and Nolet (1992).

Line 1−11: (*required*) PDF value and confidence level.

PDFvalue `C` *confidenceLevel*

Fields:

PDFvalue (*float*)
: PDF value of the constant PDF surface

confidenceLevel (*float*)
: confidence level corresponding to the constant PDF surface defined by *PDFvalue*

Notes:

1. The character `C` is required in each line but is not currently used.

Example:

```
0.000000 C 1.00
0.011893 C 0.90
0.023787 C 0.80
0.040438 C 0.70
0.054710 C 0.60
0.073740 C 0.50
0.095148 C 0.40
0.118935 C 0.30
0.142722 C 0.20
0.183160 C 0.10
0.237870 C 0.00
```

- **Quasi−HYPOELLIPSE hypocenter file** (*ASCII, FileExtension*=`*.hypo_ell`)

The Quasi−HYPOELLIPSE hypocenter format is an expanded version of the hypocenter summary record output by HYPOELLIPSE (Lahr, 1989). In the original HYPOELLIPSE format there are no spaces between fields and all decimal points are removed; the Quasi−HYPOELLIPSE hypocenter format includes spaces and decimal points, but does not include all fields of the original HYPOELLIPSE format.

Line 1: (*required*) Field description.

```
DATE ORIGIN LAT LONG DEPTH MAG NO GAP D1 RMS AZ1 DIP1 SE1
AZ2 DIP2 SE2 SE3
```

Line 2−N+1: (*required*) Hypocenter summary record for N events.

Fields:

Origin date (Columns 1−8, *int*8*) (*yyyymmdd*)
origin year (including century), month, day
Origin hour/minute (Columns 10−13, *int*4*) (`hhmm`)
origin hour, minute
Origin seconds (Columns 15−19, *float*5.2*) (*dd.dd*)
origin seconds
LatitudeDeg (Columns 21−23, *int*3*) (`ddd`)
latitude degrees
LatitudeNS (Columns 25, *char*1*) (`A`)
latitude direction `N` or `S`
LatitudeMin (Columns 27−31, *float*5.2*) (*dd.dd*)
latitude decimal minutes
LongitudeDeg (Columns 33−36, *int*4*) (*dddd*)
longitude degrees
LongitudeEW (Columns 38, *char*1*) (`A`)
longitude direction `W` or `E`
LongitudeMin (Columns 40−44, *float*5.2*) (*dd.dd*)
longitude decimal minutes
Depth (Columns 46−52, *float*7.3*) (*ddd.ddd*)
depth in kilometers
Magnitude (Columns 54−57, *float*4.2*) (`d.dd`)
preferred magnitude
Number Obs (Columns 59−61, *int*3*) (*ddd*)
number of observations used in solution
Gap (Columns 63−65, *int*3*) (*ddd*)
largest azimuth separation in degrees between stations
Distance (Columns 67−72, *float*6.2*) (`ddd.dd`)
distance in kilometers to closest station used in solution
RMS (Columns 74−78, *float*5.2*) (*dd.dd*)
RMS of residuals of observations used in solution
Azimuth 1 (Columns 80−83, *int*4*) (*dddd*)
azimuth of axis 1 of confidence ellipsoid
Dip 1 (Columns 85−88, *int*4*) (*dddd*)
dip of axis 1 of confidence ellipsoid
StdErr 1 (Columns 90−95, *float*6.2*) (`ddd.dd`)
length of semi−axis 1 of confidence ellipsoid

Azimuth 2 (Columns 97−100, *int*4*) (`dddd`)
  azimuth of axis 2 of confidence ellipsoid
Dip 2 (Columns 102−105, *int*4*) (`dddd`)
  dip of axis 2 of confidence ellipsoid
StdErr 2 (Columns 107−112, *float*6.2*) (*ddd.dd*)
  length of semi−axis 2 of confidence ellipsoid
StdErr 3 (Columns 114−119, *float*6.2*) (*ddd.dd*)
  length of semi−axis 3 of confidence ellipsoid

Example:

```
DATE      ORIGIN     LAT          LONG          DEPTH   MAG  NO  GAP
D1      RMS    AZ1  DIP1 SE1    AZ2  DIP2 SE2     SE3
19961019 1136 22.48   44 N 19.60     7 E 15.63    4.500 0.00  16  78
1.24   0.13     0    0    0.00    0    0    0.00    0.00
```

- **HYPO71 Hypocenter/Station file** (*ASCII, FileExtension=*`*.hypo_71`)

The HYPO71 Hypocenter format reproduces "hypocenter output" and "station output" produced by the program HYPO71 (Lee and Lahr, 1972). This format includes all of the fields of the original HYPO71 output, but many of the fields are not used and have dummy values such as `0.0`.

Example:

```
  DATE    ORIGIN    LAT      LONG      DEPTH    MAG NO DM GAP M  RMS
ERH  ERZ Q SQD  ADJ IN NR  AVR  AAR NM AVXM SDXM NF AVFM SDFM I
 940217 1945 10.28 43 49.04    5 47.40   4.54   0.00 10  9 235 0 0.11
1.7  0.7 C A D 0.00  0  0-0.00 0.00  0  0.0  0.0  0  0.0  0.0 0

  STN  DIST AZM AIN PRMK HRMN P-SEC TPOBS TPCAL DLY/H1 P-RES P-WT AMX
PRX CALX K XMAG RMK FMP FMAG SRMK S-SEC TSOBS S-RES  S-WT    DT
 GRX    8.8 138 105  PD4 1945 12.43  2.15  2.19  0.00 -0.04  1.31 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 GRX    8.8 138 105  S?4 1945 14.28  4.00  4.05  0.00 -0.05  1.28 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 BST   11.8 262  88  PD4 1945 13.00  2.72  2.74  0.00 -0.02  0.00 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 BST   11.8 262  87  S?4 1945 15.20  4.92  5.06  0.00 -0.15  1.28 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 BMT   12.6 221  81  PD4 1945 13.34  3.06  2.87  0.00 0.19   0.90 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 BMT   12.6 221  81  S?4 1945 15.74  5.46  5.31  0.00 0.15   0.88 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 ESC   13.3 225  77  PD4 1945 13.34  3.06  2.98  0.00 0.08   0.90 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 ESC   13.3 225  77  S?4 1945 15.73  5.45  5.51  0.00 -0.07  0.88 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 CAD   15.7 187  72  PD4 1945 13.79  3.51  3.40  0.00 0.11   1.31 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
 CAD   15.7 187  72  S?4 1945 16.45  6.17  6.29  0.00 -0.12  1.28 0.0
0.0 0.00 0 0.00 000 00.0 0.00 ??4 00.00 00.00 00.00   0.0
```

Notes:

1. The `AZM` field contains the ray take−off azuimuth, not the source to station aziumth. These two values may differ if 3D velocity models are used.
2. The quality value in the `PRMK` field contains the integer quality value coressponding to the phase time uncertainty in seconds. This correspondence is obtained by

applying the inverse of the mapping specified by the LOCQUAL2ERR statement of the NLLoc control file.

- **HypoInverse Archive file** (*ASCII*, *FileExtension*=`*.hypo_inv`)

  The HypoInverse Archive file contains a hypocenter summary record followed by optional phase records. The results for multiple events may present in a single file.

  HYPOINVERSE format, described below in a document from Fred Klein:

```
----------------------------------------------------------------------------

                                               February 13, 1991

HYPOINVERSE SUMMARY CARD FORMAT
------- ------ ------
Cols.  Format    Data
_____  ______    ____

1-10   5I2       Year, month, day, hour and minute.
11-14  F4.2      Origin time seconds.
15-16  F2.0      Latitude (deg).
17     A1        S for south, blank otherwise.
18-21  F4.2      Latitude (min).
22-24  F3.0      Longitude (deg).
25     A1        E for east, blank otherwise.
26-29  F4.2      Longitude (min).
30-34  F5.2      Depth (km).
35-36  F2.1      Primary amplitude magnitude XMAG1.
37-39  I3        Number of P & S times with final weights greater than
0.1.
40-42  I3        Maximum azimuthal gap.
43-45  F3.0      Distance to nearest station (km).
46-49  F4.2      RMS travel time residual.
50-52  F3.0      Azimuth of smallest principal error (deg E of N).
53-54  F2.0      Dip of smallest principal error (deg).
55-58  F4.2      Magnitude of smallest principal error (km).
59-61  F3.0      Azimuth of intermediate principal error.
62-63  F2.0      Dip of intermediate principal error.
64-67  F4.2      Magnitude of intermediate principal error (km).
68-69  F2.1      Primary coda magnitude FMAG1.
70-72  A3        Event location remark. (See table 7 below).
73-76  F4.2      Magnitude of largest principal error (km).
77-78  2A1       Auxiliary remarks (See note below).
79-80  I2        Number of S times with weights greater than 0.1.
81-84  F4.2      Horizontal error (km).
85-88  F4.2      Vertical error (km).
89-90  I2        Number of P first motions.
91-93  F3.1      Total of amplitude magnitude weights.
94-96  F3.1      Total of duration magnitude weights.
97-99  F3.2      Median-absolute-difference of amplitude magnitudes.
100-02 F3.2      Median-absolute-difference of duration magnitudes.
103-05 A3        3-letter code of crust and delay model. (See table 8
below).
106    A1        Crust model type code (H or T).
107    A1        Most common P & S data source code. (See table 1 below).
108    A1        Most common FMAG data source code.
109    A1        Most common XMAG data source code.
110    A1        Primary coda magnitude type code (from FC1 command).
111-13 I3        Number of valid P & S readings (assigned weight > 0).
114    A1        Primary amplitude magnitude type code (from XC1 command).


               The following are  written  only  if secondary magnitudes
are
               present.  Secondary magnitudes may appear in either
```

```
position
               and use the label codes given by the FC2 and XC2 commands.
               The code L is for ML calculated by the USGS from Berkeley
amp-
               litudes, and B is for ML from the Berkeley catalog.

115    A1        Secondary magnitude label or type code.
116-18 F3.2      Secondary magnitude.
119-21 F3.1      Total of the secondary magnitude weights.
122    A1        Secondary magnitude label or type code.
123-25 F3.2      Secondary magnitude.
126-28 F3.1      Total of the secondary magnitude weights.
```

MAGNITUDES

The magnitudes and their label codes are:

```
E   Primary coda magnitude. FMAG of Eaton (1991). Uses all components.
X   Primary amplitude magnitude. Jerry Eaton's XMAG. Uses all components.
               Secondary magnitudes:
L   Local magnitude computed from UC Berkeley Wood Anderson amplitudes.
B   Local magnitude from UC Berkeley's catalog.
               Magnitudes no longer used in CALNET catalog:
T   Lapse time coda magnitude of Michaelson (1990) from high gain
verticals.
Z   Low gain (Z component) magnitude of Hirshorn and Lindh (1989?).
```

AUXILIARY EVENT REMARKS (Summary card cols. 77-78)

Assigned by analyst (col. 77):

```
Q  Quarry blast (or NTS explosion)
N  NTS blast
F  Felt
```

Assigned by HYPOINVERSE (col. 78):

```
*  Location had convergence problems such as maximum number of iterations
or
   failure to reach a minimum RMS.
-  Depth was poorly constrained and was held fixed at its current value.
```

Example:

```
9608011344195144 2727  7E2285 4000 0  6317 45  14  0 0   0  0 0   0 0XXX
0   0   0   0 0
SURF P?0 9608011344 2857 -19100 3524 S 0  18  0100   0   0 451  000  0
0  0228 0   0   0
JAUF P?0 9608011344 2929   2100 3581 S 0 -10  0100   0   0 501  000  0
0  0228 0   0   0
OG30 P?0 9608011344 3069  16100 3795 S 0  -7  0100   0   0 614  000  0
0  0228 0   0   0
9608020434148944 2619  7E1611  500 0  6212 15   6  0 0   0  0 0   0 0XXX
0   0   0   0 0
 PZZ P?0 9608020434 1800   4100 1999 S 0  -6  0100   0   0 153  000  0
0  0161 0   0   0
 STV P?0 9608020434 1906   7100 2179 S 0   2  0100   0   0 218  000  0
0  0161 0   0   0
 ENR P?0 9608020434 1974   2100 2293 S 0  -8  0100   0   0 263  000  0
0  0161 0   0   0
```

16. NonLinLoc Software Package Control File

Overview − Definitions − Generic Control Statements − Vel2Grid Program − HGrid2Time Program − Time2EQ Program − NLLoc Program

Overview

The various NonLinLoc programs all use the same control file syntax and share some "generic" control statements. The control statements for all the NonLinLoc programs for a project (a study with common "generic" control statements) may be combined into one file without conflicts. The basic control file statement syntax consists of a control *keyword* followed by one or more *parameters* on a single line (except when a newline is explicitely required). **`KEYWORD`** ***parameter1 parameter2 ...*** The keyword must begin in the first column and is always in upper case. Keywords and parameters must be separated by one or more spaces or tabs. A required newline in a parameter list is indicated by **`[newline]`**. Blank lines and lines with **#** in the first column are ignored. Use **#** in the first column for comments and to "comment out" a statement. View a sample control file for the NonLinLoc programs.

Definitions

Statement Priority

required − must be present in control file to run the coressponding program

optional − optional in control file

repeatable − may be present multiple times in control file

ignored − may be present but is not used by program under certain conditions

Datatypes

integer decimal integer (*i.e.* **`0, 5, 285`**)

float − decimal floating point number (*i.e.* **`1.0, 3.68, -4.5, 5.4e6`**)

chars − sequence of characters without spaces (*i.e.* **`NO_SAVE, abcdef, /data/bigevent.dat`**)

string − sequence of characters which is read until the end of line, spaces are allowed (*i.e.* **`The biggest earthquake sequence in history`**)

choice − selection from a fixed list of items (*i.e.* **`SAVE NO_SAVE`**)

Miscellaneous

default: − indicates default values for parameters when an *optional* control statement is not present in the control file

min: max: − indicates minimum and maximum allowed values for parameters

VERY_LARGE_DOUBLE − a large floating point value (typically 1.0e+30)

Generic Control Statements

The generic control file statements may be used by one or more of the programs in the NonLinLoc package. The statements **TRANS** and **MAPLINE** and their parameters must be the same for all programs runs for a given location project.

INCLUDE − CONTROL − TRANS − MAPLINE − MAPTRANS − MAPGRID

INCLUDE − Include

optional, *repeatable*

Syntax 1: **INCLUDE** ***includeFile***

Inserts text from another file at current positon in control file.

includeFile (*string*) path and name of file to include

Notes:
1. This statement is implemented only for NonLinLoc programs Vel2Grid, Grid2Time, Time2EQ and NLLoc.
2. The included text must contain only valid NonLinLoc control statements, blank lines or comment lines, but may not have another **INCLUDE** statement.

CONTROL − Control

required, *non−repeatable*

Syntax 1: **CONTROL** ***messageFlag randomNumberSeed***

Sets various general program control parameters.

messageFlag (*integer*, min:**0**, default:**1**) sets the verbosity level for messages printed to the terminal (**0** = error messages only, **1** = **0** + higher−level warning and progress messages, **2** = **1** + lower−level warning and progress messages + information messages, ...)

randomNumberSeed (*integer*) integer seed value for generating random number sequences (used by program NLLoc to generate Metropolis samples and by program Time2EQ to generate noisy time picks)

TRANS − Geographic Transformation

required, *non−repeatable*

Syntax 1: **TRANS** ***SIMPLE latOrig longOrig rotAngle***

Syntax 2: **TRANS** ***LAMBERT refEllipsoid latOrig longOrig firstStdParal secondStdParal rotAngle***

Sets geographic to rectangular transformation parameters.

latOrig (*float*, min:**-90.0**, max:**90.0**) latitude in decimal degrees of the rectangular co−ordinates origin

longOrig (*float*, min:**-180.0**, max:**180.0**) longitude in decimal degrees of the rectangular co−ordinates origin

rotAngle (*float*, min:**-360.0**, max:**360.0**) rotation angle of geographic north in degrees clockwise relative to the rectangular co−ordinates system Y−axis

refEllipsoid (*choice*: **WGS-84 GRS-80 WGS-72 Australian Krasovsky International Hayford-1909 Clarke-1880 Clarke-1866 Airy Bessel Hayford-1830 Sphere**) reference ellipsoid name

latOrig (*float*, min:**-90.0**, max:**90.0**) latitude in decimal degrees of the rectangular co−ordinates origin

longOrig (*float*, min:**-180.0**, max:**180.0**) longitude in decimal degrees of the rectangular co−ordinates origin

firstStdParal secondStdParal (*float*, min:**-90.0**, max:**90.0**) first and second standard parallels (meridians) in decimal degrees

rotAngle (*float*, min:**-360.0**, max:**360.0**) rotation angle of geographic north in degrees clockwise relative to the rectangular co−ordinates system Y−axis

Notes:
1. ***rotAngle*** = **0** gives North along the positive Y−axis, ***rotAngle*** = **-30** gives North along the axis 30 deg counterclockwise from the positive Y−axis of the rotated, rectangular system.
2. The **LAMBERT** transformation is adapted from the source code of the GMT plotting package .

MAPLINE − Geographic Maplines

optional, *repeatable*

Syntax 1: **MAPLINE** ***formatType name red green blue lineStyle***

Specifies a file and drawing parameters for geographic line data.

formatType (*choice*: **GMT_LATLON GMT_LONLAT XY_LONLAT GMT_LONLATDEPTH GMT_LONLATELEV_M GMT_GRD**) line file format or GMT grd file format

name (*string*) full path and file name

red green blue (*float*, min:**0.0**, max:**1.0**) red, green and blue intensities (0.0−1.0) (not implemented)

lineStyle (*choice*: **SOLID DASHED DOTTED DASHDOT**) line style (not implemented)

Notes:
1. All formats except **GMT_GRD** specify 2D or 3D line files. Use **GMT_GRD** to specify GMT grd files, these will be plotted as a greyscale background image.
2. A GMT grid (**GMT_GRD**) cannot be used with a rotated co−ordinate system.

MAPTRANS − Geographic Transformation for Grid2GMT plot output

optional, *non−repeatable*

Syntax 1: **MAPTRANS** ***SIMPLE latOrig longOrig rotAngle***

Syntax 2: **MAPTRANS** ***LAMBERT refEllipsoid latOrig longOrig firstStdParal secondStdParal rotAngle***

Sets geographic to rectangular transformation parameters.

latOrig (*float*, min:**-90.0**, max:**90.0**) latitude in decimal degrees of the rectangular co−ordinates origin

longOrig (*float*, min:**-180.0**, max:**180.0**) longitude in decimal degrees of the rectangular co−ordinates origin

rotAngle (*float*, min:**-360.0**, max:**360.0**) rotation angle of geographic north in degrees clockwise relative to the rectangular co−ordinates system Y−axis

refEllipsoid (*choice*: **WGS-84 GRS-80 WGS-72 Australian Krasovsky International Hayford-1909 Clarke-1880 Clarke-1866 Airy Bessel Hayford-1830 Sphere**) reference ellipsoid name

latOrig (*float*, min:**-90.0**, max:**90.0**) latitude in decimal degrees of the rectangular co−ordinates origin

longOrig (*float*, min:**-180.0**, max:**180.0**) longitude in decimal degrees of the rectangular co−ordinates origin

firstStdParal secondStdParal (*float*, min:**-90.0**, max:**90.0**) first and second standard parallels (meridians) in decimal degrees

rotAngle (*float*, min:**-360.0**, max:**360.0**) rotation angle of geographic north in degrees clockwise relative to the rectangular co−ordinates system Y−axis

Notes:
1. ***rotAngle*** = **0** gives North along the positive Y−axis, ***rotAngle*** = **-30** gives North along the axis 30 deg counterclockwise from the positive Y−axis of the rotated, rectangular system.
2. The **LAMBERT** transformation is adapted from the source code of the GMT plotting package .
3. **MAPTRANS** specifies the transformation for Grid2GMT output to GMT plotting. **MAPTRANS** superseeds any other **TRANS** statement in the control file.

MAPGRID − Grid Description for Grid2GMT plot output

optional, *non−repeatable*

Syntax 1: **MAPGRID** ***xNum yNum zNum xOrig yOrig zOrig dx dy dz gridType***

Specifies the size and type of the 3D velocity grid.

xNum yNum zNum (*integer*, min:**2**) number of grid nodes in the x, y and z directions

xOrig yOrig zOrig (*float*) x, y and z location of the grid origin in km relative to the geographic origin.

dx dy dz (*float*) grid node spacing in kilometers along the x, y and z axes

gridType (*choice*: **XXX**) grid type (ignored).

Notes:
1. The 3D grid dimensions are in kilometers with Z positive down (left−handed co−ordinate system).
2. The grid is *dx*(xNum−1)* km long in the x direction, and similarly for y and z.
3. **MAPGRID** specifies the plot region for GRid2GMT output to GMT plotting. **MAPGRID** superseeds any other **xxxGRID** statements in the control file.

Vel2Grid Program

VGOUT − VGTYPE − VGGRID − LAYER − 2DTO3DTRANS − VERTEX − EDGE − POLYGON2

VGOUT − Output File Root Name

required, *non−repeatable*

Syntax 1: **`VGOUT`** ***`fileRoot`***

Specifies the directory path and file *root* name (no extension) for the output velocity grid.

`fileRoot` (*string*) full or relative path and file *root* name (no extension) for output

Notes:
1. The 3D velocity grid ouput files have names of the form:

   fileRoot.waveType **`.mod`** . *FileExtension*

VGTYPE − Wave Type

required, *repeatable*

Syntax 1: **`VGTYPE`** ***`waveType`***

Specifies the physical wave type for a velocity grid.

`waveType` (*choice*: **`P S`**) wave type

VGGRID − Grid Description

required, *non−repeatable*

Syntax 1: **`VGGRID`** ***`xNum yNum zNum xOrig yOrig zOrig dx dy dz gridType`***

Specifies the size and type of the 3D velocity grid.

`xNum yNum zNum` (*integer*, min:**`2`**) number of grid nodes in the x, y and z directions

xOrig yOrig zOrig (*float*) x, y and z location of the grid origin in km relative to the geographic origin.

dx dy dz (*float*) grid node spacing in kilometers along the x, y and z axes

gridType (*choice*: **VELOCITY VELOCITY_METERS SLOWNESS VEL2 SLOW2 SLOW_2_METERS SLOW_LEN**) physical quantity to store on grid (**VELOCITY** = km/s, **VELOCITY_METERS** = m/s, **SLOWNESS** = s/km, **VEL2** = vel**2, **SLOW2** = (s/km)**2, **SLOW_2_METERS** = slow**2 ((s/m)**2), **SLOW_LEN** = slow*dx (sec)).

Notes:
1. The 3D grid dimensions are in kilometers with Z positive down (left−handed co−ordinate system).
2. The grid is *dx*(xNum−1)* km long in the x direction, and similarly for y and z.

LAYER − Velocity Model − Layer

optional, *repeatable*

Syntax 1: **LAYER *depth VpTop VpGrad VsTop VsGrad rhoTop rhoGrad***

Specifies a constant or gradient velocity layer.

depth (*float*) depth to top of layer (use negative values for layers above z=0)

VpTop VsTop rhoTop (*float*) P velocity, and S velocity in km/s and density in kg/m**3 at the top of the layer.

VpGrad VsGrad rhoGrad (*float*) Linear P velocity and S velocity gradients in km/s/km and density gradient in kg/m**3/km increasing directly downwards from the top of the layer.

Notes:
1. Multiple layers must be specified in order of increasing depth of top of layer.
2. The layer with the deepest top extends implicitly to infinite depth.

2DTO3DTRANS − Velocity Model − 2D model to 3D model transformation

optional, *non−repeatable*

Syntax 1: **2DTO3DTRANS *xOrig yOrig rotation***

xOrig yOrig (*float*) x and y co−ordinates in kilometers of the center of rotation in the 3D model.

rotation (*float*, min:**-360.0**, max:**360.0**) rotation angle in degreees

COUNTERCLOCKWISE.

Notes:
1. The 2D to 3D transformation is applied after the general geographic transformation specified by the Generic control statement **TRANS** .
2. With ***rotation*** =0 the 2D model section will be parallel to the *x* direction in the 3D model, and the 2D model will be extended along the *y* direction in the 3D model.

VERTEX − Velocity Model − Vertex

optional, *repeatable*

Syntax 1: **VERTEX** ***id_num zloc xloc yloc***

Specifies a vertex in 2D or 3D space.

id_num (*integer*) vertex identification number (must be unique)

zloc xloc yloc (*float*) z (positive DOWN), x and y location in kilometers of vertex (*yloc* ignored for 2D models)

Notes:
1. A single vertex may be used in the definitions of multiple edges (see EDGE).

EDGE − Velocity Model − Edge

optional, *repeatable*

Syntax 1: **EDGE** ***id_num vertex1 vertex2***

id_num (*integer*) edge identification number (must be unique)

POLYGON2 − Velocity Model − 2D polygon

optional, *repeatable*

Syntax 1: **POLYGON2** ***id_num n_edges depth Vp_top Vp_grad Vs_top Vs_grad p_top p_grad***

id_num (*integer*) edge identification number (must be unique)

n_edges (*integer*, min:**0**) the number of edges defining this polygon

`depth` (*float*) reference depth for velocity and density (use negative values for depths above z=0)

`VpTop VsTop rhoTop` (*float*) P velocity, and S velocity in km/s and density in kg/m**3 at the reference depth (***depth***).

`VpGrad VsGrad rhoGrad` (*float*) Linear P velocity and S velocity gradients in km/s/km and density gradient in kg/m**3/km increasing directly downwards from the reference depth (***depth***).

Notes:

1. A 2D polygon may share edges with other 2D polygons.
2. The reference depth (***depth***) may be above, within, or below the polygon.

Grid2Time Program

GTFILES − GTMODE − GTSRCE − GT_PLFD

GTFILES − Input and Output File Root Name

required, *non−repeatable*

Syntax 1: **`GTFILES`** ***`inputFileRoot`*** ***`outputFileRoot`*** ***`waveType`*** ***`iSwapBytesOnInput`***

Specifies the directory path and file *root* name (no extension), and the wave type identifier for the input velocity grid and output time grids.

`inputFileRoot` (*string*) full or relative path and file *root* name (no extension) for input velocity grid (generated by program Vel2Grid)

`outputFileRoot` (*string*) full or relative path and file *root* name (no extension) for output travel−time and take−off angle grids

`waveType` (*choice*: **`P S`**) wave type

`iSwapBytesOnInput` (*integer*, min:`0`, max:`1`, default:`0`) flag to indicate if hi and low bytes of input velocity grid file should be swapped

Notes:
1. The ***inputFileRoot*** and ***outputFileRoot*** are appended with **`.`** ***waveType***
2. The 3D time grid ouput files have names of the form:

    outputFileRoot.waveType **`.`** ***label*** . *gridType* . *FileExtension*

where *label* is a source label (*i.e.* a station code), *gridType* is **`time`** or **`angle`** , *FileExtension* is **`.buf`** or **`.hdr`**.

GTMODE − Program Modes

required, *non−repeatable*

Syntax 1: **`GTMODE`** ***`gridMode`*** ***`angleMode`***

Specifies several program run modes.

`gridMode` (*choice*: **`GRID3D GRID2D`**) grid type (***GRID3D*** for a 3D, Nx*Ny*Nz grid or

GRID2D for a 2D, 2*Ny*Nz grid)

angleMode (*choice*: **ANGLES_YES ANGLES_NO**) sets if take−off angles are calculated and an angles grid is output (**ANGLES_YES** for angles calulcation or **ANGLES_NO** for no angles calculation)

GTSRCE − Source Description

required, *repeatable*

Syntax 1: **GTSRCE** ***label XYZ xSrce ySrce zSrce elev***

Syntax 2: **GTSRCE** ***label LATLON latSrce longSrce zSrce elev***

Syntax 3: **GTSRCE** ***label LATLONDM latDegSrce latMinSrce latDir longDegSrce longMinSrce longDir zSrce elev***

Syntax 4: **GTSRCE** ***label LATLONDS latDegSrce latMinSrce latSecSrce latDir longDegSrce longMinSrce longSecSrce longDir zSrce elev***

Specifies a source location. One time grid and one angles grid (if requested) will be generated for each source. Four formats are supported: **XYZ** (rectangular grid co−ordinates), **LATLON** (decimal degrees for latitude/longitude), **LATLONDM** (degrees + decimal minutes for latitude/longitude) and **LATLONDS** (degrees + minutes + decimal seconds for latitude/longitude).

label (*string*) source label (*i.e.* a station code: **ABC**)

xSrce ySrce (*float*) x and y grid positions relative to geographic origin in kilometers for source

zSrce (*float*) z grid position (depth, positive DOWN) in kilometers for source

elev (*float*) elevation above z grid position (positive UP) in kilometers for source

latSrce (*float*, min:**-90.0**, max:**90.0**) latitude in decimal degrees for source (pos = North)

longSrce (*float*, min:**-180.0**, max:**180.0**) longitude in decimal degrees for source (pos = East)

latDegSrce latMinSrce latSecSrce (*float*) latitude degrees, minutes and seconds for source

longDegSrce longMinSrce longSecSrce (*float*) longitude degrees, minutes and seconds for source

latDir (*choice*: **N S**) geographic direction

longDir (*choice*: **W E**) geographic direction

GT_PLFD − Podvin and Lecomte Finite Difference

required, *non−repeatable*, for Podvin and Lecomte finite difference, must not be present otherwise

Syntax 1: **`GT_PLFD`** ***`hs_eps_init message_flag`***

Selects Podvin and Lecomte finite difference method and specifies method parameters.

`hs_eps_init` (*float*, min:**`0.0`**) fraction (typically 1.0E−3) defining the tolerated model inhomogeneity for exact initialization. A tolerance larger than 0.01 will potentially create errors larger than those involved by the F.D. scheme without any exact initialization.

`message_flag` (*integer*, min:**`0`**, max:**`2`**) Message flag (0:silent, 1:few messages, 2:verbose) A negative value inhibits "clever" initialization.

Notes:

1. See Podvin and Lecomte finite difference source code and Podvin and Lecomte, 1991 for more information.

Time2EQ Program

EQFILES − EQEVENT − EQSTA − EQSRCE − EQMECH − EQMODE − EQQUAL2ERR − EQVPVS

EQFILES − Input and Output File Root Name

required, *non−repeatable*

Syntax 1: **`EQFILES`** ***`inputFileRoot outputFileName`***

Specifies the directory path and file *root* name (no extension) for the input time grids, and the path and filename for the output phase/observation file.

`inputFileRoot` (*string*) full or relative path and file *root* name (no extension) for input time grids (generated by program Grid2Time)

`outputFileName` (*string*) full or relative path and name for output phase/observation file

Notes:
1. The ***`inputFileRoot`*** should not include the standardized phase code (*i.e.* **`P`** or **`S`**).

EQEVENT − Hypocenter parameters

optional, *repeatable*

Syntax 1: **`EQEVENT`** ***`label xEvent yEvent zEvent originSeconds`***

`label` (*string*) event identification label

`xEvent yEvent zEvent` (*float*) x, y and z grid co−ordinates of hypocenter

`originSeconds` (*float*) origin time in seconds

Notes:
1. The the origin time ***`originSeconds`*** is added to the travel−time read from the time grid to get the synthetic phase time.

EQSTA − Station List

required, *repeatable*

Syntax 1: **`EQSTA`** ***`label phase errorType error errorReportType errorReport`***

Specifies a station, phase and timing error to use to generate a synthetic phase reading.

`label` (*string*) station label (*i.e.* a station code: **`ABC`**)

`phase` (*string*) phase type (*i.e.* **`P`** or **`S`**)

`errorType` (*choice*: **`GAU BOX FIX NONE`**) calculated random timing error type (**`GAU`** for normal deviate with zero mean and variance = ***error*** , or **`BOX`** for boxcar deviate with zero mean and width = 2 * ***error*** , or **`FIX`** for time error/static = ***error*** , or **`NONE`** for time error/static = ***0.0***)

`error` (*float*) error magnitude in seconds

`errorReportType` (*choice*: **`GAU`**) timing error type to write to output phase/observation file *Err* field (The current version of NLLoc recognizes only **`GAU`**)

`errorReport` (*float*) error magnitude in seconds to write to output phase/observation file *ErrMag* field.

Notes:
1. The ***`label`*** and ***`phase`*** when concatenated to the ***`inputFileRoot`*** (i.e. ***`inputFileRoot.label.phase`***) should correspond to the path and root name of an existing, travel−time grid file.
2. The error is calculated stochastically and added to the travel−time. Use ***error*** **`= 0.0`** to obtain exact synthetic travel−times.

EQSRCE − Source Description

optional, *repeatable*

Syntax 1: **`EQSRCE`** ***`label XYZ xSrce ySrce zSrce elev`***

Syntax 2: **`EQSRCE`** ***`label LATLON latSrce longSrce zSrce elev`***

Syntax 3: **`EQSRCE`** ***`label LATLONDM latDegSrce latMinSrce latDir longDegSrce longMinSrce longDir zSrce elev`***

Syntax 4: **`EQSRCE`** ***`label LATLONDS latDegSrce latMinSrce latSecSrce latDir longDegSrce longMinSrce longSecSrce longDir zSrce elev`***

Specifies a source location. Four formats are supported: **`XYZ`** (rectangular grid co−ordinates), **`LATLON`** (decimal degrees for latitude/longitude), **`LATLONDM`** (degrees + decimal minutes for latitude/longitude) and **LATLONDS** (degrees + minutes + decimal seconds for latitude/longitude).

label (*string*) source label (*i.e.* a station code: **ABC**)

xSrce ySrce (*float*) x and y grid positions relative to geographic origin in kilometers for source

zSrce (*float*) z grid position (depth, positive DOWN) in kilometers for source

elev (*float*) elevation above z grid position (positive UP) in kilometers for source

latSrce (*float*) latitude in decimal degrees for source (pos = North)

longSrce (*float*) longitude in decimal degrees for source (pos = East)

latDegSrce latMinSrce latSecSrce (*float*) latitude degrees, minutes and seconds for source

longDegSrce longMinSrce longSecSrce (*float*) longitude degrees, minutes and seconds for source

latDir (*choice*: **N S**) geographic direction

longDir (*choice*: **W E**) geographic direction

EQMECH − Event mechanism description

optional, *non−repeatable*

Syntax 1: **EQMECH** ***mechType strike dip rake***

Specifies the mechanism parameters for synthetic first motion calculations.

mechType (*choice*: **DOUBLE ISO NONE**, default:**NONE**) source mechanism type (**DOUBLE** for double couple, or **ISO** for isotropic/explosion, or **NONE** for no first motion calculation)

strike (*float*, min:**0.0**, max:**360.0**) strike of fault plane in degrees (0,360) clockwise from North in the Geographic reference frame (any ***rotAngle*** specified in the generic control statement **GTSRCE** will be added to ***strike***).

dip (*float*, min:**0.0**, max:**90.0**) dip of the fault plane in degrees (0,90) down from the horizontal.

rake (*float*, min:**−180.0**, max:**180.0**) angle in degrees (−180,180) on the fault plane between the strike direction and the slip direction.

Notes:
1. The the origin time ***originSeconds*** is added to the travel−time read from the time grid to

get the synthetic phase time.

EQMODE − Select Mode: sta−>source or source−>station

optional, *non−repeatable*

Syntax 1: **`EQMODE`** ***`mode`***

Selects calculation of times from single source to multiple stations, or from multiple sources to single station. The phase labels in the output phase/observation file are set to the station labels or to the source labels, depending on the mode.

`mode` (*choice*: **`SRCE_TO_STA`** **`STA_TO_SRCE`**, default:**`SRCE_TO_STA`**) **`SRCE_TO_STA`** for single sources to multiple stations or **`STA_TO_SRCE`** for single station to multiple sources.

EQQUAL2ERR − Quality to Error Mapping

required, *non−repeatable*

Syntax 1: **`EQQUAL2ERR`** ***`Err0 ... ... ... ...`***

Specifies the mapping of error to phase pick quality for output of phase/observations in HYPO71 file format (which does not include time uncertainties) (*i.e.* time uncertainties in seconds (*i.e.* **`0.01`** or **`0.5`**) to quality **`0,1,2,3`** or **`4`**).

`Err0 ... ErrN` (*float*, min:`0.0`) one time uncertainty value for each quality level that may be output to the phase/observation file. Synthetic errors less than or equal to the first value *Err0* are output with quality **`0`** , less than or equal to the second are output with **`1`** , etc.

EQVPVS − P Velocity to S Velocity Ratio

optional, *non−repeatable* (**ver 2.0**)

Syntax 1: **`EQVPVS`** ***`VpVsRatio`***

Specifies the P velocity to S velocity ratio to calculate S phase travel times.

`VpVsRatio` (*float*) P velocity to S velocity ratio. If ***`VpVsRatio`*** 0.0 then only P phase travel times grids are read and ***`VpVsRatio`*** is used to calculate S phase travel times. If ***`VpVsRatio`*** 0.0 then S phase travel times grids are used.

NLLoc Program

LOCSIG − LOCCOM − LOCFILES − LOCHYPOUT − LOCSEARCH − LOCMETH − LOCGAU − LOCPHASEID − LOCQUAL2ERR − LOCGRID − LOCPHSTAT − LOCANGLES − LOCMAG − LOCCMP − LOCALIAS − LOCEXCLUDE − HLOCDELAY

LOCSIG − Signature text

optional, *non−repeatable*

Syntax 1: **`LOCSIG`** ***`signature`***

Identification of an individual, institution or other entity − written in some output files.

`signature` (*line*) signature text

LOCCOM − Comment text

optional, *non−repeatable*

Syntax 1: **`LOCCOM`** ***`comment`***

Comment about location run − written in some output files.

`comment` (*line*) comment text

LOCFILES − Input and Output File Root Name

required, *non−repeatable*

Syntax 1: **`LOCFILES`** ***`obsFiles obsFileType inputFileRoot outputFileRoot`***

Specifies the directory path and filename for the phase/observation files, and the file *root* names (no extension) for the input time grids and the output files.

`obsFiles` (*string*) full or relative path and name for phase/observations files, mulitple files may be specified with standard UNIX "wild−card" characters (**`*`** and **`?`**)

`obsFileType` (*choice*: **`NLLOC_OBS HYPO71 HYPOELLIPSE RENASS_WWW RENASS_DEP NCSN_Y2K_5 SIMULPS`**) format type for phase/observations files (see Phase File

Formats)

inputFileRoot (*string*) full or relative path and file *root* name (no extension) for input time grids (generated by program Grid2Time)

outputFileRoot (*string*) full or relative path and file *root* name (no extension) for output files

LOCHYPOUT − Output File Types

optional, *non−repeatable*

Syntax 1: **LOCHYPOUT *fileType1 ... ...***

Specifies the filetypes to be used for output.

fileType1 ... fileTypeN (*choice*: **SAVE_NLLOC_ALL SAVE_NLLOC_SUM SAVE_HYPOELL_ALL SAVE_HYPOELL_SUM SAVE_HYPO71_ALL SAVE_HYPO71_SUM SAVE_HYPOINV_SUM**, default:**SAVE_NLLOC_ALL SAVE_HYPOINV_SUM**) File format types to be output: **SAVE_NLLOC_ALL** = save summary and event files of type NLLoc Hypocenter−Phase file , Phase Statistics file , Scatter file and Confidence Level file **SAVE_NLLOC_SUM** = save summary file only of type NLLoc Hypocenter−Phase file , Phase Statistics file , Scatter file and Confidence Level file **SAVE_HYPOELL_ALL** = save summary and event files of type Quasi−HYPOELLIPSE file **SAVE_HYPOELL_SUM** = save summary file only of type Quasi−HYPOELLIPSE file **SAVE_HYPO71_ALL** = save summary and event files of type HYPO71 Hypocenter/Station file **SAVE_HYPO71_SUM** = save summary file only of type HYPO71 Hypocenter/Station file **SAVE_HYPOINV_SUM** = save summary file only of type HypoInverse Archive file)

Notes:
1. The HypoInverse Archive format serves as input to the program FPFIT (Reasenberg *et al.* , 1985) for grid−search determination of focal mechanism solutions.

LOCSEARCH − Search Type

required, *non−repeatable*

Syntax 1: **LOCSEARCH *GRID numSamplesDraw***

Syntax 2: **LOCSEARCH *MET numSamples numLearn numEquil numBeginSave numSkip stepInit stepMin stepFact probMin***

Syntax 3: **LOCSEARCH *OCT initNumCells_x initNumCells_y initNumCells_z minNodeSize maxNumNodes numScatter***

Specifies the search type and search parameters. The possible search types are **GRID** (grid search), **MET** (Metropolis), and **OCT** (Octtree).

numSamplesDraw (*integer*) specifies the number of scatter samples to draw from each saved PDF grid (*i.e.* grid with ***gridType*** = **PROB_DENSITY** and ***saveFlag*** = **SAVE**) No samples are drawn if ***saveFlag*** 0.

numSamples (*integer*, min:**0**) total number of accepted samples to obtain

numLearn (*integer*, min:**0**) number of accepted samples for learning stage of search

numEquil (*integer*, min:**0**) number of accepted samples for equilibration stage of search

numBeginSave (*integer*, min:**0**) number of accepted samples after which to begin saving stage of search, denotes end of equilibration stage

numSkip (*integer*, min:**1**) number of accepted samples to skip between saves (***numSkip*** = **1** saves every accepted sample)

stepInit (*float*) initial step size in km for the learning stage (***stepInit*** **0.0** gives automatic step size selection. If the search takes too long, the initial step size may be too large; this may be the case if the search region is very large relative to the volume of the high confidence region for the locations.)

stepMin (*float*, min:**0.0**) minimum step size allowed during any search stage (This parameter should not be critical, set it to a low value.)

stepFact (*float*, min:**0.0**) step factor for scaling step size during equilibration stage (Try a value of 8.0 to start.)

probMin (*float*) minimum value of the maximum probability (likelihood) that must be found by the end of learning stage, if this value is not reached the search is aborted (This parameters allows the filtering of locations outside of the search grid and locations with large residuals.)

initNumCells_x initNumCells_y initNumCells_z (*integer*) initial number of octtree cells in the x, y, and z directions

minNodeSize (*float*) smallest octtree node side length to process (not used)

maxNumNodes (*integer*) total number of nodes to process

numScatter (*integer*) the number of scatter samples to draw from the octtree results

Notes:

1. See NLLoc Program Grid−Search Algorithm and Metropolis Sampling Algorithm for more information.
2. Samples are saved to a binary, event Scatter file (see Scatter file formats). For the grid−search, because the samples are drawn stochastically, the number of samples actually obtained my differ slightly from the requested number.
3. If a large number of samples are saved, the spatial density of samples will be proportional to the PDF.

4. The scatter samples are useful for plotting the PDF as a transparent "cloud" and for relatively compact disk storage of the PDF.

LOCMETH − Location Method

required, *non−repeatable*

Syntax 1: **`LOCMETH`** ***`method maxDistStaGrid minNumberPhases maxNumberPhases minNumberSphases VpVsRatio maxNum3DGridMemory`***

Specifies the location method (algorithm) and method parameters.

`method` (*choice*: **`GAU_ANALYTIC`**) location method/algorithm (**`GAU_ANALYTIC`** = the inversion approach of Tarantola and Valette (1982))

`maxDistStaGrid` (*float*) maximum distance in km between a station and the center of the initial search grid; phases from stations beyond this distance will not be used for event location

`minNumberPhases` (*integer*) minimum number of phases that must be accepted before event will be located

`maxNumberPhases` (*integer*) maximum number of accepted phases that will be used for event location; only the first ***maxNumberPhases*** read from the phase/observations file are used for location

`minNumberSphases` (*integer*) minimum number of S phases that must be accepted before event will be located

`VpVsRatio` (*float*) P velocity to S velocity ratio. If ***VpVsRatio*** 0.0 then only P phase travel times grids are read and ***VpVsRatio*** is used to calculate S phase travel times. If ***VpVsRatio*** 0.0 then S phase travel times grids are used.

`maxNum3DGridMemory` (*integer*) maximum number of 3D travel time grids to attempt to read into memory for Metropolis−Gibbs search. This helps to avoid time−consuming memory swapping that occurs if the total size of grids read exceeds the real memory of the computer. 3D grids not in memory are read directly from disk. If ***maxNum3DGridMemory*** 0 then NLLoc attempts to read all grids into memory.

Notes:
1. See NLLoc Program Inversion Approach for more information on the **`GAU_ANALYTIC`** method.
2. Phases that are not used for location are written to output files and are used for calculating average residuals.

LOCGAU − Gaussian Model Errors

required, *non−repeatable*

Syntax 1: **`LOCGAU`** ***`SigmaTime CorrLen`***

Specifies parameters for Gaussian modelisation−error covariances **`Covariance ij`** between stations **`i`** and **`j`** using the relation (Tarantola and Valette, 1982):

$$\texttt{Covariance ij} = SigmaTime^2 \exp(-0.5(\texttt{Dist}^2\ \texttt{ij})/CorrLen^2)$$

where **`Dist`** is the distance in km between stations **`i`** and **`j`** .

`SigmaTime` (*float*, min:**`0.0`**) typical error in seconds for travel−time to one station due to model errors

`CorrLen` (*float*, min:**`0.0`**) correllaton length that controls covariance between stations (*i.e.* may be related to a characteristic scale length of the medium if variations on this scale are not included in the velocity model)

LOCPHASEID − Phase Identifier Mapping

optional, *repeatable*

Syntax 1: **`LOCPHASEID`** ***`stdPhase phaseCode1 ... ... ... ... ...`***

Specifies the mapping of phase codes in the phase/observation file (*i.e.* **`pg`** or **`Sn`**) to standardized phase codes (*i.e.* **`P`** or **`S`**).

`stdPhase` (*string*) standardized phase code (used to generate time−grid file names)

`phaseCode1 ... phaseCodeN` (*string*) one or more phase codes that may be present in a phase/observation file that should be mapped to the ***stdPhase*** .

Notes:

1. In the current version of NLLoc, it is assumed for some processing (such as the calculation of average P and S station residuals) that the standardized phase codes are **`P`** and **`S`** . Thus it is important to use these codes, if possible.
2. A phase/observation file code will be used unchanged if no **`LOCPHASEID`** statement is specified, or the code is not present in any **`LOCPHASEID`** statement.

LOCQUAL2ERR − Quality to Error Mapping

required, *non−repeatable*, for phase/observation file formats that do not include time uncertainties ;

ignored, *non−repeatable*, otherwise

Syntax 1: `LOCQUAL2ERR` ***`Err0 ... ... ... ...`***

Specifies the mapping of phase pick qualities phase/observation file (*i.e.* `0,1,2,3` or `4`) to time uncertainties in seconds (*i.e.* `0.01` or `0.5`).

`Err0 ... ErrN` (*float*, min:`0.0`) one time uncertainty value for each quality level that may be used in a phase/observation file. The first value *Err0* is assigned to picks with quality `0` , the second to picks with quality `1` , etc.

Notes:
1. NLLoc requires Gaussian timing error estimates in seconds for the data (phase picks), the `LOCQUAL2ERR` statement allows a conversion of commonly used integer quality codes to *float* time values.
2. Use a large, positive value (*i.e.* `99999.9`) to indicate a phase pick that should have zero weight (infinite uncertainty).

LOCGRID − Search Grid Description

required, *repeatable*

Syntax 1: `LOCGRID` ***`xNum yNum zNum xOrig yOrig zOrig dx dy dz gridType saveFlag`***

Specifies the size and other parameters of an initial or nested 3D search grid. The order of `LOCGRID` statements is critical (see Notes).

`xNum yNum zNum` (*integer*, min:`2`) number of grid nodes in the x, y and z directions

`xOrig yOrig zOrig` (*float*) x, y and z location of the grid origin in km relative to the geographic origin. Use a large, negative value (*i.e.* `-1.0e30`) to indicate automatic positioning of grid along corressponding direction (valid for nested grids only, may not be used for initial grid).

`dx dy dz` (*float*) grid node spacing in kilometers along the x, y and z axes

`gridType` (*choice*: `MISFIT PROB_DENSITY`) statistical quantity to calculate on grid

`saveFlag` (*choice*: `SAVE NO_SAVE`) specifies if the results of the search over this grid should be saved to disk

Notes:
1. The order of `LOCGRID` statements is critical: the first `LOCGRID` is the initial search grid which may not have automatic positionig along any axes. The succeeding `LOCGRID` statements may specify automatic positioning along one or more axes (***xOrig, yOrig, zOrig*** = `-1.0e30`), but must all be sized (*i.e.* ***dx*(xNum−1)*** , etc.) so that they can be fully contained within the preceeding grid. The NLLoc program will attempt to translate a nested grid that intersects a boundary of the initial grid so that it is contained inside of the initial grid; if this is not possible the

location will be terminated prematurely.

2. With automatic positioning (***xOrig, yOrig, zOrig*** = **-1.0e30**), a grid is shifted in x/y/z so that it is centered on the minimum misfit hypocenter x/y/z of the preceeding grid.

3. Each search over a grid with ***gridType*** = **PROB_DENSITY** is time consuming and should generally only be used for a nested grid on which the full PDF is required and will be saved to disk. Use ***gridType*** = **MISFIT** for the initial grid, for larger nested grids, and for smaller nested grids in maximum−likelihood hypocenter searches (*i.e.* where the PDF is not if interest).

4. The 3D grid dimensions are in kilometers with Z positive down (left−handed co−ordinate system).

5. The grid is *dx*(xNum−1)* km long in the x direction, and similarly for y and z.

LOCPHSTAT − Phase Statistics parameters

optional, *non−repeatable*

Syntax 1: **LOCPHSTAT** ***RMS_Max*** ***NRdgs_Min*** ***Gap_Max*** ***P_ResidualMax*** ***S_ResidualMax***

Specifies selection criteria for phase residuals to be included in calculation of average P and S station residuals. The average residuals are saved to a summary, phase statistics file (see Phase Statistics file formats).

RMS_Max (*float*, default:**VERY_LARGE_DOUBLE**) the maximum allowed hypocenter RMS in seconds

NRdgs_Min (*integer*, default:**-1**) the minimum allowed hypocenter number of readings

Gap_Max (*float*, default:**VERY_LARGE_DOUBLE**) the maximum allowed hypocenter gap in degrees

P_ResidualMax ***S_ResidualMax*** (*float*, default:**VERY_LARGE_DOUBLE**) the maximum allowed residual in seconds for a P or S phase

Notes:

1. Because the maximum residual cut−off is abrupt, it should be chosen and used with care.

2. In the current version of NLLoc, it is assumed in the calculation of average P and S station residuals that the standardized phase codes are **P** and **S** . Thus it is important to use these codes, if possible.

LOCANGLES − Take−off Angles parameters

optional, *non−repeatable*

Syntax 1: **LOCANGLES** ***angleMode*** ***qualtiyMin***

Specifies whether to determine take−off angles for the maximum likelihood hypocenter and sets minimum quality cut−off for saving angles and corresponding phases to the HypoInverse Archive file .

angleMode (*choice*: **ANGLES_YES ANGLES_NO**, default:**ANGLES_YES**) sets if take−off angles are read from angles grid files and output to locations files. (**ANGLES_YES** for angles determination or **ANGLES_NO** for no angles determination)

qualtiyMin (*integer*, default:**5**) sets the minimum quality (see Take−Off Angles Algorithm) for writing take−off angles and corresponding phase to the HypoInverse Archive file . (**0** to **10**)

LOCMAG − Magnitude Calculation Method

optional, *non−repeatable*

Syntax 1: **LOCMAG** ***ML_HB f n K***

Syntax 2: **LOCMAG** ***MD_FMAG c1 c2 c3 c4 c5***

Specifies the magnitude calculation type and parameters. The possible magnitude types are:

ML_HB (Local (Richter) magnitude *ML* from Hutton and Boore (1987)),

$$ML = \log(A\,f) + n\log(r/100) + K(r-100) + 3.0 + S,$$

MD_FMAG (Duration magnitude *ML* from Lahr, J.C., (1989) HYPOELLIPSE),

$$MD = C1 + C2\log(Fc) + C3r + C4z + C5[\log(Fc)]^2,$$

f (*float*, min:**0.0**) scaling factor to convert *A* to an equivalent Wood−Anderson amplitude.

n (*float*) *n* from Hutton and Boore (1987), related to geometrical spreading.

K (*float*) *K* from Hutton and Boore (1987).

c1 c2 c3 c4 c5 (*float*) *c1 c2 c3 c4 c5* from Lahr, J.C., (1989) HYPOELLIPSE

LOCCMP − Magnitude Calculation Component

optional, *repeatable*

Syntax 1: **`LOCCMP`** ***`label inst comp ampFactor sta_corr_ml_hb sta_corr_fd_fmag`***

`label` (*string*) station label (*i.e.* a station code: **`ABC`**)

`inst` (*string*) instrument identification (*i.e.* **`SP, BRB, VBB`**)

`comp` (*string*) component identification (*i.e.* **`Z, N, E, H`**)

`ampFactor` (*float*, min:**`0.0`**) amplitude factor, amplitude read from phase file is multiplied by ***ampFactor*** to obtain the amplitude used for magnitude calculation.

`sta_corr_ml_hb` (*float*) **`ML_HB`** station correction, from Hutton and Boore (1987)

`sta_corr_fd_fmag` (*float*) **`FD_FMAG`** station correction, from Lahr, J.C., (1989) HYPOELLIPSE

Notes:
1. Component specific paramaters are applied to all phase observations with matching label, instrument and component. Use **`?`** or **`*`** to disable matching of label, instrument or component.

LOCALIAS − Station Code Alias

optional, *repeatable*

Syntax 1: **`LOCALIAS`** ***`code alias yearStart monthStart dayStart yearEnd monthEnd dayEnd`***

Specifies (1) an alias (mapping) of station codes, and (2) start and end dates of validity of the alias. Allows (1) station codes that vary over time or in different pick files to be homogenized to match codes in time grid files, and (2) allows selection of station data by time.

`code` (*string*) station code (or station name or source label) as read from the phase/observation files, or from the result of another alias evaluation

`alias` (*string*) new station code which will replace ***code*** if the relevant phase pick time falls within the start and end dates of validity of the alias

`yearStart monthStart dayStart` (*integer*) year (including century), month and day of start date of validity of the alias (**`0 0 0`** = no start date)

`yearEnd monthEnd dayEnd` (*integer*) year (including century), month and day of end date of validity of the alias (**`9999 99 99`** = no end date)

Notes:

1. In NLLoc, the alias evaluation is applied recursively, regardless of the order of the **LOCALIAS** statements. Thus, when selecting and specifying alias names, beware of infinite recursion.

2. A trailing underscore "_" in an alias will only be used for time grid identification, not for output. This allows, for example, a station name **ABC** to be aliases to the name **ABC_** to enforce certain dates of validity for the station, this requires that the time grids generated by Grid2Time use the station code **ABC_** ; in all NLLoc output, the code **ABC** will be used.

LOCEXCLUDE − Exclude Observations

optional, *repeatable* (**ver 2.0**)

Syntax 1: **LOCEXCLUDE** ***name phase***

name (*string*) station label (*i.e.* a station code: **ABC**) identifier after application of any alias

phase (*string*) phase code beofore mapping by **LOCPHASEID** (**P** , **S** , **PN** , etc).

Notes:

1. Excluded station/phase observations are weighted to 0 and so will not be used for location. The residual is calculated for these observations and they are written to output files, if a travel time is available.

LOCDELAY − Phase Time Delays

optional, *repeatable*

Syntax 1: **LOCDELAY** ***code phase numReadings delay***

Specifies P and S delays (station corrections) to be subtracted from observed P and S times.

code (*string*) station code (after all alias evaluations)

phase (*string*) phase type (**P** or **S**)

numReadings (*integer*) number of residuals used to calculate mean residual/delay (not used by NLLoc, included for compatibility with the format of a summary, phase statistics file)

delay (*float*) delay in seconds, subtracted from observed time

Notes:

1. The body of a summary, phase statistics file (see Phase Statistics file formats) can be used directly as a set of **LOCDELAY** statements. Thus the average phase residuals from a run of NLLoc can be used as the station corrections for later runs of NLLoc.

17. References

Hutton, L.K., and Boore, D.M., 1987. The ML scale in Southern California, *Bull. Seism. Soc. Am.*, **77**, 2074−2094.

Lahr, J.C., 1989. HYPOELLIPSE − Version 2.0: A computer program for determining local earthquake hypocentral parameters, magnitude and first motion pattern, *U.S. Geological Survey Open−File Report 89−116*, 92p.

Le Meur, Helene, 1994. Tomographie tridimensionnelle a partir des temps des premieres arrivees des ondes P et S, application a la region de Patras (Grece), These de Doctorat, Paris VII, France.

Le Meur H, J. Virieux et P. Podvin, 1997. Seismic Tomography of the gulf of Corinth: a comparison of methods, *Annali di Geofisica*, **40**, 6, 1−24.

Lee, W.H.K. and Lahr, J.C., 1975. HYPO71 (revised): A computer program for determining hypocenter, magnitude and first motion pattern of local earthquakes, *U.S. Geological Survey Open−File Report 75−311*, 116p.

Lienert, B.R., 1994. HYPOCENTER 3.2: A computer program for locating earthquakes locally, regionally and globally, *Hawaii Institute of Geophysics & Planetology* (lienert@soest.hawaii.edu), 70p.

Lomax, A., J. Virieux, P. Volant and C. Berge, 2000. Probabilistic earthquake location in 3D and layered models: Introduction of a Metropolis−Gibbs method and comparison with linear locations, in *Advances in Seismic Event Location* Thurber, C.H., and N. Rabinowitz (eds.), Kluwer, Amsterdam, 101−134..

Lomax, A., A. Zollo, P. Capuano, and J. Virieux, 2001. Precise, absoute earthquake location under Somma−Vesuvius volcano using a new 3D velocity model, *Gephys. J. Int.*, *xxx*, **nnn**, nnn−nnn. ***Accepted, in press.***

Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P., 1983. Optimization by simulated annealing, *Science*, **220**, 671−680.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., and Teller, E., 1953. Equation of state calculations by fast computing machines, *J. Chem. Phys.*, **1**, 1087−1092.

Mosegaard, K., and Tarantola, A., 1995, Monte Carlo sampling of solutions to inverse problems, *J. Geophys. Res.*, **100**, 12431−12447.

Moser, T.J., T. van Eck and Nolet, G., 1992, Hypocenter determination in strongly heterogeneous earth models using the shortest path method., *J. Geophys. Res.*, **97**, 6563−6572.

Nelson, G.D. and Vidale, J.E., 1990, Earthquake locations by 3−D finite−difference travel times, *Bull. Seism. Soc. Am.*, **80**, 395−410.

Podvin, P. and Lecomte, I., 1991, Finite difference computation of traveltimes in very contrasted velocity models: a massively parallel approach and its associated tools., *Geophys. J. Int.*, **105**, 271−284.

Press, W.H., Teukolosky, S.A., Vetterling, W.T. and Flannery, B.P., 1992, *Numerical recipies in C: the art of scientific computing* or *Numerical recipies in FORTRAN: the art of scientific computing*, Cambridge University Press, Cambridge.

Reasenberg, P. A., and Oppenheimer, D., 1985, FPFIT, FPPLOT, and FPPAGE: Fortran computer programs for calculating and displaying earthquake fault−plane solutions, *U.S. Geological Survey Open−File Report 85−739*.

Sen, M. and Stoffa, P.L., 1995, *Global optimization methods in geophysical inversion*, Elsevier, Amsterdam, 281.

Shearer, P.M., 1997, Improving local earthquake locations using the L1 norm and waveform cross correlation: Application to the Whittier Narrows, California, aftershock sequence., *J. Geophys. Res.*, **102**, 8269−8283.

Tarantola, A. and Valette, B., 1982, Inverse problems = quest for information., *J. Geophys.*, **50**, 159−170.

Tarantola, A., 1987, *Inverse problem theory: Methods for data fitting and model parameter estimation*, Elsevier, Amsterdam, 613p.

Wittlinger, G., Herquel G., and Nakache, T., 1993, Earthquake location in strongly heterogeneous media, *Geophys. J. Int.*, **115**, 759−777.

Appendix 1 − Seismicity Viewer software guide (April 2001)

The **Seismicity Viewer** is a Java program for interactive viewing of earthquake locations and geographic information in a 3D space.

Overview − Copyright notice, Source code − Installing and running: stand−alone − Installing and running: applet − Parameter arguments − Seismicity Viewer File Formats

Overview

Seismicity Viewer displays earthquake locations in a 3D volume as probability density functions (PDFs), maximum−likelihood hypocenters, Gaussian expectation hypocenters, Gaussian confidence ellipsoids, and focal mechanisms. Station locations, associated P and S residuals, and geographic and geologic features are also displayed. The 3D volume can be interactively rotated, zoomed and viewed along the co−ordinate axes in perspective or orthographic projections. The events can viewed together, individually, or in an animated sequence.

View an example earthquake location with the Seismicity Viewer at http://www.alomax.net/seismicity/VentimigliaMain.html.

The event data can be in NonLinLoc Hypocenter−Phase File format or in several other formats (SEISAN, ORFEUS−Spyder...). (Conversion of HYPOELLIPSE summary or FPFIT summary formats to NonLinLoc Hypocenter−Phase format is available with the NonLinLoc Utility programs.)

Seismicity Viewer displays events in a local/region mode using rectangular Cartesian co−ordinates, and in a global mode using spherical co−ordinates (default if no NonLinLoc Grid Header file is used; see below).

The Seismicity Viewer is written in the Java language.

The Seismicity Viewer can be run as:

- A stand−alone application (**Seismicity**) that runs locally and which can read data files locally or across the Internet. Seismicity Viewer stand−alone will run on any type of computer which has the runtime for Java 1.1 or higher installed. Additional features of the stand−alone application Seismicity Viewer include:
  - direct reading of data files over the Internet from any server
  - printing
- An "applet" (**SeismicityApp**) that is installed on a central web server and which is launched

over the Internet from an HTML web page on the server. SeismicityApp "applet" can be viewed from any type of computer with Internet access and a recent web browser that supports Java 1.1 or higher (i.e. Netscape 4 or higher, Internet Explorer 5 or higher). The default applet security settings restrict SeismicityApp to reading data files only from the same server from which it was launched. These security restrictions also prevent or limit the printing and writing of data files.

Important note

This page gives instructions and links for downloading and installing the Seismicity Viewer v2.0.

NOTE: Parts of this software are new and have not yet been reviewed or thoroughly tested − IT IS IMPORTANT THAT YOU VERIFY THE CORRECT FUNCTIONING OF THE SOFTWARE FOR YOUR PURPOSES.

If you have problems or suggestions, please contact Anthony Lomax; please specify the type and version of your computer hardware, Java installation, and browser. I would also appreciate hearing about the details of your installation and application of Seismicity Viewer.

Copyright notice

The Seismicity Viewer program and source code are distributed under the terms of the GNU General Public License (GNU GPL). PLEASE READ, UNDERSTAND AND AGREE WITH THE GNU GPL BEFORE DOWNLOADING OR USING THE SEISMICITY VIEWER SOFTWARE: GNU General Public License (GNU GPL http://www.gnu.org/copyleft/gpl.html).

Source code

The Seismicity Viewer is currently under development. If you would like a copy of the source code in its current state, please contact Anthony Lomax.

Installing and running

Seismicity Viewer Stand−alone

Download and install Seismicity Viewer version 2.1 *stand−alone* by following the steps below:

1. **Install Java SDK, JDK or JRE 1.1 or higher:**

   You can check for the presence and version of Java on your system with the command:

   ```
   java -version
   ```

or

```
jre
```

If you do not have Java SDK/JDK (Java Software Development Kit) or JRE (Java Runtime Environment) 1.1 or higher available on your system, you can download Java from Sun for Solaris, Windows 95/98/NT or LINUX (Java2) free of charge. For a minimum installation use the JRE for Java 2 or Java 1.1. For other operating systems (LINUX, MacOS, etc) see Sun's Java(TM) Platform Ports page.

If not done automatically at installation, remember to update the `path` environment variable (**UNIX**) or the `PATH` variable (**Windows**) if you want to be able to run the Java executables (`java`, `jar`, etc.) from any directory without having to type the full path of the command. See the Java installation documentation for more details.

2. **Create the installation directories and download the Seismicity Viewer program files:**

   Create a program directory for the Seismicity Viewer class files, i.e.

   my_java_dir/seismicity

   Download the following files to the program directory (To download from a browser directly to local disk, press the right mouse button over each link and select "Save Link As..." (Netscape) or "Save Target As..." (IE), or press Shift+Button1 (Netscape)):

   SeisView20_global.jar − Seismicity Viewer class files (the Java "byte−code" for Seismicity Viewer − DO NOT UNPACK THIS FILE))

   Create a work directory for the Seismicity Viewer sample data files, i.e.

   my_work_dir

   Download the sample data files to the work directory:

   vinti.hyp − NonLinLoc Hypocenter−Phase file
   vinti.hdr − NonLinLoc Grid Header file
   vinti.cont.xyz − topography 3D line file
   vinti.text − 3D text file

   Download the defaults file to the work directory:

   seismicitydefaults (or .seismicitydefaults) − Seismicity Viewer defaults file (the user default preferences for Seismicity Viewer, and specifications of line files, text file and other display parameters)

3. **Set the CLASSPATH**

   The `CLASSPATH` environment variable tells the Java Virtual Machine and other Java applications where to find the class libraries, including user−defined class libraries. If you are only running Seismicity Viewer you can set the `CLASSPATH` as follows:

UNIX:

```
setenv CLASSPATH
my_java_dir/seismicity/SeisView20_global.jar
```
(*all on one command line*)

Windows:

```
set CLASSPATH =
my_java_dir\seismicity\SeisView20_global.jar
```
(*all on one command line*)

In these expressions, *my_java_dir* is substituted by the absolute path to the Java SDK/JDK or JRE installation directory directory.

The full name of jar files (compressed archives containing class and other files) must always be explicitly listed in the `CLASSPATH`. To access un−archived class files (*.class) directly you must include in the `CLASSPATH` the path to the directory containing the *.class files.

Alternatively, the `−classpath` command−line switch to the Java executables can be used to specify the location of class files. Sun suggests the use of this switch and not the `CLASSPATH` variable. See the Java installation documentation for more details.

NOTE: JRE (1.1 only?) on Windows ignores the `CLASSPATH` environment variable and the `−classpath` command−line switch. Instead use the `−cp` command−line switch. See the JRE README installation documentation for more details.

4. **Run Seismicity Viewer**

Run Seismicity Viewer and view the sample data files:

UNIX:

Go to the work directory created above and type:

```
java seismicity.Seismicity Viewer -help
```
(*if your are using the* `CLASSPATH` *environment variable*)

or

```
java -classpath
my_java_dir/seismicity/SeisView20_global.jar \
    seismicity.Seismicity -help
```
(*all on one command line; if your are using the* `−classpath` *command−line switch*)

You should see the Seismicity Viewer usage instructions and a list of command line arguments. Next, type:

```
java seismicity.Seismicity vinti.hyp
```
(*all on one command line; add the* `−classpath` *command−line switch if*

necessary)

This should launch the Seismicity Viewer viewer and read and display an earthquake location. Read the documentation in the Help−>Help sub−menu, and try the various Seismicity Viewer options for "viewing" the earthquake.

The launcing of Seismicity Viewer can be simpified by the use of an alias, i.e.

```
alias svw 'java seismicity.Seismicity'
```

Windows (use `jre` or `jrew` in place of `java`, as appropriate):

Open a DOS console window by selecting Start−>Programs−>MS DOS Prompt.

Make sure in the new DOS console window that the `PATH` includes the path to the Java executables, and (if you are not using the `-classpath` command−line switch) that the `CLASSPATH` is set as described above. (You can check the environment by typing the DOS command `set`. `PATH` and `CLASSPATH` may be set in `AUTOEXEC.BAT`, in another batch file or from the command line.)

Go to the work directory created above and type:

```
java seismicity.Seismicity -help
```
(*add the* `-classpath` *command−line switch if necessary*)

or

```
jre -cp my_java_dir\seismicity\SeisView20_global.jar
seismicity.Seismicity -help
```
(*all on one command line*)

You should see the Seismicity Viewer usage instructions and a list of command line arguments. Next, type:

```
java seismicity.Seismicity vinti.hyp
```
(*all on one command line; add the* `-classpath` *command−line switch if necessary*)

This should launch the Seismicity Viewer viewer and read and display an earthquake location. Read the documentation in the Help−>Help sub−menu, and try the various Seismicity Viewer options for "viewing" the seismogram. (Note that wildcard characters (*,?) can be used in filenames on a local file system.)

The launcing of Seismicity Viewer can be simpified by the use of a batch file, i.e. `sgm.bat` containing, for example,

```
java seismicity.Seismicity %1 %2 %3
```

NOTE: With the JRE command `jrew` (Java 1.1 only?) you can launch Seismicity Viewer directly from the Windows Start menu without the need to launch a DOS console window. See the JRE README installation documentation for more details.

Read data over the Internet:

You can read data files directly over the internet by giving the full URL (Uniform Resource Locator) address of one or more seismogram trace files as arguments to the Seismicity Viewer command, i.e.

```
java seismicity.Seismicity
http://www.alomax.net/seismicity/data/vinti.hyp
```
(*all on one command line; add the* `-classpath` *command-line switch if necessary.*)

or by entering or pasting the URL in the File->OpenURL dialog of Seismicity Viewer stand-alone. (Note that wildcard characters (*,?) cannot be used in URL's.)

To view global event for 1999 in Seismicity Viewer global mode, try:

```
java seismicity.Seismicity
http://orfeus.knmi.nl/ODC_Data/new_spyder_interface/lo
c_files/loc1999
```
(*all on one command line; add the* `-classpath` *command-line switch if necessary.*)

Seismicity Viewer applet

To install and run the Seismicity Viewer you must create an HTML document containing an `<APPLET>` tag that loads the Seismicity Viewer applet code and specifies various attributes using `<PARAM>` tags. You must also put all of the Seismicity Viewer compiled code (`*.jar` and/or `*.class` files) and event and geographic data files in a public directory (for browser use) or other convenient directories. The Seismicity Viewer can then be run by opening this HTML document in a web browser.

The HTML document has the following form:

```
<HTML>
<HEAD>
<TITLE>Seismicity Viewer</TITLE>
</HEAD>
<BODY>

<APPLET CODEBASE=codebase CODE=SeismicityApp.class
WIDTH=width HEIGHT=height>

<PARAM NAME=appletAttribute1 VALUE=value1>
...
<PARAM NAME=appletAttributeN VALUE=valueN>

</APPLET>

</BODY>
</HTML>
```

where:

`codebase` (*chars*)
: is an optional attribute specifies the base URL of the applet − the directory that contains the applet's `*.class` files. If this attribute is not specified, then the HTML document's URL is used.

`width height` (*integers*)
: are required attributes give the initial width and height (in pixels) of the applet display area.

`<PARAM NAME=`*`appletAttribute1`* `VALUE=`*`value1`*`>`
: are tags that specify Seismicity Viewer program parameter arguments (see below).

Notes:

1. For more information on the `<APPLET>` tag see the Sun site `<APPLET>` tag documentation.
2. Applets have strong security restrictions that limit their functionality. For example, applets loaded over the internet can only open files on the same server they were loaded from, they cannot open or write to files on the local system. See the Sun site Applet Security page for more information.

Download and install Seismicity Viewer applet by following the steps below:

1. **Install an appropriate browser:**

   To view the Seismicity Viewer applet, you need a browser with Java 1.1 or higher installed and enabled. Netscape 4.5 or higher, Internet Explorer 5, or equivalent is recommended. (You can also use the `appletviewer` program to view the Seismicity Viewer applet if you have the Java SDK/JDK available on your system. See the SDK/JDK documentation for more information.)

2. **Create the html and installation directories and download the Seismicity Viewer program files:**

   The Seismicity Viewer applet is launched from an HTML page in a public space on a web server (a directory accessible from from the Internet with a URL, i.e. http://www.alomax.net/seismicity). Default applet security restrictions limit applets to reading files only from the server from which it was launched, and default local security restrictions limit locally run applet to reading files only under the directory from which it is launched (The directory specified by the `codebase` applet tag, if this tag exists, otherwise the directory containing the HTML page). Thus, for your first Seismicity Viewer applet installation, it is easiest to begin with a directory structure with the HTML page and Seismicity Viewer program files in a top directory, and the data files in a sub−directory.

   Create the top directory for the HTML files and Seismicity Viewer program files, i.e.

   ```
   .../public_html/seismicity
   ```

   Download the following files to the top directory (To download from a browser directly to local disk, press the right mouse button over each link and select "Save Link As..." (Netscape) or "Save Target As..." (IE), or press Shift+Button1 (Netscape)):

Seismicity_basic_applet.html − HTML text file that will launch the Seismicity Viewer applet
SeisView20_global.jar − Seismicity Viewer class files (the Java "byte−code" for Seismicity Viewer − DO NOT UNPACK THIS FILE))

Directly under the top directory, create a sub−directory named "data" for the data files, i.e.

```
.../public_html/seismicity/data
```

Download the following data files to this sub−directory:

vinti.hyp − NonLinLoc Hypocenter−Phase file
vinti.hdr − NonLinLoc Grid Header file
vinti.cont.xyz − topography 3D line file
vinti.text − 3D text file

3. **Run your browser and launch the Seismicity Viewer applet**

   Start your browser and open the HTML page Seismicity_basic_applet.html.

   Click on the link. This will load the Seismicity Viewer applet, and read and display a 3−component seismogram. Read the documentation in the Help−>Help sub−menu, and try the various Seismicity Viewer options for "viewing" the seismogram.

   (Click here Seismicity_basic_applet.html to see how the Seismicity Viewer applet loads directly from the Seismicity Viewer site.)

Program parameter arguments

The Seismicity Viewer accepts a number of parameter tag attributes to specify an event location file name, geographic data files and other plotting parameters.

For the stand−alone version of Seismicity Viewer, each parameter is given as a command line argument of the form:

`-`*`name`*`=`*`value`*

For the applet version of Seismicity Viewer, each parameter is contained in an HTML tag has the form:

`<PARAM NAME=`*`name`*` VALUE=`*`value`*`>`

See the Seismicity Viewer Help for a complete list of parameters. Some of the principal parameters arguments are:

`NAME=`**`grid`** `VALUE=`*`GridHeaderFileURL`* (*chars*, *required*)
: the URL (Uniform Resouce Locator − file name or internet link) of a NonLinLoc 3D Grid Header file. The bounds of the 3D grid will determine the size of the default view and are plotted as a wire frame. Only the first line of the 3D Grid Header file should be present. Optionally, a second line in the format of the NonLinLoc Generic Control Statement TRANS may be included to specify a geographic transformation if one is not specified in an event file.

`NAME=`**`event`** `VALUE=`*`LocationFileURL`* (*chars*, *optional*)
: the URL (Uniform Resouce Locator − file name or internet link) of an ASCII NonLinLoc Hypocenter−Phase File, or another file type as specified in the `NAME=`**`eventfiletype`** parameter tag, or a Java binary event file ending in `.jbin`. The events in this file will be displayed by the viewer.

`NAME=`**`eventfiletype`** `VALUE=`*`eventFileType`* (*chars*, *optional*)
: the event file format (`NLLOC` = ASCII NonLinLoc Hypocenter−Phase File; `HYPO` = NonLinLoc Quasi−Hypoellipse hypocenter file; `SEISAN` = SEISAN hypocenter summary file) Conversion of HYPOELLIPSE summary or FPFIT summary formats to NonLinLoc Hypocenter−Phase format is available with the NonLinLoc Utility programs.

`NAME=`**`lines.white`** `VALUE=`*`geogLineFileURL#lineName`* (*chars*, *optional*)
: the URL (Uniform Resouce Locator − file name or internet link) of a Seismicity Viewer Geographic line file. The lines in the file *`geogLineFileURL`* will be plotted in 3D in color white. The optional string *`lineName`* will be the label on a button to toggle the line display on and off.

`NAME=`**`lines.white`**, `NAME=`**`lines.black`**, `NAME=`**`lines.blue`**, `NAME=`**`lines.cyan`**, `NAME=`**`lines.darkGray`**, `NAME=`**`lines.gray`**, `NAME=`**`lines.lightGray`**, `NAME=`**`lines.green`**, `NAME=`**`lines.magenta`**, `NAME=`**`lines.orange`**, `NAME=`**`lines.pink`**, `NAME=`**`lines.red`**, `NAME=`**`lines.yellow`** (*chars*, *optional*)
: the same as above for other line colors.

`NAME=`**`text`** `VALUE=`*`geogTextFileURL`* (*chars*, *optional*)
: the URL (Uniform Resouce Locator − file name or internet link) of a Seismicity Viewer Geographic text file. The text in the file *`geogTextFileURL`* will be plotted in 3D. A button withe the label `Text` allows on/off toggling of the text display.

`NAME=`**`show.probability`** `VALUE=YES` *`or`* `NO` (*boolean*, *optional*, default=`YES`)
: if `YES` then event probability density scatter clouds will be plotted on the initial display, otherwise they will not be plotted

`NAME=`**`init.plotfirst`** `VALUE=YES` *`or`* `NO` (*boolean*, *optional*, default=`NO`)
: if `YES` then only the first location in the event file will be plotted on the initial display, otherwise all events will be plotted

`NAME=`**`read.ellipsoids`** `VALUE=YES` *`or`* `NO` (*boolean*, *optional*, default=`YES`)
: if `YES` then confidence ellipsoids are read from the event file and stored in memory, otherwise they are ignored. The storage of confidence ellipsoids can use excessive memory for large event files.

`NAME=`**`read.mechanisms`** `VALUE=YES` *`or`* `NO` (*boolean*, *optional*, default=`YES`)
: if `YES` then focal mechanisms are read from the event file and stored in memory, otherwise they are ignored. The storage of mechanisms can use excessive memory for large event files.

`NAME=`**`read.phases`** `VALUE=YES` *`or`* `NO` (*boolean*, *optional*, default=`YES`)
: if `YES` then individual phase data are read from the event file and stored in memory, otherwise they are ignored. The storage of phase data can use excessive memory for large event files.

Seismicity Viewer File Formats

Geographic line file − Geographic text file

- **Geographic line file** (*ASCII*)

  Contains 2D or 3D co−ordinates of one or more line segments.

  Line 1: (*required*) Specifies the type of the Geographic line file.

  `> fileFormat comment`

  Fields:

  `fileFormat` (*chars*)
  specifies the type of data and format of each line:
  `XY_` = x y (*floats*, z is set to 0.0 km);
  `XYZ` = x y z (*floats*, z positive up);
  `GMT_LATLONDEPTH` = latitude longitude depth (*floats*, depth in kilometers positive down);
  `GMT_LATLONELEV_M` = latitude longitude elevation (*floats*, elevation in meters positive up);
  `GMT_LONLATELEV_M` = longitude latitude elevation (*floats*, elevation in meters positive up);
  `GMT_LONLAT` = longitude latitude (*floats*, z is set to 0.0 km);
  `GMT_LATLON` = latitude longitude (*floats*, z is set to 0.0 km);
  `comment` (*chars*)
  a comment string which is ignored

  Notes:

  1. Each line of the file specifies the co−ordinates of one point on the line.
  2. x, y and z values are in kilometers, latitude and longitude values are in degrees
  3. Separate line segments must be separated by a line containing only the > character
  4. The program aborts with an error if a format with latitude and longitude values are used and a geographic transformation is not specified in the event file or in the 3D Grid Header file.

  Example:

  > GMT_LONLATELEV_M −− 200m contours

  Remaining Lines: (*required*) Each line specifies the co−ordinates of one point on the geographic line following the `fileFormat` given in the first line of the file. A line containing only the > character and optional comment indicates the start of a new line segment.

Example:

14.3667 40.9285 −2800
14.3655 40.9292 −2800
14.3667 40.9302 −2800
> the 2600 meter contour
14.3667 40.9355 −2600
14.3708 40.9349 −2600
14.375 40.9335 −2600

- **Geographic text file** (*ASCII*)

  Contains 3D co−ordinates, orientation in 3D space and color of one or more text strings.

All lines: (*required*) Specifies the text string attributes.

theText horizAllign vertAllign XYZ *xRef yRef zRef xPath yPath zPath xUp yUp zUp redCol greenCol blueCol relSize*
or *theText horizAllign vertAllign* LATLON *latRef longRef zRef xPath yPath zPath xUp yUp zUp redCol greenCol blueCol relSize*
or *theText horizAllign vertAllign* LATLONDM *latDegRef latMinRef latNS longDegRef longMinRef longEW zRef xPath yPath zPath xUp yUp zUp redCol greenCol blueCol relSize*

Fields:

theText (*chars*)
the text string
horizAllign (*integer*)
the horizontal allignment of the text string with respect to the reference location (0 = beginning; 1 = middle; 2 = end;)
vertAllign (*integer*)
the vertical allignment of the text string with respect to the reference location (0 = top; 1 = middle; 2 = bottom;)
xRef yRef zRef (*floats*, z positive up)
the x, y and z co−ordinates of the reference location
xPath yPath zPath (*floats*, z positive up)
the x, y and z co−ordinates of a vector specifying the direction in 3D space of the text baseline
xUp yUp zUp (*floats*, z positive up)
the x, y and z co−ordinates of a vector specifying the direction in 3D space of the text up direction
redCol greenCol blueCol (*floats*)
the read, green and blue intensities for the text color (0.0 − 1.0)
relSize (*float*)

relative font size for the text characters

`latDegRef latMinRef` (*floats*)
: the integer degrees and decimal minutes of the latitude co−ordinates of the reference location

`latRef` (*float*)
: the latitude of the reference location

`longRef` (*float*)
: the longitude of the reference location

`latNS` (*char*)
: latiutude north or south character (`N` or `S`)

`longDegRef longMinRef` (*floats*)
: the integer degrees and decimal minutes of the longitude co−ordinates of the reference location

`longEW` (*char*)
: longitude east or west character (`E` or `W`)

Notes:

1. Each line of the file specifies the attributes of one text string.
2. x, y and z values are in kilometers, latitude and longitude values are in degrees

Example:

Ventimiglia 0 0 XYZ 9.0 4.5 0.0 1.0 1.0 1.0 −1.0 1.0 1.0 0.8 0.8 0.8 1.0
"shot B3" 0 0 LATLON 40.812219 14.403339 1.352100 1.0 1.0 1.0 −1.0 1.0 1.0 1.0 1.0 1.0 1.0
COLF 1 1 LATLONDM 43 01.13 N 12 53.07 E 0.904 1.0 0.0 0.0 0.0 1.0 0.0 0.4 1.0 0.4 1.0

Appendix 2 − GNU General Public License

(http://www.gnu.org/copyleft/gpl.html)

GNU General Public License

Table of Contents

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software−−to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- **a)** You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

- **b)** You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

- **c)** If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- **a)** Accompany it with the complete corresponding machine−readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

- **b)** Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine−readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

- **c)** Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty−free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices.

Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH

DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and an idea of what it does.
Copyright (C) yyyy  name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) yyyy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'.  This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.
```

The hypothetical commands `` `show w' `` and `` `show c' `` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `` `show w' `` and `` `show c' ``; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.
