User's Manual for *mloc*

v1.0

January 26, 2018

Dr. Eric Bergman

Global Seismological Services

http://www.seismo.com

bergman@seismo.com

Table of Contents

Introduction **6**

Development History 6

What is and is not in the Manual 6

Calibrated Earthquake Locations from a Multiple Event Relocation Analysis **8**

Abstract 8

Applicability 8

Location Calibration Methodology 8

Indirect Calibration *9*

Direct Calibration *11*

Groomed Readings *11*

Calibration Levels *13*

Focal Depth *14*

An Example *14*

Applications 17

References 18

Technical Aspects of Installing and Running mloc **19**

The Code 19

Utility Programs 19

Conversion Codes *19*

MNF Bulletin *19*

Editing *19*

Version History 20

Operating System and Supporting Software 20

Directory Structure for mloc 20

Top-Level Directories *21*

Configuration File for mloc 24

Filename and Path of the Configuration File *24*

Keywords *25*
A Sample Configuration File *25*
mloc Command Files **27**
Background 27
Processing a Command File 27
Command File Structure 27
Defaults 28
Defining Events 28
Terminating a Command File 28
MNF Native Format for mloc **29**
MNF v1.3 29
Background *29*
Description and Features *30*
Defined Record Types *34*
Changes from Previous Versions *39*
MNF v1.4 40
Background *40*
Description and Features *40*
Optional Fields *41*
Defined Record Types *42*
Changes from Previous Versions *45*
MNF v1.5 46
Background *46*
Parameterization of Differential Time Data *46*
Description and Features *47*
File Structure *48*
Defined Record Types *48*
Obtaining ISC Data for mloc **52**

Searching the ISC Bulletin 52
Conversion to an MNF Bulletin 53
Searching an MNF Bulletin and Extracting Files 54
Duplicate Readings *56*
Searching by Event Number *56*
Station File Formats for mloc 58
Format Index 58
Master Station File Format (isstn = 0) 58
Station Codes *59*
Operational Epoch *59*
Authorship *60*
Station Elevation and Depth of Burial *60*
Format *60*
Supplemental Station File Formats 61
Master Station File Format (isstn = 0) *61*
ISC Fixed Format (isstn = 1) *61*
SEISAN Station Format (isstn = 2) *61*
Simplified Format (isstn = 3) *62*
China Seismic Bureau Format (isstn = 4) *62*
NEIC Format (isstn = 5) *63*
MSU Format (isstn = 6) *64*
Managing Station Data in mloc 66
The Short Version 67
The New IASPEI Station Coding Standard 67
NEIC Station Metadata 68
Other Networks and Stations 68
Station files in mloc 68
Master Station File *69*

Supplemental Station Files *70*
Order of Precedence *70*
An Alternative to Supplemental Station Files *70*
Station Code Conflicts 71
Missing Station Codes 73
Skipping Stations 73
Using Deployment Codes to Resolve Station Code Conflicts 74

Introduction

This Manual deals with the installation and use of a program called "*mloc*" which implements the Hypocentroidal Decomposition (HD) method of multiple event relocation that was introduced in Jordan and Sverdrup (1981). The program is most commonly employed in the analysis of arrival time data from natural earthquakes but has also been used successfully on induced earthquakes and man-made explosions, including nuclear tests. It has been extensively developed for research on what are referred to as "calibrated" earthquake locations, which was not contemplated in the original publication. Calibrated locations arise from the use of procedures to minimize the biasing effect of unknown Earth structure and to more accurately estimate the uncertainties of the arrival time data which leads to more reliable identification and rejection of outlier readings and more accurate estimates of uncertainties for derived hypocentral parameters.

Development History

Development of *mloc* began in 1989 when the author was at the Massachusetts Institute of Technology along with Tom Jordan, who suggested looking into the HD algorithm as a research tool. The computer code used for the 1981 paper was not considered a suitable basis for development, so *mloc* was built on the code base of an advanced single-event location program called LOC that had been written in the early 1980s by Ken Creager. In its first incarnation, *mloc* was, like other multiple event relocation codes, oriented to obtaining improved relative locations of seismic events, but not calibrated locations.

The next phase of development of *mloc* began in the late 1990s, when the application to calibrated locations began in close collaboration with Bob Engdahl. During this period, many code segments from Engdahl's single event location code (the basis for the EHB Catalog) migrated to *mloc* and many detailed comparisons of intermediate results between the two codes were made to ensure correct processing.

Since about 2013 development of *mloc* has branched away from the EHB code, especially with the adoption of a new standard data format (MNF) that better supports the procedures used for calibrated relocations. Another major area of development has been refinement of methodologies to more reliably constrain focal depths.

What is and is not in the Manual

The Manual provides broad guidance on installing and running *mloc*, and there are detailed discussions of certain technical matters that often arise in practice, but it is not anticipated that most potential users would be able to use the Manual alone to navigate the extensive array of options in processing that are available and conduct a competent relocation analysis. *mloc* has a certain amount of built-in help for the commands that control processing, and the code itself carries extensive comments, but the concepts of relocation employed in *mloc* are foreign enough to most seismologists that an actual course of training is required to reach a basic level of competency. This Manual is therefore best viewed as a supplement to a training course. That is

why the next section of this Manual, before any discussion how to install and operate the program, is an introduction to the strategy for relocation that *mloc* implements. It is essential to have a basic understanding of the concepts presented in the next section before trying to use *mloc*.

Calibrated Earthquake Locations from a Multiple Event Relocation Analysis

Last updated: November 1, 2011

Abstract

Two methods for determining bias-free ("calibrated") locations for clusters of earthquakes are briefly described. They are both based on a multiple event relocation analysis (Hypocentroidal Decomposition, or HDC) using standard seismic bulletin arrival times, in which arrival time differences are used to constrain the relative locations and origin times of events in a cluster. The key to calibrated locations, however, is the exclusive use of near-source data to establish the location of the cluster in absolute spatial and temporal coordinates. One method of calibration is based on external information on the location of one or more cluster events. Such information can be derived from seismological, geological, or remote sensing data. The second method of calibration is based on analysis a subset of the arrival time information in the cluster at small epicentral distances. The statistical power of both methods is greatly enhanced by a careful analysis of empirical readings errors derived from the arrival time data, which are used both for weighting in the inversion for location and also for identifying outlier readings. Calibrated locations with uncertainties of 2-3 km are achievable with either method. An example is given of the indirect method of calibration, using a cluster of earthquakes in Kyrgyzstan.

Applicability

The methods of location calibration described here can be used at a great range of scales, from a handful of small events recorded locally to several hundred large events recorded globally. The algorithm is best-suited for data sets of about 200 events or less, and it is desirable to restrict the geographic extent of a cluster to approximately 100 km or less. There must be a minimal level of "connectivity" between events in a cluster, meaning repeated observations by the same seismic station. Since calibration depends on data at short range from the source, it is not possible to calibrate the locations of deep earthquakes, although the relative locations can be improved.

Location Calibration Methodology

The location calibration analysis is based on the Hypocentroidal Decomposition (HDC) method for multiple event relocation introduced and by (Jordan and Sverdrup, 1981). The basic algorithm is completely described in that reference. The essence of the HDC algorithm is the use of orthogonal projection operators to separate the relocation problem into two parts:

- The *cluster vectors*, which describe the relative locations in space and time of each event in the cluster. They are defined in kilometers and seconds, relative to the current position of the *hypocentroid*.

- The *hypocentroid*, which is defined as the centroid of the current locations of the cluster events. It is defined in geographic coordinates and Coordinated Universal Time (UTC).

The cluster vectors are defined only in relation to the hypocentroid. The hypocentroid can be thought of as a virtual event with geographic coordinates and origin time in UTC. The orthogonal projection operators act on the data set of arrival times to produce a data set that includes only data that actually bears on the relative location of cluster events, i.e., multiple reports of a given seismic phase at the same station for two or more events in the cluster.

The hypocentroid is located very much as an earthquake would be, except that the data are drawn from all the cluster events. Thus it is typical for the hypocentroid to be determined by many thousands of readings. Nevertheless, the hypocentroid is subject to unknown bias because the theoretical travel times (typically ak135) do not fully account for the three-dimensional velocity structure of the Earth. Geographic locations for the cluster events are found by adding the cluster vectors to the hypocentroid.

The HDC method works iteratively. At each iteration, two inversions are performed, first for the cluster vectors relative to the current hypocentroid, then for an improved hypocentroid. The cluster vectors are added to the new hypocentroid to obtain updated absolute coordinates for each event. The convergence criteria are based on the change in relative location of each event (0.5 km) and the change in the hypocentroid (0.005°). The convergence limits for origin time and depth, for cluster vectors and hypocentroid, are 0.1 s and 0.5 km, respectively. Convergence is normally reached in 2 or 3 iterations.

The data sets used for the two problems need not be (and usually are not) the same. Because the inverse problem for changes in cluster vectors is based solely on arrival time differences, baseline errors in the theoretical travel times drop out and it is desirable to use all available phases at all distances outside the immediate source region. For the hypocentroid, baseline errors in theoretical travel times are more important and one may wish to limit the data set to a phase set, e.g., teleseismic P arrivals in the range 30-90°, to achieve a more stable result. The choice of data set for determining the hypocentroid has great importance in the “direct” calibration method described below.

Similarly, weighting schemes are different for the two inversions, reflecting the different natures of the two problems. Empirical reading errors for each station-phase pair are used in weighting data for estimating both the hypocentroid and cluster vectors, but the uncertainty of the theoretical travel times, which are estimated empirically for each phase from the residuals of previous runs, is relevant only to the hypocentroid.

Until this point, the HDC algorithm is used only to obtain improved relative locations for the cluster events, with a geographic location for the cluster (the hypocentroid) that is biased to an unknown degree by unmodeled Earth structure convolved with the unbalanced distribution of reporting seismic stations. The calibration process attempts to remove this bias.

Calibration of a cluster is done in two ways, which we refer to as “indirect” and “direct” calibration.

Indirect Calibration

If the location and origin time of one or more of the cluster events can be specified with high accuracy from independent information, we can calibrate the entire cluster by shifting it in space and time to optimally match the known location of the calibration event(s). The approach is illustrated in Figure 1.

The most common source of such independent information are temporary seismic network deployments that capture an event with a large number of stations at very short epicentral distances, and which is also large enough to be well recorded at regional and teleseismic distances. Aftershock studies are a frequent source of such data. We normally obtain the temporary network arrival time data and relocate calibration events ourselves, using a local velocity model when available, to ensure reliable locations. We have also used InSAR data for this purpose, but this requires the use of at least some seismic data at short distance to calibrate origin time. In a few cases, mapped faulting from large events can be used to help constrain the location of calibration events.

When we use the direct calibration approach we take into account the uncertainty of the calibration data, and when there is more than one calibration event we also include a contribution to uncertainty to reflect any discrepancy between the relative locations of the calibration events and the cluster vectors of the corresponding events.

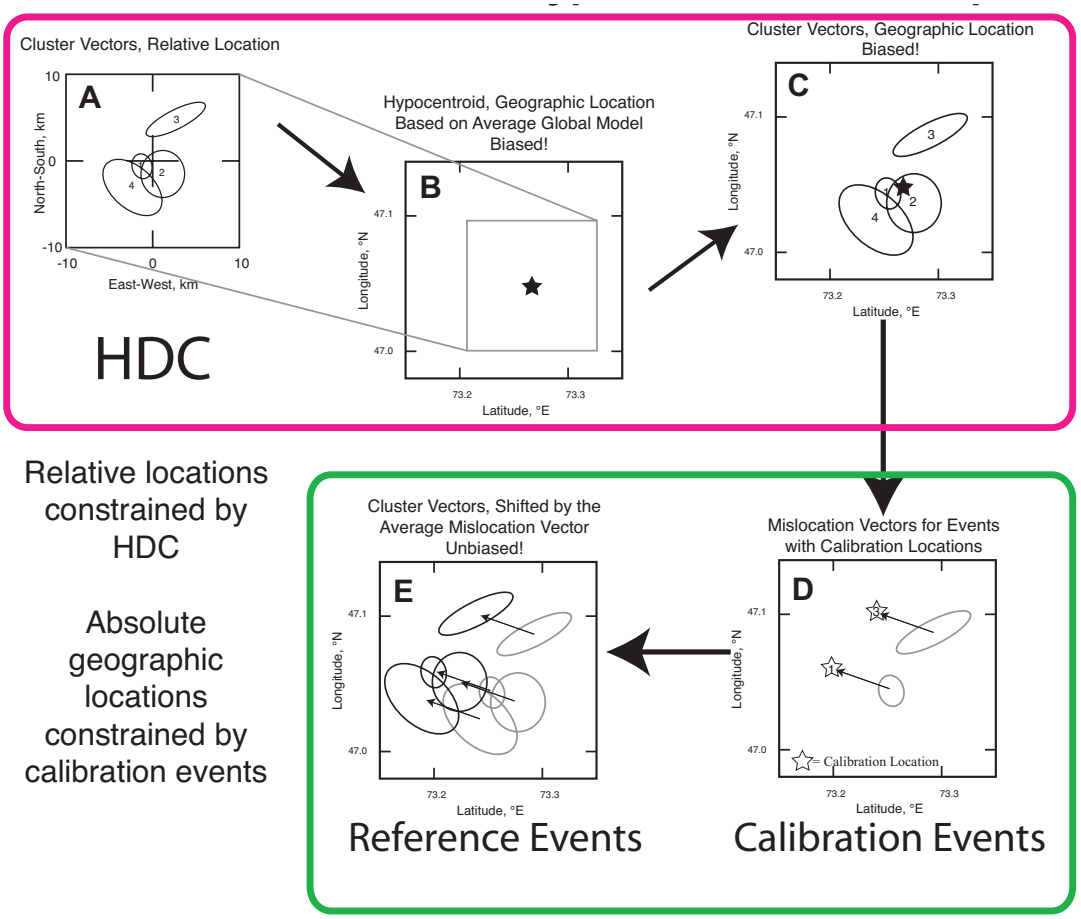

Figure 1. Indirect Calibration

Direct Calibration

It is sometimes the case that there are a few permanent seismic stations close to a cluster, but that no single event is well-enough recorded to reach the level of accuracy necessary to serve as a calibration event. On the other hand, the handful of local seismic stations may have recorded many events in the cluster, so that the number of "short distance" readings is rather large, and well-enough distributed to allow the hypocentroid to be located using only these data. We have found that good results can be obtained as long as the epicentral distance is kept less than about 150-200 km. At greater distances we often see rapidly increasing scatter in residuals at different azimuths. The process is illustrated in Figure 2, which can be contrasted with Figure 1.

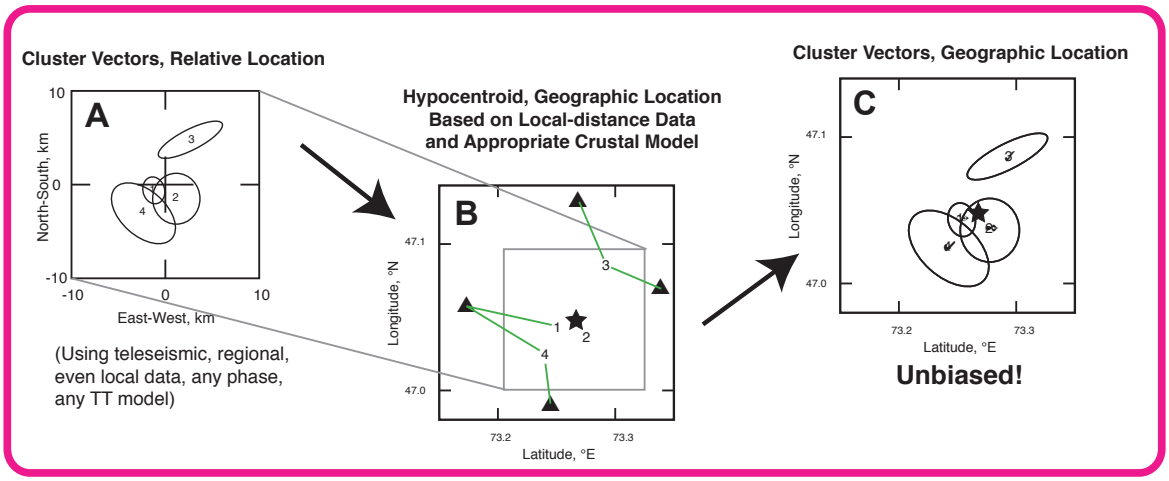

Figure 2. Direct Calibration

In any case where we have the data to locate one or more calibration events for the indirect method, we also have the option to use those same data in the direct calibration method. The decision on which to use is made on a case-by-case basis, because characteristics of the data sets may lead to a better result with one method than the other. Of course, if we are using InSAR data or other remote sensing or geological information to constrain the calibration, we must use the indirect method.

Groomed Readings

Our relocation and calibration methodology departs from standard practice in dealing with outlier arrival time readings by emphasizing consistency between repeated readings (of a given phase from different events in a cluster to the same station) rather than the size of residual against some reference travel time model. A considerable amount of work goes into estimation of what we refer to as "empirical reading errors" from the specific arrival time data set. This estimate is based on a robust estimator of spread (Croux and Rousseeuw, 1992) applied to the

travel time residuals for a specific station and phase. The estimator S_n is defined as

$$S_n = c_n 1.1926 \underset{i=1,\ldots,n}{lomed} \underset{j \neq i}{lomed} |x_i - x_j|$$

where "lomed" is the low median and the factor 1.1926 establishes consistency at a normal distribution. The term c_n is a correction term for small sample sizes, which has been tabulated. S_n makes no assumption about the underlying distribution and requires no estimate of central location. A minimum value of 0.15 s for S_n is normally enforced to avoid numerical instability.

In addition to their use in weighting the arrival time data for inversion, we also use empirical reading errors to detect outliers in the data, which are flagged. Because outlier readings can cause good readings to have large residuals, the process needs to be done incrementally, starting with the largest residuals, to avoid rejection of good data. We gradually remove the largest outliers, followed by relocation, until the normalized, de-meaned distribution of each station-phase approaches a Gaussian distribution. It is continued out until the residuals satisfy a 3σ criterion, using the current empirical reading error as the estimate of σ. This "cleaning" process is crucial in providing a self-consistent statistical framework for estimating location uncertainties. The process is illustrated in Figure 3.

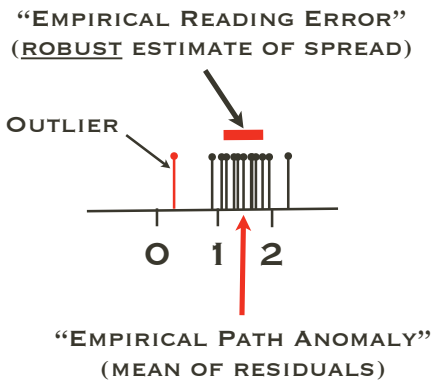

Figure 3. Outlier Removal

The effect of systematic and comprehensive (all phases and stations with 2 or more samples) estimation of empirical reading errors and associated outlier rejection is that the arrival time data set is reduced to one that more closely approaches the assumption of zero-mean, normally-distributed data which underlies the estimation of improved locations and their uncertainties. Gross errors are still possible from, for example, mis-identified phases, mis-associated readings, incorrect station locations, and temporary equipment or operational problems at stations. The distributions of empirical reading errors for the most common phases, measured from a calibrated event data set, and after cleaning, are shown in Figure 4. Note that the time scales differ significantly for different phases.

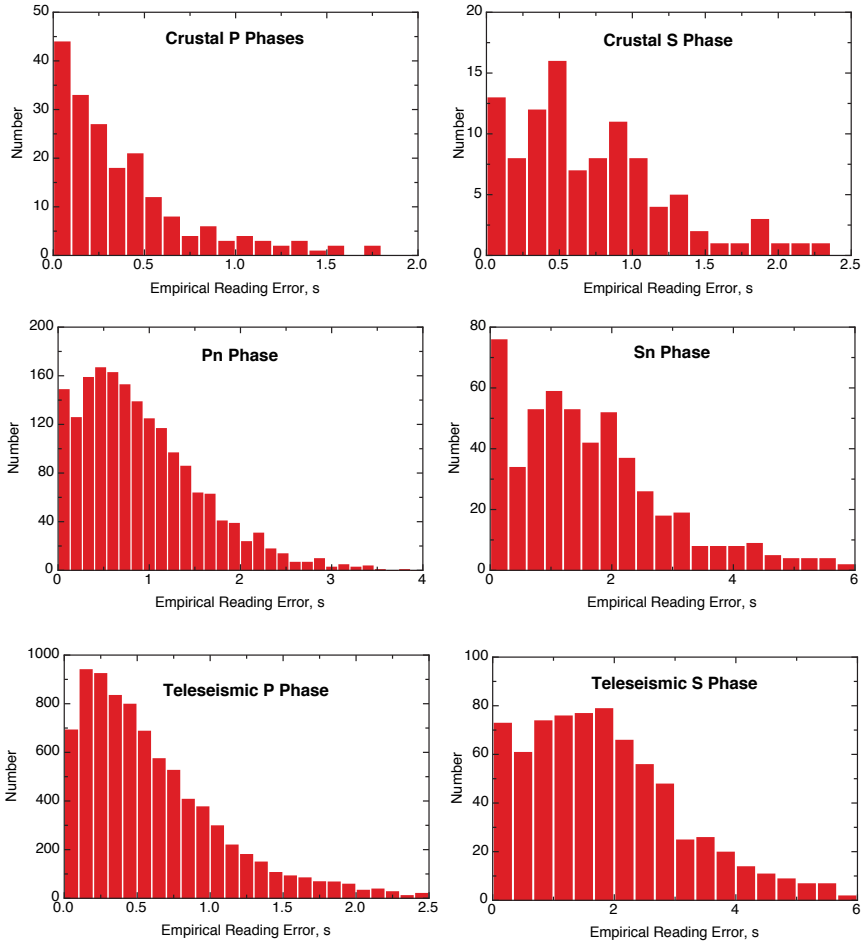

Figure 4. Histograms of Empirical Reading Errors

Calibration Levels

It has become common to use the GTX formulation ("Ground Truth to within X km") to describe the location accuracy of seismic events, although actual ground truth information is rarely available. As Bondár et al. (2004) noted, it is also necessary to specify a level of confidence for such a designation. In our study confidence ellipses are calculated at the 90% level. We always use the confidence level subscript when making a quantitative statement or when referring to a specific calibrated data set, but it may be dropped in a more general context.

Moreover, it is necessary to be clear about what metric is used for the GT classification. We take the nearest integer of the length of the semi-major axis of the 90% confidence ellipse that includes both the uncertainty of relative location and the uncertainty of the calibration process. In our experience it is possible to achieve GT_{90} levels of 2-3 km in the most favorable circumstances.

Depending on the purpose to which one might want to apply a set of reference events, the desired level of accuracy will vary. GT5 is the most commonly discussed level for calibrated earthquake locations, and GT5 (or better) locations certainly represent a significant improvement in accuracy over what is available in standard catalogs. In our calibration studies we have generally discarded events whose relative location within the cluster cannot be determined to better than 10 km (semi-major axis length).

Focal Depth

The HDC algorithm is poorly suited to including focal depth as a free parameter in relocation because it would require "connectivity" between all events with observations constraining depth, such as stations within 1-2 focal depths or teleseismic depth phases. If even one event in a cluster has no near-source readings or teleseismic depth phases that are shared with other events, the inversion for cluster vectors becomes singular.

For this reason most calibration location studies are performed with fixed focal depths and considerable effort goes into obtaining the best possible constraints on focal depth for these events. We make a careful review of observed depth phases and body waveform studies for this purpose. During relocation we often experiment with a range of depths and discover that there is strong sensitivity to changes of more than about 5 km in average assumed depth, relative to the preferred depth. In addition, when we have data at short epicentral distance for direct calibration, it is common to observe curvature of the travel-time vs. distance curve at short distances, which is diagnostic of source depth.

Our general philosophy has been to establish a default depth for each cluster, based on the best-constrained events in the cluster. Tests have shown that errors of less than about 15 km in depth have negligible effect on the estimated epicenter.

An Example

As an example of the application of the calibrated location procedures described above, we consider a cluster of earthquakes in Kyrgyzstan based mainly on a large (M_S 7.5), damaging earthquake on August 19, 1992, called the "Susamyr" event and it's aftershocks. Calibration is done with the indirect method, using the closely-controlled location and origin time of a large engineering explosion conducted in the region on December 22, 1999, the so-called "Kambarata" shot.

The cluster was formed by searching the ISC Bulletin for events in the vicinity of the Kambarata explosion, yielding 38 earthquakes between 1966 and 2008, 25 of which belong to the Susamyr sequence in 1992. The first relocation was done without the Kambarata explosion in the data set, locating the hypocentroid of the cluster by fitting 2628 teleseismic P readings (30°-90° epicentral distance) to the ak135 travel time model (Figure 5). Depths were fixed at the values reported in the ISC Bulletin, ranging from 0 to 70 km.

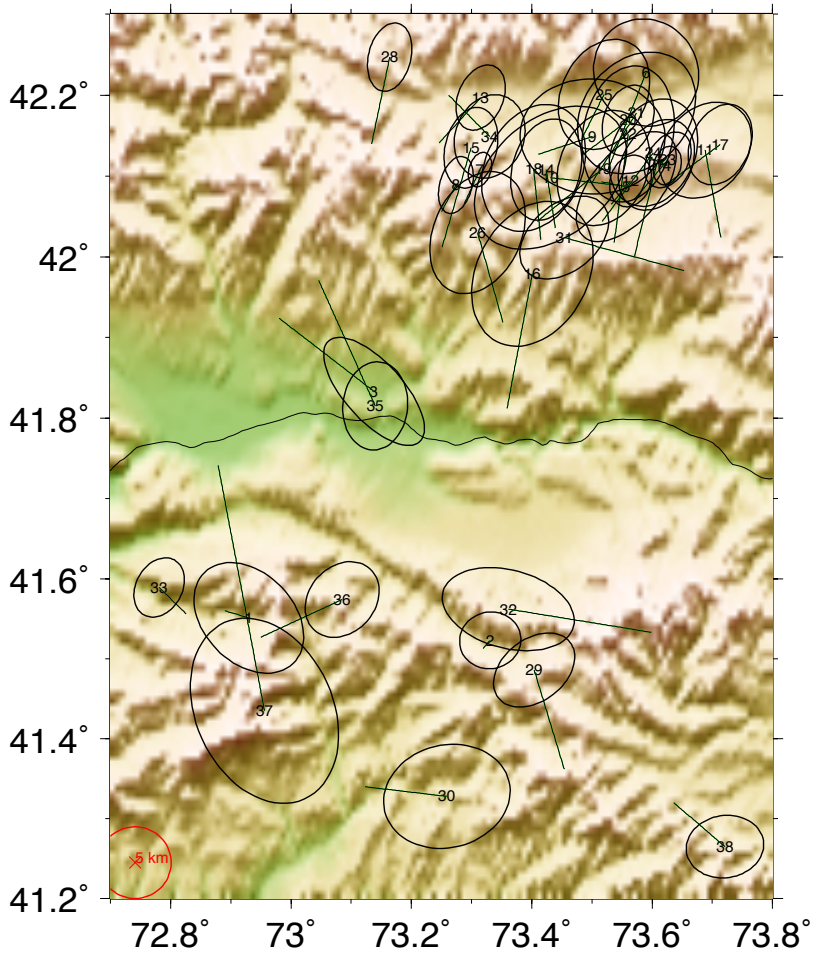

Figure 5. The Susamyr cluster initial relocation, without the calibration event, with default reading errors, and no cleaning of outliers. Black vectors show change in location from the location provided in the input data files (ISC locations). Confidence ellipses are at 90% level of confidence, for relative locations. The red circle (radius 5 km) is shown for scale.

The formal uncertainty of the location of the hypocentroid is 3.4 km (largest semi-axis of the 90% confidence ellipse), but there is an unknown level of bias in the location, which propagates to the location of each event, due to departures of the true Earth from the assumed velocity model. Our goal is to try to minimize this bias, and also to reduce the size of the confidence ellipses for relative location in Figure 5 by estimating empirical reading errors for each station-phase set and using that information to identify and remove outlier readings, and to weight the inversion. The uncertainties of the relative locations range from 2.4 to 13.4 km, measured as the largest semi-axis of the confidence ellipse.

For subsequent relocation runs, the events for which teleseismic depth phases were reported were reviewed and depths were fixed at values that satisfy the depth phase data. The remaining events

were held to a default depth for the cluster of 16 km that is based on the average depth of events for which there is depth constraint from teleseismic depth phases.

The cleaning process, removing the largest outliers and relocating, required 16 more runs of the relocation program. The calibrated locations are shown in Figure 6.

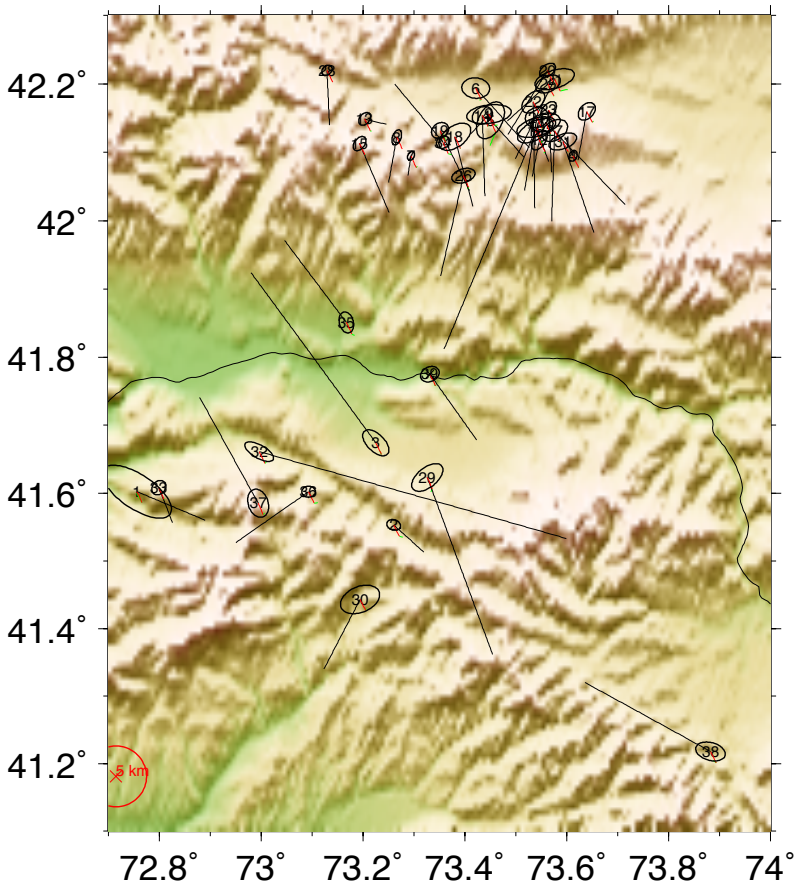

Figure 6. Final calibrated locations of the Susamyr cluster, after outlier removal and weighting according to empirical reading errors, with calibration based on the Kambarata explosion. Red line (barely visible) shows the small shift of the cluster required to fit the calibration event. Other features as in Figure 5.

In the case of the Susamyr cluster, there was very little location bias (2.1 km) for the cluster as a whole, because of the excellent azimuthal coverage by teleseismic P waves and the fact that the ak135 travel time model performs rather well in this region. Location bias of 10 km or more is not uncommon in many regions. Although the epicentral bias in this case is small, the shift in origin time for the cluster as a whole (-1.1 s) is significant for any application, such as tomography, that would make use of absolute travel times from this cluster.

Moreover, the changes to the locations of individual events in the cluster from their single event

locations are tens of km in many cases, and resolution of the pattern of aftershock seismicity for the 1992 Susamyr earthquake is dramatically improved by relocation. After relocation the uncertainty in absolute location ranges from 1.6 to 6.7 km. Because the absolute location of the cluster has been calibrated, these locations can be more usefully interpreted in terms of seismotectonics and seismic hazard.

The Susamyr cluster is typical in terms of the improvement in resolution of relative locations and the level of absolute location accuracy that can be obtained through the procedures described here.

Applications

The methods described here for developing calibrated clusters of earthquakes have been used in a number of studies, including:

Biggs, J., Bergman, E.A., Emmerson, B., Funning, G.J., Jackson, J.A., Parsons, B.E., and Wright, T.J., 2006, Fault identification for buried strike-slip earthquakes using InSAR: The 1994 and 2004 Al Hoceima, Morocco earthquakes: Geophysical Journal International, v. 166, p. 1347–1362.

Bondar, I., Bergman, E.A., Engdahl, E.R., Kohl, B., Kung, Y.-L., and McLaughlin, K., 2008, A hybrid multiple event location technique to obtain ground truth event locations: Geophysical Journal International, v. 175, no. 1, p. 185–201.

Bondar, I., Engdahl, E.R., Yang, X., Ghalib, H.A., Hofstetter, A., Kirichenko, V., Wagner, R., Gupta, I., Ekström, G., Bergman, E.A., Israelsson, H., and McLaughlin, K., 2004, Collection of a reference event set for regional and teleseismic location calibration: Bulletin of the Seismological Society of America, v. 94, no. 4, p. 1528–1545.

Nissen, E., Yamini-Fard, F., Tatar, M., Gholamzadeh, A., Bergman, E.A., Elliott, J.R., Jackson, J.A., and Parsons, B.E., 2010, The vertical separation of mainshock rupture and microseismicity at Qeshm island in the Zagros fold-and-thrust belt, Iran: Earth and Planetary Science Letters, v. 296, no. 3-4, p. 181–194.

Parsons, B.E., Wright, T.J., Rowe, P., Andrews, J., Jackson, J.A., Walker, R.T., Khatib, M., Talebian, M., Bergman, E.A., and Engdahl, E.R., 2006, The 1994 Sefidabeh (eastern Iran) earthquakes revisited: new evidence from satellite radar interferometry and carbonate dating about the growth of an active fold above a blind thrust fault: Geophysical Journal International, v. 164, p. 202–217.

Rastogi, B.K., Bergman, E.A., and Engdahl, E.R., 2005, Improved earthquake locations and estimation of Pn and Sn path anomalies for India, using multiple event relocation and reference events: Current Science, v. 88, no. 10, p. 1586–1591.

Ritzwoller, M.H., Shapiro, N.M., Levshin, A.L., Bergman, E.A., and Engdahl, E.R., 2003, Ability of a global three-dimensional model to locate regional events: Journal of Geophysical Research, v. 108, no. B7, p. 2353–2353.

Roustaei, M., Nissen, E., Abbassi, M.R., Ghorashi, M., Gholamzadeh, A., Tatar, M., Yamini-Fard, F., Bergman, E.A., Jackson, J.A., and Parsons, B.E., 2009, Vertical separation of surface folding, earthquake faulting, and aftershocks in the Zagros Simply Folded Belt (Iran): Geophysical Journal International, v. 142, p. 1–24.

Tatar, M., Jackson, J.A., Hatzfeld, D., and Bergman, E.A., 2007, The 2004 May 28 Baladeh earthquake (Mw 6.2) in the Alborz, Iran: Overthrusting the South Caspian Basin margin, partitioning of oblique convergence and the seismic hazard of Tehran: Geophysical Journal International, v. 170, p. 249–261.

Walker, R.T., Bergman, E.A., Szeliga, W., and Fielding, E.J., 2011, Insights into the 1968-1997 Dasht-e-Bayaz and Zirkuh earthquake sequences, eastern Iran, from calibrated relocations, InSAR and high-resolution satellite imagery: Geophysical Journal International,, p. no–no.

Walker, R.T., Bergman, E.A., Jackson, J.A., Ghorashi, M., and Talebian, M., 2005, The 2002 June 22 Changureh (Avaj) earthquake in Qazvin province, northwest Iran: epicentral location, source parameters, surface deformation and geomorphology: Geophysical Journal International, v. 160, p. 707–720.

References

Bondar, I., Myers, S.C., Engdahl, E.R., and Bergman, E.A., 2004, Epicentre accuracy based on seismic network criteria: Geophysical Journal International, v. 156, p. 483–496.

Croux, C., and Rousseeuw, P.J., 1992, Time-efficient algorithms for two highly robust estimators of scale: Computational Statistics, v. 1, p. 411–428.

Jordan, T.H., and Sverdrup, K.A., 1981, Teleseismic location techniques and their application to earthquake clusters in the South-Central Pacific: Bulletin of the Seismological Society of America, v. 71, no. 4, p. 1105–1130.

Technical Aspects of Installing and Running *mloc*

Last updated: January 28, 2018

The Code

mloc was originally written in Fortran 77 but nearly all code has been updated to use structures and features from Fortran 90/95 that improve legibility and robustness. There are multiple user-controlled levels of warnings for errors and debugging. The code is extensively commented, and a full understanding of the functionality of *mloc* requires at least occasional reference to the source code. Some rarely-needed options are only accessible by editing the code and re-compiling. The code is divided into multiple packages with related functionality. Most variables that are used beyond a given program unit are declared in a master include file (mloc.inc). This file also contains a large number of named common blocks and most communication between program units occurs through them.

Aside from the code base borrowed from Ken Creager's LOC program as a basis for *mloc*, the major bits of source code that have been adapted from other people's work are:

- Code from Johannes Schweitzer's HYPOSAT program for calculating travel times in the custom crustal models.
- Code from Ray Buland's software implementing the Tau-P travel time calculations.
- Code from Bob Engdahl's location program (i.e., EHB) for various odious tasks, such as ellipticity corrections and bounce-point corrections.

The code has been successfully compiled using the open source gfortran compiler, as well as commercial Fortran compilers from Absoft and Intel.

Utility Programs

Several types of utility programs are essential to efficient use of *mloc*. These include several programs for converting arrival time datasets to *mloc's* native data format (MNF), a program for searching MNF Bulletins (concatenations of event files), and several programs for editing MNF files. The source codes for an "essential" set of these programs are distributed with that of *mloc.*

Conversion Codes

- **isc_ims2mnf**: conversion to MNF from the version of IMS format returned by the bulletin search function of the ISC website.
- **seisan2mnf**: conversion to MNF from the "nordic" format produced by SEISAN.

MNF Bulletin

- **mnf_search**: search a bulletin in MNF format and extract events selected according to various criteria into a new bulletin or into individual event files (as used by *mloc*).

Editing

- **lres**: batch processing to flag a set of readings across multiple event files listed in one of the standard output files of *mloc*, according to the size of "cluster residuals" (normalized, demeaned travel-time residuals). Flagged readings will not be used in future runs.
- **rstat**: an interactive program to explore cluster residuals, an essential aid to manual editing (flagging) of outlier readings.
- **xdat**: batch process to flag readings with very large residuals, relative to other observations of that seismic phase.

Version History

There is a good bit to be learned about *mloc* by reviewing the history of changes, which is distributed along with the source code. Unfortunately, changes prior to late 2012 have not been recorded.

Operating System and Supporting Software

All recent development and most use of *mloc* has been done under OS X, but it has been successfully installed and used under several major distributions of Linux as well. I am not aware of any attempt to use *mloc* under Windows, and it unlikely to be feasible without substantial modifications to the code.

mloc makes extensive use of the Generic Mapping Tools (GMT) for plotting. Full support for plotting requires GMT v5. One of the standard output files is a '.kml' file which requires Google Earth for display.

Directory Structure for *mloc*

This section describes the recommended arrangement of directories and files for use with the multiple event relocation program *mloc* v10.2.8 and later. The following schematic shows the main directory structure:

mloc

clusters

documents

mloc_build

mloc_src

mloc_working

tables

crust

ellipticity

faults

gmt

cpt

dem

ETOPO

GINA

GLOBE

kml

spread

stn

tau-p

utlities

The paths that are underlined above are hard-wired in the main program (mloc.f90):

```
taup_path = 'tables/tau-p'
ellip_path = 'tables/ellipticity'
station_path = 'tables/stn'
cpt_path = 'tables/gmt/cpt'
dem_path = 'tables/gmt/dem'
psdre_path = 'tables/spread'
```

If you want to depart from the scheme shown above, you will need to edit mloc.f90 and recompile.

Top-Level Directories

The name and location of the top-level directory **mloc/** shown above is arbitrary. The names of the directories immediately below **mloc/** are also arbitrary. The role of each of these directories is described here:

clusters

The directories for individual earthquake clusters that will be analyzed using *mloc* are stored here. This directory can be located anywhere in the file system; it does not have to be under the top **mloc/** directory. It is convenient to have it nearby, however, because the directories that hold individual series of runs (see below) will be moved back and forth between here and the **mloc_working/** directory. Each cluster should be named after a geographic feature that exists within the cluster. Google Earth is a good tool for exploring possible names. Avoid using names that apply to a region larger than the cluster, (e.g., “zagros”), and choose a name that is fairly easy to type; you will be typing it frequently.

In practice, it is very common to have to make several (or many) series of runs on a cluster. The top directory for a set of such series can have a more descriptive name. Different series can be distinguished by significantly changed data sets (e.g., more or fewer events, new readings for some events) or application of a different relocation strategy. Each series would have its own directory under the main directory for the cluster, e.g.:

"Qeshm Island, Iran"

qeshm1

qeshm2

…

qeshm23

A specific run (relocation) within a series has its own index, e.g., qeshm23.1.

documents

This directory contains a variety of documentation about *mloc* and it's use. It can have any name you like and can be stored anywhere, but keeping it under the **mloc/** directory makes sense.

mloc_build

This directory is used by the makefile that controls compilation and building of the *mloc* executable. The makefile is stored here, and the object files for individual program units will be stored here also. The source code units to which the makefile refers are stored elsewhere, in the directory **mloc_src/**. The executable file will be created here and then moved to the **mloc_working/** directory for use.

If you have more than one FORTRAN compiler that you wish to use, you can create several of these build directories and name the build file after the compiler, e.g., mloc_gfortran and mloc_intel. In that case you can name the executables accordingly, "mloc_g", "mloc_i", etc.

mloc_src

Source code files are stored here. The makefile in **mloc_bld/** must have the correct pathnames to the source codes. Having only the source code files in this directory makes editing easier, and is essential if more than one compiler is to be used.

mloc_working

This is where most of the relocation analysis takes place. The executable *mloc* is stored here. The absolute pathname of this directory must be supplied to *mloc* by the configuration file (mloc.conf), which must exist in this directory. The cluster-series directory (e.g., **qeshm23/**) of the cluster to be relocated must be moved to this directory from its permanent home in the **clusters/** directory.

The other essential directory that must exist in this directory is the **tables/** directory and it's subdirectories. It contains a variety of data files used by *mloc*, organized in a number of

subdirectories, as shown in the schematic above. The contents of these directories are described below.

It is convenient, but not required, to create a directory **utilities/** in which to store the executables of several utility programs that are used regularly along with *mloc*:

lres: automatic flagging of readings with cluster residuals above a given threshhold

rstat: interactive investigation of cluster residuals for specific station-phases

xdat: automatic flagging of readings with gross outliers

To use these utilities, the executables are copied into the cluster-series directory, and then deleted when done, so it's convenient to have them close at hand.

tables/ and subdirectories

The **tables/** directory must exist within what I call the **mloc_working/** directory (or whatever you name it). Many pathnames in *mloc* are hardwired to look in the **tables/** directory for required data files.

crust

The **crust/** directory holds custom crustal models for use in *mloc* (command **lmod**). The argument to **lmod** is the pathname of the custom crustal model, relative to the *mloc* executable, so you may store crustal models in the **tables/crust/** directory or in the cluster-series directory or, indeed, anywhere you like. Therefore the **crust/** directory can be considered optional.

ellipticity

The **ellipticity/** directory holds a single data file (tau.table) that is used to calculated ellipticity corrections to theoretical travel times.

faults

The **faults/** directory holds digitized fault traces for plotting in GMT (command **fmap**). The argument to the **fmap** command is a relative pathname so the data file could be stored anywhere. The **faults/** directory can therefore be considered optional.

gmt

The **gmt/** directory contains two subdirectories, one (**cpt/**) to hold color palette tables and one (**dem/**) to hold digital elevation models for plotting topography. The **cpt/** directory should contain, at a minimum, the standard topo.cpt file that is distributed with GMT. If other color palette tables are available, a different one can be selected with the **cptf** command.

The **dem/** directory contains several subdirectories. Three of them are for the standard global DEMs distributed with *mloc*: GLOBE, GINA and ETOPO. The **dem1** command is used to select among these. If a high-resolution DEM is available for the particular source area of a cluster, it can be used for the baseplot and related plots that cover smaller areas, using the **dem2** command. The argument to **dem2** is a relative pathname, so it can be stored anywhere, within the **dem/**

directory (or within a custom/ subdirectory), in the cluster-series directory or elsewhere in the filesystem.

kml

The **kml/** directory holds graphic files of icons used in the .kml file that is created for each run.

spread

The **spread/** directory holds one or more data files that prescribe default reading errors for specific phases. These are usually over-ridden, after a few runs, by the values determined in the HD analysis, but in some cases it may be useful not to use the empirically-determined reading errors.

stn

The **stn/** directory holds data files describing the locations of stations. The most important such file is the master station file, which is often adequate by itself to provide coordinates for all necessary stations. The name of the master station file (master_stn.dat) is hardwired as a default in the code, but an alternative file can be specified in the configuration file (mloc.conf). *mloc* also allows the use of multiple supplemental station files (command **sstn**) and these files can be stored in the **stn/** directory, in the cluster-series directory or elsewhere. This is also the normal place to store a data file of stations that are suspected of reporting bogus depth phases (command **bdps**).

tau-p

The **tau-p/** directory holds data files needed by the tau-p software to calculate theoretical travel times. Two files, a '.hed' and a '.tbl' file are needed for each earth model. *mloc* normally uses the model ak135, but the tau-p files for other 1-D earth models are also available and can be selected with the **taup** command in *mloc*.

Configuration File for *mloc*

Beginning with *mloc* v9.9.0 (release date 9/30/2013), the specification of certain environmental variables was moved out of the code, into a configuration file that is read when the program is launched. Experience showed that some aspects of that design were flawed. With *mloc* v10.2.9, release date 12/16/2015, the configuration file has been modified in important ways, such that previous configuration files will need to be edited. This document describes the configuration file as of *mloc* v10.2.9 (release date 12/16/2015).

The biggest change is that the configuration file now uses keywords to define the nature of each line, rather than requiring a specific number and order of lines. All elements have default values defined in the code itself, so it is not necessary to set all possible keywords if the default values are correct for your installation of *mloc*.

Filename and Path of the Configuration File

The name of the configuration file is ‘mloc.conf’ and it must be found in the same directory as the executable *mloc*.

Keywords

The currently-defined keywords are listed here. The keyword starts in column 1 of the configuration file and is always followed by a colon and a blank, before the argument.

WORKING_DIR:

Defines “mloc_path”: The full pathname of the directory in which the *mloc* executable and the configuration file reside. I refer to this directory as the “working” directory. Maximum 100 characters. The default value is ‘ ’, which will almost never be correct, so all configuration files should carry a line with the ‘WORKING_DIR: ’ keyword.

AUTHOR:

Defines “mloc_author”: a code for the person running *mloc*. This code will be written into the **.datf** file as the author of the location (“origin”) in the hypocenter record (MNF v1.3 format), and it will show up in the **.summary** file and perhaps some other output files. Maximum 8 characters. The default value is ‘**default**’; it is highly recommended that all users of *mloc* set an author code for themselves.

STATION_MASTER:

Defines the name of the file that is to be used as the master station file. This is only the name of the file, not the path. The master station file is expected to be found in the **tables/stn** subdirectory under the *mloc* working directory. If necessary, the path can be changed in mloc.f90, and the program will need to be re-compiled. The default value of the master station file is ‘**master_stn.dat**’, the master station file normally distributed with *mloc*. Therefore, in a standard installation there is no need to use this keyword.

GMT_VER:

Defines the version of GMT that will be used for plots. The only acceptable arguments are ‘4’ or ‘5’. GMT4 is deprecated, and recent development of the plotting capabilities, such as the empirical path anomaly plots (command ‘**epap**’) are available only under GMT5. The default is GMT5, so it is only necessary to use this keyword if you are still using GMT4.

SHELL:

Defines the operating system shell to be used in formulating GMT scripts. Supported arguments are ‘**csh**’ and ‘**bash**’. The default is ‘**bash**’. The command ‘**shel**’, which formerly was used for this purpose, has been removed.

A Sample Configuration File

The mloc.conf file that I use is distributed with *mloc* but it must be customized for each new installation. My configuration file consists of only two lines:

```
WORKING_DIR: /Users/eab/Documents/Seismology/Projects/Software/mloc/
mloc_working
AUTHOR: EBergman
```

All other values (master station file, GMT version and shell) that could be set in the configuration file are satisfied by the defaults for my installation.

mloc Command Files

Last updated: June 9, 2013

This document describes the basic structure of command files for use with the multiple event relocation program *mloc*.

Background

Control over the processing of mloc is managed by commands, of which there are about 70. Only a small number of commands are always required for a run. Commands may be entered interactively or read from a text file containing the commands and arguments. In practice we use both approaches, starting with a basic command file that handles commands that are mainly repeated from run to run, and issuing a small number of commands interactively to launch the run. Indeed, a command file cannot be read except via an interactive command (cfil). It is common to edit the command file slightly between runs as well, but most of the information (notably the event definitions) in a command file stays constant from run to run.

Command files can be given any name, but standard practice is to make a new command file for each run and name it after the run ID, e.g., "cluster1.2.cfil". The filename suffix "cfil" is standard, but it is not required.

Processing a Command File

When mloc is launched there are several interactive steps that occur first:

- Giving a name for the run
- Giving the name of the subdirectory where the related files are stored
- Specifying if data flags are to be honored

After this, the program lists all the available commands and asks for interactive command input. The normal procedure is to tell mloc to process the command file that has been prepared for this run:

```
cfil cluster1.2.cfil
```

During processing of the command file (in the main program mloc.f90) the user may be asked for some additional input, or warning messages may be issued. The processing of commands is handled by the subroutines in mloc_commands.f90. When the command file is finished the program asks for more interactive input. The relocation does not begin until the user issues the "run" command.

Command File Structure

Command files consist of two primary sections. The first section contains commands that control the procedures that will be used for the run, affecting all events, but it does not define the actual events that will be relocated. The second section defines the events and allows the user to issue

commands which are event-specific. Some commands can be issued in either section and have a different action, depending on whether they are issued before any events have been defined.

Defaults

It is possible to make a run of mloc with no commands except the ones that define events, because all parameters that control how the relocation is done have defaults. The defaults are defined in the main program mloc.f90. In this case the relocation would be done using teleseismic P arrivals and the ak135 travel time model. Data would be weighted inversely to their uncertainty and phase re-identification would be done. All four hypocentral parameters would be free. Only a basic output plot would be made, as well as the core output files.

Defining Events

Events are defined with a minimum of three commands, for example:

```
memb

even 20090807.1859.26

inpu 20090807.1859.26.mnf
```

The “memb” (MEMBer) command starts the definition of a new event. The “even” (EVENt) command provides an identifying name for the event that will appear in various parts of the output files. It is normally derived from the filename of the datafile itself, which is supplied in the “inpu” (INPUt) command. Additional commands could be given and they would apply only to the current event until another “memb” command is encountered.

Except for the very smallest clusters, assembly of the event definition section would be very tedious to do by hand, so it is normally done by one of several utilities, especially ones that search through a concatenated set of event files (i.e., a bulletin) and extract individual events for a cluster.

Terminating a Command File

No special steps are required to terminate a command file. It normally ends with the definition of the last event.

MNF Native Format for *mloc*

This section describes the format of data files (arrival time data) that can be read or written by *mloc*. There are several distinct formats, but all are considered to belong to the family of "*mloc* native format" or "MNF"; they are distinguished by version number family:

- v1.3: Format for input of individual event files or output of bulletins of event arrival time data.
- v1.4: Format for output of a specialized format for an entire cluster, designed originally for import to the NEIC ComCat database.
- v1.5: Format for input of differential time data for a cluster.

MNF v1.3

This document describes the native data format for use in the multiple event relocation program *mloc*. Data files with this format have the filename suffix ".mnf" and the format will be referred to as MNF for "*mloc* native format". This document describes v1.3.3 of the MNF format, a minor change from v1.3.2, adding a record type to explicitly carry event ID information. The first version of *mloc* to fully support this file format is v10.3.4 with a release date of October 11, 2017, but earlier versions of *mloc* should be able to process event files in this format. A warning would be issued on encountering the unknown record type.

The history of changes to this format is listed at the end of this section.

Background

In its early development in the late 1980s, *mloc* used a data format inherited from Ken Creager's LOC program, on which *mloc* was based. In the late 1990s, when *mloc* began to be heavily developed for application in research on calibrated ("ground truth") locations in close collaboration with E. R. Engdahl, the ISC's 96-byte fixed format (FFB), on which Engdahl had standardized his processing, was adopted as the default data format for *mloc*. The FFB format had to be extended in certain ways to meet the needs of *mloc* (as well as Engdahl's codes), and it was always an awkward format, particularly in the fact that very different formats are used for initial and secondary phase readings, on the need to break up station codes longer than 4 characters, and in the strict requirements on ordering of records that is imposed by the concept of carrying in each record the record type for the following record.

At this time, as support for FFB is being discontinued by the ISC, it has become necessary and desirable to devise a format that specifically meets the requirements of *mloc* and which is more compatible with current practice in seismological data formats and exchange mechanisms. This results in data files that are reasonably compact and easy to process and edit as required by the typical processing strategies that have proven to be effective with *mloc*. Data files with this format have the filename suffix ".mnf" and the format will be referred to as MNF for "*mloc* native format".

Starting with version 9.8.0, *mloc* supports only MNF format. Starting with version 10.0.5, *mloc* drops support for versions of MNF format before v1.3.

Description and Features

MNF is a fixed format based on the common concept of a small set of distinct record types with different formats, identified by a single-character flag in the first column. Each record is a single line; each record type has a minimum line length determined by the defined fields (even if they are not all required). There are minimal constraints on the order in which different record types may occur. Each type of information (e.g., event, hypocenter, depth, magnitude, phase arrival) has a specific record type, and there are a few utility record types. The format can be used to present data for a single event, an earthquake catalog (multiple events, hypocenters only) or a seismic bulletin (multiple events, including phase readings). Events with phase data can be intermixed with events lacking phase data. The **bulletin record** "B" is expected to be the first record of all multi-event data files and it should not appear in single-event files. The currently-defined record types, their flags, their minimum (i.e., required fields) and full (i.e., defined fields) line lengths and their normal order of appearance are given in the following table:

Flag	Record Type	Minimum Length	Full Length
B	Bulletin	1	121
F	Format version	15	15
E	Event	1	121
I	ID	1	51
H	Hypocenter	74	121
D	Depth	9	121
M	Magnitude	8	121
P	Phase reading	55	121
#	Comment	1	121
S	Stop event	1	4
EOF	End of file	3	3

Unlike all other record types, which are distinguished by the flag in column 1, the **end-of-file record** ("EOF") uses columns 1-3; it has no other arguments. It causes processing of a data file to end, so it would normally only be found once, at the end of the file, whether it holds a single event or multiple events.

The 'natural' line length for MNF files is 121 characters, because most of the information-carrying record types have fields defined to this length.

Data for a single event is carried in a block of records that must start with an **event record** "E" and end with a **stop record** "S". Within the block, data is carried in a combination of **hypocenter** "H", **depth** "D", **magnitude** "M", and **phase reading** "P" records. At least one **hypocenter record** "H" is required, but multiple estimates of hypocenter are permitted. **ID**, **magnitude** and **depth records** are optional and there is no limit to how many can be supplied of each. **Phase**

reading records are also optional, in the case of an earthquake catalog. The MNF format can represent a catalog but it is not designed to do so in an efficient manner, since it requires a minimum of three lines (an **event record**, at least one **hypocenter record**, and a **stop record**) for each event.

Some of the fields in an MNF-formatted dataset are not required for *mloc* processing, but they carry information which is often important to retain for interpretation of results and for maintaining compatibility of data products with the NEIC and ISC and other agencies that have standardized on formats such as the IASPEI ISF format for data exchange. Conversely, MNF does not carry some (actually, many) fields which would be considered essential for a general-purpose seismic bulletin format such as ISF, because it is optimized for use in relocation studies using *mloc*.

Several record types carry a "usage" flag (always in column 3) that determines the way (or if) the information in that record will be used. Some records have an "ID" field to carry an ID number that may have been assigned elsewhere, typically in a relational database (e.g., EvID, OrID, ArrID). Event IDs have their own record type. **Comment records** (flag "#") are supported; they can be inserted anywhere in an MNF-formatted file. Obviously, any text that occurs after the first **EOF** record will not be processed and can be considered as a comment, regardless of the formatting, but the use of the **EOF** record in this manner is not recommended.

Except for the requirement to start each event block with an **event record** and end the block with a **stop record**, there are few requirements on the order of records within an event block. In most cases the recommended order would be **event record**, **id record(s)**, **hypocenter record(s)**, **depth record(s)**, **magnitude record(s)**, **phase reading records**, followed by a **stop** record**.**

There must be at least one **hypocenter record**, but multiple hypocenter estimates can be carried. The usage flag should be used to determine which of several **hypocenter records** will be honored for the starting location in *mloc* (or is otherwise the "preferred" location); otherwise precedence will be determined by the software reading the file, probably either the first **hypocenter record** encountered or the last. The same principle applies to **depth** and **magnitude** records: the usage flag should be used to specify a preferred value, if there is one, rather than relying on the sequence.

Format Versions

Each data file, whether for a single event or multiple events, should contain a **format version record** ("F") before the first **event record** ("E"). The **format version record** provides a version number for the MNF format in which the file is written. A program that reads an MNF file should check the version number to be sure it will correctly interpret the data records. A bulletin or catalog made up of event files written in different MNF format versions could be processed by including the necessary **format version records** ("F") when the format changes for the next event, but this is not recommended.

Starting with MNF v1.3.3 and *mloc* v10.3.4 (October 11, 2017 release), format versions are expected to be complete, i.e., "1.3.3" rather than "1.3" which was adequate for previous versions of *mloc*.

Depths

The **hypocenter record** normally (but not always) carries an estimate of focal depth, but there may be additional estimates of depth that should be carried. The **depth record** is intended to carry information on depth that is considered credible. The most common source of such depth estimates is waveform analysis of some kind. Multiple depth records are permitted and one of them can be designated as preferred by the usage flag '=' in column 3. Depth records can carry an optional depth code flag which *mloc* uses to keep track of the nature of depth constraint. It is highly recommended to use the standard flags (defined below). For example, the focal depth in the preferred **hypocenter record** will be taken as the preferred depth by default in *mloc*, but that will be over-ridden if a following **depth record** meets these requirements:

- The depth record has been designated as preferred by the usage flag "=" in column 3
- The depth record carries a depth code that is considered "constrained"
- The hypocenter record's depth estimate does not carry a depth code that is considered to be "constrained"

Like the **hypocenter record**, a **depth record** can carry an asymmetric estimate of uncertainty. The depth uncertainties are optional. Focal depth and both uncertainties can be given to a tenth of a kilometer, but the decimal point should be present even if the value is only given to the nearest kilometer, to ensure correct reading by Fortran formatted read statements. The authorship field can be used for comments about the nature of the depth estimate. There is no concept of authorID for depth.

Magnitudes

Magnitude estimates are carried in a **magnitude record**, one magnitude estimate per record. Multiple magnitude estimates can be carried. A usage flag can be used to select a preferred estimate from among multiple records, but is not required. If no magnitude record is marked as preferred, *mloc* uses the first one encountered as a measure of magnitude. In cases where it is desired to carry a magnitude associated with a specific reading, a **magnitude record** could be interspersed with **phase reading records**; in any case the author field could be used to indicate the desired association. Magnitudes can be carried to two decimal places. In any case the decimal point should always be given, to ensure correct reading by Fortran formatted read statements.

Station Codes

The concept of station codes is evolving away from the traditional strategy of attempting to carry all necessary information about a seismic station (plus the installed instrumentation and who operates it) in a single code of 4 or 5 characters. The MNF format implements the New IASPEI Station Coding Standard:

or “ADSLC” formulation, using the fixed format display standard described in the defining documentation <http://www.isc.ac.uk/registries/download/IR_implementation.pdf>. In this format, what used to be known as the “station code” is carried in 3-5 characters. In *mloc*, the variable that holds the station code is declared with 6 characters, so the 6th character is available for specialized purposes. The main such purpose is to resolve station naming conflicts, in which data may be obtained from a station which has not been registered for international data exchange at the ISC, and furthermore the code selected by the station operator conflicts with a code that has been registered for a different station. This causes a problem if both stations appear in a data set. For this reason, the MNF format has two fields for the station code: once as a 6 character field that is read by *mloc*, and optionally, as a 5-character subfield of the ADSLC formulation. This latter instance of the station code is not used by *mloc*, but it’s useful to carry in the MNF format for forensic purposes.

Phase Names

Phase name is carried twice in the MNF format. The first instance (columns 24:31 of a “P” record, see below) is read by *mloc*, and then it is subject to the various procedures within *mloc* that may change the syntax of the phase name or change the phase ID completely. The second instance (columns 66:73) carries the phase name as reported by the original source. It is sometimes useful to change the input phase name from the original phase name, to assist *mloc’s* phase identification algorithm in determining the correct (or at least the desired) phase identification. It is also useful to retain the original phase name unchanged for reference. The MNF format also carries a position for a special flag (“!”) which informs *mloc* that the input phase name should not be changed during relocation.

ID Numbers

For informational and forensic purposes, MNF includes fields for identification numbers for various kinds of data that are provided by seismological centers. The only use that *mloc* makes of such information is to use event IDs (evids), if provided, to correctly match events with the relocation results of a previous run, as carried in an hdf-formatted file.

Conventionally, ID numbers assigned to events (“evid”), hypocenters (“orid”), phase readings (“arrid”) and other kinds of data in relational databases are integer numbers. Standards on how many digits are carried vary from system to system, but 10-digit integers are likely to be necessary before long, especially for arrids. The MNF format does not enforce a data type on those IDs; they are character fields as far as *mloc* is concerned. Even so, arrids and orrids should be right-justifed to aid in correct reading of integers by a Fortran code. Although 10-character arrids are specified in MNF it has proven necessary to provide larger fields for evids and orids.

The larger field for evids is driven by NEIC, where current practice places no practical limit on the length of an event ID, which is no longer an integer, but a combination of letters (e.g. network codes) and digits. The 40-character field provided in the MNF format should be adequate to handle these, but *mloc* only reads the first 10 characters from this field, because this

is assumed to be adequate to distinguish between events in a cluster. An evid of less than 10 characters can be placed anywhere in the 10-character field (columns 12:21) that *mloc* reads; it will be right-justified inside *mloc*.

Prior to v1.3.3 of the MNF format, only a single evid could be entered for each event; it was carried at the end of the event record. In v1.3.3, evids are read from one or more **id records**. The format also includes a character field to identify the source of the evid. The current version of mloc can still read an older MNF file in "1.3" format and extract an evid carried in an **event record**. The first-encountered **id record** will be the preferred value in *mloc* by default, or the usage code ("=") can be used to specify among multiple records.

The *orid* field in a **hypocenter record** (columns 104:121) is 18 characters in length because it is used by *mloc* to record the cluster name/series code (left-justified, starting in column 104). For **magnitude** and **phase** records, the associated 10-character ID field is in columns 112:121. Although 10 digit integer IDs can be carried in the ID field, it should be noted that 32-bit computer systems may have problems processing integers of more than 9 digits.

Defined Record Types

The concept of "optional" fields in the descriptions of record types is specifically in the context of use by *mloc*. Fields that are required by *mloc* are printed in bold below. In writing MNF-formatted data files it is advisable to pad lines to the full length of that record type, which is based on the defined fields, not the required fields, and it is not unwise to pad all lines to 121 characters, the full length of the longest defined record types, regardless of record type.

Bulletin Record ("B" in column 1)

Column	Description
1	**Line format flag "B"**
5:121	Bulletin description, optional (a117)

Format Version Record ("F" in column 1)

Column	Description
1	**Line format flag "F"**
10-15	**Format version (a6)**

Note: It is useful for readability to include extra text, such that columns 1:9 read "F MNF v", but all that is required is the "F" in column 1 and the version number in columns 10-15.

Event Record ("E" in column 1)

Column	Description
1	**Line format flag "E"**
3	Usage flag, optional (a1)
5:121	Annotation, optional (a117)

Note: A usage flag of "-" indicates that there is no phase data available for this event.

ID Record ("I" in column 1)

Column	Description
1	**Line format flag "I"**
3	Usage flag, optional (a1)
5:10	ID source, optional (a6)
12:51	Event ID, optional (a40)

The only recognized usage code is "=", which makes this entry preferred, regardless of its position relative to other ID records. Otherwise the first **id record** encountered will be used. An **id record** in which the Event ID field is left blank is legal. If there are multiple **id records** and it is desired to have *mloc* ignore them, an empty **id record** with usage code set to make it the preferred ID could be used.

Hypocenter Record ("H" in column 1)

Column	Description
1	**Line format flag "H"**
3	Usage flag, optional
5-8	**Year (i4)**
10:11	**Month (i2)**
13:14	**Day (i2)**
16:17	**Hour (i2)**
19:20	**Minute (i2)**
22:26	**Seconds (f5.2)**
28:32	OT uncertainty, optional (f5.2)
35:42	**Latitude (f8.4)**
44:52	**Longitude (f9.4)**
54:56	S_{min} azimuth, optional (i3)
58:62	Error ellipse S_{min}, optional (f5.2)
64:68	Error ellipse S_{maj}, optional (f5.2)
70:74	Focal Depth, optional (f5.1)
76:76	Depth code, optional (a1)
78:82	Plus depth uncertainty, optional (f5.1)
84:88	Minus depth uncertainty, optional (f5.1)
90:93	GTCU, optional (a4)
95:102	Author, optional (a8)
104:121	Origin or cluster ID, optional (a18)

Note: *mloc* recognizes a hypocenter record in which the usage flag is "=" as the preferred hypocenter for setting the starting location. If no hypocenter record carries the usage flag, the first hypocenter record encountered will be taken as preferred by *mloc*. Other software may behave differently.

Both depth uncertainties are provided as positive numbers. "Plus" depth uncertainty is on the deeper side; "Minus" uncertainty is shallower, and therefore should not be greater than the focal depth in absolute value. If only one depth uncertainty is encountered, it should be interpreted as a symmetric uncertainty. It is important to put the decimal place into the depth field, even if precision is only to the nearest kilometer (or more), to ensure correct reading by a Fortran formatted read statement.

No information on the statistical level of the uncertainties of origin time, depth, or epicenter is provided in the MNF format because there is so little standardization at present. Such values are commonly interpreted as ± 1 σ for origin time and depth, but confidence ellipses are usually calculated at 90% or 95% confidence levels.

The "GTCU" field carries a four-character code relating to calibration status (I prefer this term to "ground truth" level). It could be the GTX formulation (e.g., Bondar et al., (2004)), but I have developed the GTCU nomenclature to provide much more detailed information on the subject of what hypocentral parameters are considered to be calibrated (i.e., thought to be bias-free). The GTCU nomenclature is documented fully elsewhere.

The placement of a traditional (integer) "orid" value within the 18-character field is optional, but it is probably best to right-justify it (i.e., in columns 112:121) to facilitate reading as an integer. *mloc* writes the cluster name and series number as a character string that is left-justified in the field, beginning at column 104.

Depth Record ("D" in column 1)

Column	Description
1	**Line format flag "D"**
3	Usage flag, optional
5:9	**Depth (f5.1)**
11:11	Depth code, optional (a1)
13:17	Plus depth uncertainty, optional (f5.1)
19:23	Minus depth uncertainty, optional (f5.1)
25:121	Authorship and comments, optional (a97)

Note: Both depth uncertainties are provided as positive numbers. "Plus" depth uncertainty is on the deeper side; "Minus" uncertainty is shallower, and therefore should not be greater than the focal depth in absolute value. If only one depth uncertainty is encountered, it should be interpreted as a symmetric uncertainty. It is important to put the decimal place into the depth field, even if precision is only to the nearest kilometer (or more), to ensure correct reading by a

Fortran formatted read statement. The “depth code” in column 11 is an optional character flag that informs *mloc* about the nature of the depth constraint. Standard values are:

Depth Code	Meaning
c	Cluster default depth
d	Teleseismic depth phases
e	Engineered (man-made explosion)
f	Fault model (InSAR, GPS, etc)
l	Local-distance readings
m	*mloc* solution with free depth
n	Near-source station readings
r	Relocation outside *mloc* with free depth
u	Unknown
w	Waveform analysis

Although the use of depth codes is optional (blank is processed without problems), it is recommended to use them whenever possible, because certain procedures in *mloc* make use of these codes, for example, in determining the starting depth for an event.

The depth code “l” indicates a constraint on focal depth from readings at local distance (but still direct, crustal P and S phases), beyond the distance range normally considered appropriate for depth control (for which the “n” flag is provided). This is an advanced topic that requires careful analysis in *mloc*.

Magnitude Record ("M" in column 1)

Column	Description
1	**Line format flag “M”**
3	Usage flag, optional (a1)
5:8	**Magnitude (f4.2)**
10:14	Magnitude scale, optional (a5)
16:110	Author and comments, optional (a95)
112:121	Magnitude ID, optional (a10)

Note: *mloc* recognizes a magnitude record in which the usage flag is “=” as the preferred magnitude for this event. *mloc* uses only the first two characters of magnitude type. Magnitude ID would normally be the orid from a **hypocenter record**; it should be right-justified in the field.

Phase Reading Record ("P" in column 1)

Column	Description
1	**Line format flag “P”**
3	Usage flag, optional (a1)

5:10	**Station code (a6)**
12:17	Epicentral distance, optional (f6.2)
19:21	Azimuth, event to station, optional (i3)
23:23	Prevent phase re-identification flag, optional (“!”)
24:31	Input phase name, optional (a8)
33:36	**Arrival time year (i4)**
38:39	**Arrival time month (i2)**
41:42	**Arrival time day (i2)**
44:45	**Arrival time hour (i2)**
47:48	**Arrival time minute (i2)**
50:55	**Arrival time seconds (f6.3)**
57:58	Reading Precision, optional (i2)
60:64	TT residual, optional (f5.1)
66:73	Original phase name, optional (a8)
75:79	Agency, optional (a5)
81:88	Deployment or network, optional (a8)
90:94	Station, optional (a5)
96:97	Location, optional (a2)
99:101	Channel, optional (a3)
103:110	Author, optional (a8)
112:121	Arrival ID, optional (a10)

Note: It is helpful, but not required, to place a “dot” between the ADSLC fields. The usage flag carries the variable “fcode” in *mloc*. Common values of fcode are:

Value	Meaning
x	outlier, do not use
d	duplicate reading, do not use
m	missing station coordinates, do not use
p	phase that is unknown or not used
s	reading that is being skipped

Readings marked with ‘s’ are skipped according to an explicit command (SKIP command) that has been given in *mloc* to specify a combination of station-phase-authorship to be skipped. Wildcards are supported so that, for example, all instances of a certain station can be skipped, or only specific phase readings by a specific author.

Comment Record ("#" in column 1)

Column	Description
1	**Line format flag “#”**

2:121	Comment, optional (a120)

Stop Record ("S" in column 1)

Column	Description
1	**Line format flag “S”**

Note: Only the “S” in column 1 is required, but for better readability it is useful to write “STOP” in columns 1:4.

End of File Record ("EOF" in columns 1-3)

Column	Description
1:3	**Line format flag “EOF”**

Changes from Previous Versions

Changes from v1.3.2

- A new record type is defined, the **ID record** (for event ID or evid), denoted by “I” in column 1. It became desirable to be able to carry more than one event ID, since there is no universal registry. The event ID field has been removed from **event records**.
- The version field of format record is now expected to have the full version number (e.g., “1.3.3”) rather than the abbreviated “1.3” used previously.

Changes from v1.3.1

- The field for event ID (evid) in the **event record** has been expanded from 10 characters to 40 characters, to accommodate current practice at NEIC which features no limit on the length of event identifiers. The content of the field is still right-justified in column 121. The length of the event annotation field is reduced correspondingly.

Changes from v1.3

- Focal depth is no longer a required field in the **hypocenter record**. There are cases, e.g., macroseismic locations, for which no estimate of depth is provided (technically, it becomes an “epicenter” record.

Changes from v1.2

- Moved the event ID field to the end of the **event record**, and left the interior of the record for use as an ‘annotation’ so it can carry the names that are commonly assigned to large or significant earthquakes.
- Added a single-character depth code field to the **hypocenter record** and adjusted positions of the depth uncertainty fields.
- Added the **depth record**, denoted by “D” in column 1.

- Changed the format of the **magnitude record** to allow a larger field for author and comments, in order to accommodate reference-style authorship, e.g., “Baker et al. (1993)”. Put the magnitude field in front of the magnitude type field.

Changes from v1.1

- The **hypocenter record** format has been changed to accommodate asymmetric uncertainties in focal depth.
- A field for GTCU (a new nomenclature I designed to carry information about calibrated events—it supersedes the old GTX formulation) is added to the **hypocenter record** (a4).
- The length of the field for cluster ID in a **hypocenter record** is expanded (first to 20 characters, then back to 18), to accommodate *mloc* cluster names (and series numbers).

MNF v1.4

This document describes a variant of the the native data format used in the multiple event relocation program *mloc*. Data files with the native format have the filename suffix ".mnf" and the format will be referred to as MNF for “*mloc* native format”.

Version 1.4 of MNF is a special-purpose variant of what would be considered the “standard” version, v1.3.1. It was created to address a specific problem, the creation of an output file for import into the Global Calibrated Earthquake Cluster (GCEC) catalog in the USGS COMCAT catalog server. This version of MNF should only be found in “.comcat” output files created by the “ccat” command in *mloc*. Version 1.4 of MNF contains several new record types and the formatting of some record types is different from that used in v1.3.1 of MNF.

Version 1.4.2 of MNF makes a significant change to the format of phase records, adding a field for a deployment/network code.

Background

See the documentation of MNF v1.3.1 for a full discussion. Version 1.4 of MNF is only supported in versions of *mloc* from 10.0.7, and only as an output format for “comcat” files, which is controlled by the “ccat” command. Version 1.41 is first supported in v10.2.6 of *mloc*, released on July 5, 2015.

Description and Features

MNF is a fixed format based on the common concept of a small set of distinct record types with different formats, identified by a character flag in the first column. Each record is a single line. Each type of information (e.g., event, hypocenter, magnitude, phase arrival) has a specific record type, and there are a few utility record types. The currently-defined record types and their flags are given in the following table:

Flag	Record Type
B	Bulletin
F	Format version

E	Event
H	Hypocenter
M	Magnitude
P	Phase reading
#	Comment
C	Station Coordinates
S	Stop event
L	Layer velocity
EOF	End of file

Unlike all other record types, which are distinguished by the flag in column 1, the **end-of-file record** ("EOF") uses columns 1-3; it has no other arguments. It should only be found once, at the end of the .comcat file.

An MNF v1.4 file (or "comcat" file) always starts with a **bulletin record** "B", and it will carry a descriptive comment if it was created by *mloc*. The **bulletin record** is always followed by a **format record** "F". If there is commentary describing the most important features of the earthquake cluster and its relocation, which is highly recommended, it will follow the **format record** "F" as a series of **comment records** "#". If a custom crustal velocity model has been used this section will be followed by a series of **layer velocity records** "L". Otherwise it should be assumed that the ak135 travel-time model was used for all phases. If any station coordinates have been taken from sources other than the NEIC metadata server or the ISC Station Registry, there will next be a series of **station coordinate records** "C". This constitutes the header block of a comcat file. This is followed by a set of event blocks. The comcat file is terminated by an **end of file record** "EOF".

Data for a single event is carried in a block of records that must start with an **event record** "E" and end with a **stop event record** "S". Within the block, data is carried in a combination of **hypocenter** "H", **magnitude** "M", and **phase reading** "P" records. Only one **hypocenter record** "H" is permitted. **Magnitude records** are optional and there is no limit to how many can be supplied.

Magnitudes

Magnitude estimates are carried in a **magnitude record**, one magnitude estimate per record. Multiple magnitude estimates can be carried. Magnitudes can be carried to two decimal places.

Optional Fields

All fields defined below will normally be present in a comcat file written by *mloc*. Under certain circumstances the following record types might be absent:

- Comment records, if no commentary text has been provided.
- Layer velocity records, if ak135 was used for all travel-time calculations.

- Magnitude records, if an event has no magnitude estimates.

Defined Record Types

Bulletin Record ("B" in column 1)

Column	Description
1:1	**Line format flag "B"**
5:121	**Bulletin description (a117)**

Format Version Record ("F" in column 1)

Column	Description
1:1	**Line format flag "F"**
5:9	**Format version (a5)**

Comment Record ("#" in column 1)

Column	Description
1:1	**Line format flag "#"**
2:121	**Comment, optional (a120)**

Layer Velocity Record ("L" in column 1)

Column	**Description**
1:1	**Line format flag "L"**
8:14	**Depth or layer thickness (f7.3)**
20:24	**Vp (f5.3)**
30:34	**Vs (f5.3)**

Note: *mloc* writes the crustal velocity model with two values for each layer, the depth of the upper and lower interface. Internal interface depths are therefore repeated, giving velocities above and below. This accommodates a model in which a layer can have a linear gradient in velocity. Alternatively, a flat-layered model could be defined by with one line per layer, in which the first parameter is interpreted as layer thickness. Depth/layer thickness is given in km. Velocities are given in km/s.

Station Coordinates Record ("C" in column 1)

Column	**Description**
1:1	**Line format flag "C"**
3:8	**Station (a6)**
10:14	**Agency (a5)**
16:23	**Deployment (a8)**
25:32	**Latitude (f8.4)**

34:42	**Longitude (f9.4)**
44:49	**Elevation (i6)**

Note: Station elevation is the elevation of the instrument, given in meters, relative to mean sea level (positive or negative).

Event Record ("E" in column 1)

Column	Description
1	**Line format flag “E”**
5:121	**Event ID (a116)**

Note: Event IDs are built up from the prefix “cec_” (Calibrated Earthquake Cluster), the cluster name, and the event number within the cluster.

Hypocenter Record ("H" in column 1)

Column	Description
1	**Line format flag “H”**
5-8	**Year (i4)**
10:11	**Month (i2)**
13:14	**Day (i2)**
16:17	**Hour (i2)**
19:20	**Minute (i2)**
22:26	**Seconds (f5.2)**
28:32	**OT uncertainty (f5.2)**
35:42	**Latitude (f8.4)**
44:52	**Longitude (f9.4)**
54:56	**S_{min} azimuth (i3)**
58:62	**Error ellipse S_{min}, (f5.2)**
64:68	**Error ellipse S_{maj} (f5.2)**
70:74	**Focal Depth (f5.1)**
76:76	**Depth code (a1)**
78:82	**Plus depth uncertainty (f5.1)**
84:88	**Minus depth uncertainty (f5.1)**
90:93	**GTCU (a4)**
95:102	**Author (a8)**
104:121	**Cluster ID (a18)**

Note: Both depth uncertainties are provided as positive numbers. “Plus” depth uncertainty is on the deeper side; “Minus” uncertainty is shallower, and therefore should not be greater than the focal depth in absolute value.

The “GTCU” field carries a four-character code relating to calibration status (I prefer this term to “ground truth” level). It could be the GTX formulation (e.g., Bondar et al., (2004)), but I have developed the GTCU nomenclature to provide much more detailed information on the subject of what hypocentral parameters are considered to be calibrated (i.e., thought to be bias-free). The GTCU nomenclature is documented fully elsewhere.

The “depth code” in column 76 defines the nature of the depth constraint. Standard values are:

Depth Code	Meaning
c	Cluster default depth
d	Teleseismic depth phases
e	Engineered (man-made explosion)
f	Fault model (InSAR, GPS, etc)
l	Local-distance readings
m	*mloc* solution with free depth
n	Near-source station readings
r	Relocation outside *mloc* with free depth
u	Unknown
w	Waveform analysis

Magnitude Record ("M" in column 1)

Column	Description
1	**Line format flag “M”**
5:8	**Magnitude (f4.2)**
10:14	**Magnitude scale (a5)**
16:110	**Author and comments (a95)**

Phase Reading Record ("P" in column 1)

Column	Description
1	**Line format flag “P”**
3	**Usage flag (a1)**
5:10	**Station code (a6)**
12:19	**Deployment code (a8)**
21:26	**Epicentral distance (f6.2)**
28:30	**Azimuth, event to station (i3)**
32:39	**Phase name (a8)**
41:44	**Arrival time year (i4)**
46:47	**Arrival time month (i2)**
49:50	**Arrival time day (i2)**

52:53	**Arrival time hour (i2)**
55:56	**Arrival time minute (i2)**
58:63	**Arrival time seconds (f6.3)**
65:66	**Reading precision (i2)**
68:73	**TT residual (f6.2)**
75:79	**Empirical reading error (f5.2)**

Note: The usage flag will be either "+", meaning the reading was used for either the hypocentroid or cluster vectors or both, "-", meaning the reading was not used in the relocation for some reason, or "x" meaning that the reading was flagged as an outlier and not used for relocation. It is important to note that the factors controlling how an individual phase reading was used (or not) in an *mloc* analysis are quite complex and there is no information in the v1.4 format concerning why a reading was not used, except for the special case when it has been identified as an outlier. Reference to other *mloc* input and output files is needed to answer that question.

The arrival time seconds field is written with the precision implied by the "reading precision" field.

Stop Event Record ("S" in column 1)

Column	Description
1:1	**Line format flag "S"**

Note: Only the "S" in column 1 is required, but for better readability it is useful to write "STOP" in columns 1:4.

End of File Record ("EOF" in columns 1-3)

Column	Description
1:3	**Line format flag "EOF"**

Changes from Previous Versions

Changes from v1.41 to v1.4.2

- In **phase records**, an 8-character field for the deployment code is inserted after the station code. Under IASPEI's recommended ADSLC station-definition format the deployment code can be up to 8 characters in length. It is legitimate to use FDSN network codes (2 characters) as deployment codes. Agency codes are not supported.
- All seismograph stations used in the analysis are now listed. Formerly, only stations defined in supplemental station files (i.e., not the master station file, which is based on the ISC's International Registry) were listed.
- **Station records** now include fields for agency and deployment.

Changes from v1.4 to v1.41

- In **phase records**, the travel time residual is now carried with two decimal places of precision, instead of one.

Changes from v1.3.1 to v1.4

- A **station coordinates record**, identified by "C" in column 1, is now defined.
- A **layer velocity record**, identified by "L" in column 1, is now defined.
- **Depth records** are not supported in v1.4.
- Only one **hypocenter line** (the preferred one) is permitted, and the usage code is omitted.
- Usage codes are stripped out of **magnitude records**.
- A "+" usage code is defined in **phase records** for readings that were used.
- A special event ID is defined in **event records**.
- The format of **phase records** is substantially changed.

MNF v1.5

This document defines the format for input of differential time data for use in *mloc*. The format is consider part of the MNF (*mloc* native format) family of data formats, and is assigned version 1.5.0

Background

The multiple event relocation code *mloc* was initially designed to use arrival time data only, but like most multiple event relocation codes, arrival time data is converted to differential times (i.e., the same phase at a specific station, observed from two or more events) to obtain improved resolution of the relative locations of events in the cluster. As the use (and evident value) of differential time data from waveform cross-correlation has increased in earthquake location algorithms, *mloc's* code was modified to allow explicit input of differential time data. This change was first implemented in v6.0 of *mloc*, in September 2006. At that time, the only source of waveform cross-correlation data for *mloc* was from Benjamin Kohl of SAIC and code was written to read the specific data format he provided.

Later, differential time data from waveform cross-correlation was provided by Michael Begnaud of LANL for a special study, and the code used to read Kohl's data format was hastily modified to work with the Begnaud's data. Currently, Will Yeck of NEIC is beginning to produce differential time data for use in *mloc* and it seems desirable to stabilize the file format for this kind of data, rather than create yet another subroutine to read a specific researcher's format.

Parameterization of Differential Time Data

There are several ways in which differential time data can be reported. The MNF v1.5 format supports only one form, "reduced relative arrival time". To understand this parameterization of

differential time, it is helpful to think of measuring the time lapse between the arrival times at some station of two phase groups of the same type (say, teleseismic P) with cross-correlation and reporting the full time (in seconds), including the years, months, and days between the two events being compared. We call this the “relative arrival time”. If we ignore the differences in years, months, and days for this calculation, and treat the two events as if they occurred on the same day, then the time difference between the two phase groups is considered “reduced” and we have the “reduced relative arrival time”. This value will be less than 86,400 (in absolute value), the number of seconds in one day, but it may be positive or negative, depending on which event is taken as the reference or “template” event:

reduced_relative_arrival_time = reduced_target_arrival_time – reduced_template_arrival_time

where the template and target event arrival times have been reduced to the same day.

For use in *mloc*, differential time data provided in the reduced relative arrival time parameterization is converted into dummy arrival times for the template and target events:

template_dummy_arrival_time = template_origin_time + theoretical_TT

target_dummy_arrival_time = template_dummy_arrival_time + reduced_relative_arrival_time

where the theoretical travel-time is calculated for the phase of interest at the appropriate station, using the current best estimate of the hypocenter of the template event (with origin time in the reduced sense). The initial origin time of the template event used in this formulation can be provided internally in *mloc* in several ways, but the underlying differential arrival time datum is preserved.

This is a natural approach in *mloc* because all arrival time data are processed in “reduced” form, i.e., as if all events occurred on the same day. The dummy arrival times are combined with the other, genuine arrival time data available for the template and target events. Dummy times for differential data are only used for estimating the cluster vectors, so only their time difference is used and the potential bias from the assumed initial origin time of the template event and the theoretical travel-time calculation are removed by differencing.

Description and Features

Like other varieties of MNF, v1.5 uses several types of ‘records’ with defined formats and a character in column 1 that defines the type of record. Unlike the other MNF formats, which are event-oriented (i.e., records are grouped by event), v1.5 data files are cluster-oriented, and each differential time record carries references to two events in the cluster. Only 4 record types are defined:

Flag	Record Type	Minimum Length	Full Length
F	Format Version	14	14
D	Differential Time	87	149
EOF	End of File	3	3
#	Comment	1	149

Unlike all other record types, which are distinguished by the flag in column 1, the **end-of-file record** (“EOF”) uses columns 1:3; it has no other arguments. It causes processing of a data file to end, so it would normally only be found once, at the end of the file. If an **end-of-file record** is placed in the middle of a file, processing will stop there.

The **differential time record** carries a "usage" flag (in column 3) that determines the way (or if) the information in that record will be used by *mloc*. None of the other record types carries a usage flag.

Comment records (flag “#“) can be inserted anywhere in an MNF-formatted file, but should not be the first or last record. Obviously, any text that occurs after the first **end-of-file record** will not be processed and can be considered as a comment, regardless of the formatting, but the use of the **end-of-file record** in this manner is not recommended.

Information about the nature of the differential time data set can be carried in **comment records**.

File Structure

The first line in a data file must be a **format record**. This is followed by some number of **differential time records** (and **comment records**, optionally**)** and the file must end with an **end-of-file record**.

Defined Record Types

The concept of “optional” fields in the descriptions of record types is specifically in the context of use by *mloc*. Fields that are required by *mloc* are printed in bold below. In writing MNF-formatted data files it is advisable to pad lines to the full length of that record type, which is based on the defined fields, not the required fields, and it is not unwise to pad all lines to 149 characters, the full length of the longest defined record types, regardless of record type.

Format Version Record (“F” in column 1)

Column	Description
1	**Line format flag “F”**
10-14	**Format version (a5)**

It is useful for readability to include extra text, such that columns 1:8 read “F MNF v”, but all that is required is the “F” in column 1 and the version number in columns 10:14. Version number is treated by *mloc* as a character string that can accommodate three-level versioning (e.g., ‘1.5.1’).

Differential Time Record (“D” in column 1)

Column	Description
1	**Line format flag “D”**
3	Usage flag (a1), optional
5:20	**Template Event Designator (a16)**

22:31	Template Event EVID (a10), optional
33:48	**Target Event Designator (a16)**
50:59	Target Event EVID (a10), optional
61:66	**Station (a6)**
68:75	**Phase (a8)**
77:87	**Reduced Relative Arrival Time (f11.4)**
89:90	Reading Precision (i2), optional
92:97	Uncertainty, s (f6.4), optional
99:103	Correlation Coefficient (f5.3), optional
105:112	Original phase name, optional (a8)
114:118	Agency, optional (a5)
120:127	Deployment or network, optional (a8)
129:133	Station, optional (a5)
135:136	Location, optional (a2)
138:140	Channel, optional (a3)
142:149	Author, optional (a8)

The usage field is used here the same way as it is used in MNF v1.3 phase records to flag certain readings so that they will not be used. Some common values that could be relevant to differential time data are:

Value	Meaning
x	outlier, do not use
d	duplicate reading, do not use
m	missing station coordinates, do not use
p	phase that is unknown or not used

The “event designator” field for the template and target events carries an identifier for each event based on an estimate of the date and origin time, to the nearest second. The format is:

yyyymmdd.hhmm.ss

with the character ‘0’ filling any blanks. Although the identifier could be read as numeric values for year, month, day, etc, *mloc* treats it as a character string that is only used to make the correct association with two corresponding events in the cluster being relocated. An equivalent event identifier is declared in the *mloc* command file for each event, using the ‘even’ command. Therefore it is desirable to ensure that the integer seconds part of the identifiers match. The evid (event id) fields are provided for the same reason. While evids are not always available, if they are they provide a more robust way to make the association with events in the cluster. *mloc* attempts to make the match using evids first.

The definition of ‘reduced relative arrival time’ is discussed above.

Reading precision is an integer that indicates the number of significant decimal places in the reduced relative arrival time datum. It is used in *mloc* to correctly format output. The values appropriate for differential time data are:

Value	Meaning
0	nearest second
-1	nearest tenth
-2	nearest hundredth
-3	nearest thousandth
-4	nearest ten-thousandth

If no value for reading precision is provided, *mloc* attempts to determine it from the formatting of the datum. No great harm will be done if this process is inaccurate, but specification of a reading precision is strongly recommended as good practice.

If an uncertainty for the estimate of reduced relative arrival time is provided, it will be read and may be used in *mloc*, at least initially, although it can be over-ridden in various ways. It is desirable to have an estimate of uncertainty from the cross-correlation analysis but *mloc* can proceed without one.

If the correlation coefficient field is non-blank, it is read by *mloc*. It can be used on input to filter out differential times with correlation coefficients below a threshold set by the user. It may also be helpful in deciding whether or not to flag a particular datum as an outlier.

The “original phase name” field provides a way for the user to change the phase name of a differential time datum without losing the information on the original phase ID. The same facility exists in the MNF v1.3 format for arrival time data, but this is likely to be a rare need with differential time data.

The next five fields, all optional, identify the seismograph instrumentation from which the measurement of reduced relative arrival time was made, in the new IASPEI Station Coding Standard. Known as the ADSLC code, this identification standard greatly expands the traditional concept of describing the source of seismic data by a single 3-5 character station code. See the documentation for MNF v1.3 for a fuller discussion of this, especially the rationale for carrying station code in two places in the record. Channel information may be of special significance with differential time data because it is common practice to carry out the cross-correlation analysis on more than one channel of a station and select the one with the highest correlation coefficient for use in relocation. Therefore differential time data sets may well include multiple estimates of reduced relative arrival time, differing only in the channel that was analyzed.

The author field identifies the source (usually a person) of the datum. The field is optional but it is strongly recommended that it be utilized. The character string employed to identify someone is completely arbitrary, but it is limited to 8 characters in *mloc*.

Comment Record ("#" in column 1)

Column	Description
1	**Line format flag “#”**
2:149	Comment, optional (a148)

The length of the comment field is rather arbitrary but it is convenient to limit the **comment record** to the length of the main ‘data’ record, in this case the **differential time record**. *mloc* ignores **comment records**.

End of File Record ("EOF" in columns 1-3)

Column	Description
1:3	**Line format flag “EOF”**

Obtaining ISC Data for *mloc*

Last updated: November 29, 2013

The multiple event relocation program *mloc* recognizes only one format for files carrying arrival time data: MLOC Native Format or MNF. The current version of MNF is v1.3; it is fully defined in a separate document. *mloc* can still read and process data files in MNF v1.1 and v1.2. No data center provides data in MNF format so it will be necessary for users of *mloc* to convert from other formats into MNF. This document describes the procedure for obtaining data from the ISC website and converting it into MNF v1.3 format for use in *mloc*. The source codes of the utility programs used in this process are provided with the *mloc* distribution and may serve as templates for conversion programs for other formats. Please contact me if you need help with a conversion program for some commonly-encountered format, as I may already have written one.

Searching the ISC Bulletin

1. Navigate to:
   <http://www.isc.ac.uk/iscbulletin/search/bulletin/>
   or the mirror site at
   <http://isc-mirror.iris.washington.edu/iscbulletin/search/bulletin/>
2. Select the desired database. The "Reviewed ISC Bulletin" is of higher quality but is usually about 2 years behind real time. The "ISC Bulletin" will provide data for events that are quite recent, within the last day or so, but the quality of the data and the hypocenters is, well, unreviewed.
3. Select "ISF Bulletin" as output format. The data file that you will obtain is not actually in ISF format, but rather in a version of IMS format. I have not investigated why this is the case.
4. Choose search region, time period, and optional search parameters to narrow your search. Most of these options make no difference for *mloc*, but selecting "Include events without defining phases" is not going to be helpful as far as relocation goes. It may be helpful to specify a modest value (say, 20) for "Min defining phases" to weed out events with only a few readings. You will be able to do a selection with minimum number of readings at a later stage of the conversion process, so I recommend keeping this value fairly small in the web search, to avoid losing good events. I have the impression that setting a minimum number of defining phases only works properly for the Reviewed ISC Bulletin, but I have not confirmed this.
5. Choose output options. Select "Prime hypocenters", "phases" and "magnitudes". Deselect the other check boxes.
6. Click the "Search Bulletin" button. After a short delay (maybe a longer delay if you are using the home ISC server), a window will pop up and start filling up with data. It may take a while for all the data to transfer. At the end you will see a STOP line and some other output.

7. Select everything in the output window and paste it into a new text document, named in some reasonable fashion after the nature of the search you have done.

You now have a data file in what I refer to as "ISC" format (but it is only one of several formats the ISC can serve), that needs to be converted into MNF format. Make a new directory for this file with an appropriate name.

Conversion to an MNF Bulletin

The next step in the conversion process is to turn the ISC-format text file, which may be thought of as a seismic bulletin, into a bulletin in MNF format. See the MNF documentation for discussion of the bulletin concept and how it is handled in MNF. The program *isc_ims2mnf* (actually, just the Fortran source code isc_ims2mnf.f90) is provided for this purpose. So you will need to compile the program:

1. Place the file isc_ims2mnf.f90 and the include file isc_ims2mnf.inc into a convenient folder and compile with whatever basic command your Fortran compiler uses, for example: f90 isc_ims2mnf.f90 -o isc_ims2mnf
2. Then copy the executable *isc_ims2mnf* into the folder containing the ISC data file.

Before running the conversion code, it is useful to do a little editing on the ISC data file. If you don't, you'll need to do a little editing on the MNF file afterwards. The editing of the ISC data file consists of:

1. At the top, delete all lines above the line "DATA_TYPE BULLETIN IMS1.0:short".
2. At the end, delete all lines below the line "STOP"
3. Delete one blank line above the line "STOP", such that there is only a single blank line between the last phase line and the "STOP" line.

I have provided a sample of an ISC data file in two forms, "isc_test1.txt" which is copied straight from the ISC website and "isc_test2.txt" which is the same file edited as recommended above. Here is the terminal session in which I ran *isc_ims2mnf* on both data files:

```
$ cd /Users/eab/Desktop/untitled\ folder/test
$ ./isc_ims2mnf
Enter input filename:
isc_test1.txt
Bulletin output?:
y
Bulletin comment:
test1
EOF reached after      6 events
$ ./isc_ims2mnf
```

```
Enter input filename:
isc_test2.txt
Bulletin output?:
y
Bulletin comment:
test2
EOF reached after      6 events
$ ls -all
total 768
drwxr-xr-x  8 eab  staff     272 Nov 29 22:43 .
drwxr-xr-x  7 eab  staff     238 Nov 29 22:38 ..
-rw-r--r--@ 1 eab  staff    6148 Nov 29 22:42 .DS_Store
-rwxr-xr-x  1 eab  staff  210924 Nov 19 18:46 isc_ims2mnf
-rw-r--r--@ 1 eab  staff   43138 Nov 29 22:37 isc_test1.txt
-rw-r--r--  1 eab  staff   40016 Nov 29 22:42 isc_test1.txt.mnf
-rw-r--r--@ 1 eab  staff   41768 Nov 29 22:41 isc_test2.txt
-rw-r--r--  1 eab  staff   38552 Nov 29 22:43 isc_test2.txt.mnf
$
```

There are now two MNF-formatted files, derived from the two versions (original and edited) of the ISC data file. By comparing the two files you will easily see the consequences of not editing the input file, and you can decide for yourself whether you'd rather edit the input ISC data file or clean up the MNF bulletin file afterwards. Further processing will be done with isc_test2.txt.mnf.

Searching an MNF Bulletin and Extracting Files

The next step is to extract data files for selected events from a bulletin formatted in MNF. *mloc* requires the input data to be presented as individual files. In fact you could have extracted individual data files in the previous step, while converting the ISC-format data file to MNF, by answering 'n' to the question "Bulletin output?". This could be a useful shortcut in the case where the ISC-formatted data file contained only a few events and you knew that you wanted them all. In most cases I find it easier to do a broader search in the ISC bulletin and then use a separate process to search that data set and extract certain events for relocation. The program used for this is called *mnf_search*, and the source code for this program (mnf_search.f90) is provided in the *mloc* distribution.

Like *isc_ims2mnf*, *mnf_search* is compiled using a plain vanilla compiler command: "f90 mnf_search.f90 -o mnf_search", where you'll need to substitute the executable name for your own Fortran compiler. Then move a copy of the executable into the directory where you've stored isc_test2.txt.mnf. Here is a terminal session to illustrate how *mnf_search* works:

```
$ cd /Users/eab/Desktop/mloc\ distribution/Utilities/isc_test
$ ./mnf_search
Enter input filename:
isc_test2.txt.mnf
Create mloc command file?
y
Enter command file basename:
test2
Use lat-lon limits?
n
     6 events read
     6 events pass the search criteria
   6 events  that pass the search criteria and have  10 or more phase readings
   6 events  that pass the search criteria and have  20 or more phase readings
   4 events  that pass the search criteria and have  30 or more phase readings
   4 events  that pass the search criteria and have  40 or more phase readings
   4 events  that pass the search criteria and have  50 or more phase readings
   1 events  that pass the search criteria and have  60 or more phase readings
   1 events  that pass the search criteria and have  70 or more phase readings
   1 events  that pass the search criteria and have  80 or more phase readings
   0 events  that pass the search criteria and have  90 or more phase readings
   0 events  that pass the search criteria and have 100 or more phase readings
Minimum number of phase arrivals:
30
Event number selection: beginning and end numbers:
1 6
EOF reached after      6 events
   4 events selected
$ ls -al
total 1256
drwxr-xr-x  14 eab  staff     476 Nov 30 16:46 .
drwxr-xr-x   7 eab  staff     238 Nov 30 10:35 ..
-rw-r--r--@  1 eab  staff    6148 Nov 29 22:42 .DS_Store
-rw-r--r--   1 eab  staff    7329 Nov 30 16:46 20110531.0113.25.mnf
```

```
-rw-r--r--   1 eab  staff   10989 Nov 30 16:46 20110531.0541.20.mnf
-rw-r--r--   1 eab  staff    6963 Nov 30 16:46 20110531.0646.19.mnf
-rw-r--r--   1 eab  staff    6475 Nov 30 16:46 20110531.1345.22.mnf
-rwxr-xr-x   1 eab  staff  210924 Nov 19 18:46 isc_ims2mnf
-rw-r--r--@  1 eab  staff   43138 Nov 29 22:37 isc_test1.txt
-rw-r--r--   1 eab  staff   40016 Nov 29 22:42 isc_test1.txt.mnf
-rw-r--r--@  1 eab  staff   41768 Nov 29 22:41 isc_test2.txt
-rw-r--r--   1 eab  staff   38552 Nov 29 22:43 isc_test2.txt.mnf
-rwxr-xr-x   1 eab  staff  205692 Nov 19 19:00 mnf_search
-rw-r--r--   1 eab  staff     212 Nov 30 16:46 test2.cfil
$
```

After launching *mnf_search* and specifying the input file, you need to decide if you want the program to create a command file for you or not. *mnf_search* cannot create the complete command file, but it creates the event definition section (memb-even-inpu) that is so tedious to write by hand, so usually you will want to take advantage of this offer.

The search options available in *mnf_search* are rather limited (so far). The options are:

- Search a rectangular region (lat-long bounds)
- Search by minimum number of phase readings
- Search by event number range

The search by minimum number of phase readings is a useful way to pare down the number of events in a cluster to a number that is comfortable for *mloc* (i.e., 200 or less). It usually makes sense to start with the events that have the most phase readings. You can go back later and pull out events with fewer readings if you like. You can enter any value for the minimum number of phase_readings (not just multiples of 10).

Duplicate Readings

One problem that has arisen in recent years with the ISC is that they receive many duplicate phase readings because of the multiple distribution channels that exist for many seismograph stations. The duplicates are not always perfect; sometimes they vary slightly in arrival time or have different levels of precision which make it very difficult to be sure if they should be treated as separate samples or as duplicates. The ISC is so far incapable of managing this problem, so it falls to you and your programs to decide what to do. *mloc* contains some logic to detect duplicates and flag them, but it is not perfect. As far as *mnf_search* is concerned, duplicates count towards the number of phase readings so you will find that some of the events that you thought had plenty of readings actually are much less well recorded.

Searching by Event Number

The option for searching by event number is useful when searching through a very large bulletin, containing thousands of events for a region of interest. This often happens in mainshock aftershock sequences when a temporary network is installed that records small aftershocks that would normally be missed by the permanent network. In that case you may decide to break the analysis down to a set of subclusters based on chronological order. In other words, you might limit a search to the first 200 events (events 1-200) for one cluster, then a second cluster would be produced by searching only events 201-400 and so on. If you use lat-long bounds and minimum number of phase readings as search options it will be difficult to predict how many events you'll get from each search, but a bit of experimentation should make it possible to guide the process effectively.

In the example case above I did not use any latitude-longitude bounds for the search and there were only 6 events in the bulletin so it made no sense to use event number selection. The only search option was for the minimum number of phase readings (30 or more), which selected 4 of the 6 events in the bulletin. The individual event files (and test2.cfil) can now be moved or copied into a new directory for the relocation of this cluster. Before running *mloc*, you'll have to edit test2.cfil to add the other commands needed for your particular analysis.

Station File Formats for *mloc*

Last updated: September 4, 2017

This document describes the formats for use for station data files in the multiple event relocation program *mloc*. There is a "standard" format designed specifically for *mloc* which is used for the master station file, but for convenience and backward compatibility, several other formats are supported for use as supplemental station data files (command "sstn"). The format used for the master file has changed several times during development of *mloc*. The current master station file format was introduced with *mloc* v10.0.0, released on April 28, 2014.

Format Index

The first line of all station data files must be a header line that carries a 1-digit integer (*isstn*) in the first column that defines the format of the data file. *mloc* reads this digit and branches to different sections of code to read and process the appropriate format. The rest of the header line (up to 96 characters total for the line) is considered an optional comment, which can be used to describe the dataset.

The current list of defined values for *isstn* is given in the following table:

ISSTN	Description
0	Master station data file format
1	ISC FFB format (geographic coordinates in deg-min-sec*10, elevation in m)
2	SEISAN station list, with station code starting in column 3. Degree-minute-decimal seconds.
3	Simplified format, decimal degrees, geographic coordinates. A basic format for putting in supplemental station data, supports 6-character station codes.
4	China Seismic Bureau format
5	NEIC format
6	MSU format (Kevin Mackey)
7	undefined
8	undefined
9	Previous master station file format

Details of each format can be found in subroutine "redsta" in mloclib_stations.f90.

Master Station File Format (*isstn* = 0)

The master station file used by mloc is keyed to the International Registry of Seismograph Stations (IR) maintained at the ISC. Specifically, it should contain only station codes that have been registered in the IR, even if the details (e.g., coordinates) have been altered (corrected). To use a station with a code that has not been registered, a supplemental station data file should be

declared (command sstn) in *mloc*. A supplemental station file can use the master station file format.

The format used for the master station file is unique to *mloc*. It departs from standard practice in permitting multiple entries for the same station code. There are two main reasons for this complexity:

1. In some cases stations have been moved without changing the station code. The format includes the concept of epochs that define when a station occupied the given location. Therefore it is necessary to check the date for which coordinates of a given station code are needed to ensure that the correct coordinates are used.
2. The coordinates carried in the IR are sometimes found to be in error, for example, by investigation with Google Earth. Rather than delete the IR coordinates, it is preferable to put the improved information in as a new entry for the same epoch, thus documenting the discrepancy. The concept of authorship of an entry also helps in this regard.

The order in which "duplicate" entries in the master station file are listed controls which coordinates are used: **the first entry that matches the station code and the operational epoch will be used**.

Station Codes

The master station file format carries a maximum of 5 characters for station code, following the standard for the IR. *mloc* can accommodate 6-character station codes, but information about such stations must be provided in a supplemental station data file in a format that allows 6 character codes, for example, the Simplified format (*isstn* = 3) defined below.

Operational Epoch

The period of time during which a set of coordinates is valid for a given station code is called the "operational epoch". It is defined by two variables, *date_on* and *date_off*, which are given as a 7-digit integer composed of the year (left-most four digits) and Julian date (right-most three digits). Either one or both may be left blank, with the obvious meaning.

Seeding of the operational epochs for many codes in the master station file was based on an analysis of data holdings at Lawrence Livermore National Laboratory by Steve Myers, supplemented by information acquired by Bob Engdahl. I have found that there are many errors in these epoch data, especially in the *date_off* field, such that *mloc* often reports a "date range" failure in processing actual arrival time data against the master station file. In the case of erroneous *date_off* entries the problem is usually that Steve filled in the field with the date of the most recent data holding at that time, but obviously most stations have continued to operate past that point; therefore the correct course of action is to delete the *date_off* entry completely. In the case of violations of the *date_on* entry, the preferred course of action is usually to update the *date_on* value to the earliest date of the available data.

The definition of the SEISAN station file format (*isstn* = 2) has been extended to include optional *date_on* and *date_off* fields, because this format is often used for temporary networks that had a limited period of operation, and the station codes for such deployments often conflict with registered stations.

Authorship

I have introduced the notion of authorship to the master station data file to help document discrepancies in reported station coordinates that are often encountered. The author of a set of coordinates is carried in a character variable with 8 characters, the same as authorship for hypocenters and phase readings in the ISF format and the MLOC Native Format for arrival time data. There are no standard definitions for these authorship entries, but I have used "IR" as the code for entries downloaded from the International Registry, the base set of coordinate information for the master station file. Definitions of authorship codes can be carried in the beginning of the master station file in lines that begin with '#' in column 1 (comment lines).

A standard variation on authorship codes is the use of the '+' character to indicate a different author for the information on station elevation and depth of burial (if given) than the author appropriate for the latitude and longitude. This is useful because the IR in many cases does not carry any information on station elevation. A very easy way to rectify this lapse is to enter the coordinates in Google Earth and take the elevation from that, in which case the authorship code would be amended from "IR" to "IR+GE". In some cases, where the seismic vault can be clearly identified in Google earth, I have used the authorship code "EAB+GE" to indicate that I specified all coordinates of the station using Google Earth.

Station Elevation and Depth of Burial

The master station file format follows the IR practice of carrying two values related to the vertical coordinate of a station. Elevation is the elevation of the ground surface at the latitude and longitude provided, and depth of burial defines how far below the surface the sensor is emplaced. Both values are given in meters. Elevation can be a negative value, but depth of burial is always positive. Both values follow the IR in assuming the reference plane to be the reference geoid WGS84.

In reality, depth of burial information is often missing in the IR (and therefore, in the master station file) and in some cases the elevation field has been filled with the elevation at the sensor. It is very difficult and time-consuming to correct these errors, but in most cases the discrepancy in elevation will be on the order of 100 m or less, which is usually not very important in the context of earthquake location. If corrections to depth of burial and/or elevation are made, the authorship code should be modified accordingly.

Format

The first line must contain the digit "0" in column 1, to indicate the format. Comment lines, indicated by '#' in column 1, can be included anywhere in the file. All other lines in the file have the following format:

Column	Description
1:5	Station code (a5)
7:15	Latitude (f9.5)
17:26	Longitude (10.5)
28:32	Elevation, m (i5)
34:37	Depth of burial, m (i4)
39:46	Author (a8)
48:52	Agency (a5)
54:61	Deployment (a8)
63:64	Location (a2)
66:72	Date_on (i7)
74:80	Date_off (i7)
82:	Station name (no limit)

Supplemental Station File Formats

Master Station File Format (isstn = 0)

See above.

ISC Fixed Format (isstn = 1)

This format is based on the record type 91 used in the ISC's old Fixed Format ("96 Byte") data format, extended to support 6-character station codes. It features geographic coordinates in deg-min-sec*10, elevation in m.

Column	Description
15:20	station code (a6)
62:63	Latitude degrees (i2)
64:65	Latitude minutes (i2)
66:68	Latitude seconds*10 (i3)
69:69	"N" or "S" (a1)
70:72	Longitude degrees (i3)
73:74	Longitude minutes (i2)
75:77	Longitude seconds*10 (i3)
78:78	"E" or "W" (a1)
79:82	Elevation, m (i4)

SEISAN Station Format (isstn = 2)

The standard form of station data produced by the SEISAN software. Limited to 4-characters for station code. Geographic coordinates in degrees-decimal minutes. Elevation in m. The fields *date-on* and *date_off* are optional.

Column	Description
3:6	Station code (a4)
7:8	Latitude degrees (i2)
9:13	Latitude minutes (f5.2)
14:14	“N” or “S” (a1)
15:17	Longitude degrees (i3)
18:22	Longitude minutes (f5.2)
23:23	“E” or “W” (a1)
24:27	Elevation, m (i4)
34:40	Date_on (i7)
42:48	Date_off (i7)

Simplified Format (isstn = 3)

This is a basic format designed for quick entry of a few supplemental stations. Geographic coordinates in decimal degrees. Supports 6-character station codes, operational epoch and both station elevation and depth of burial (positive number). Elevation, depth of burial and operational epoch are all optional.

Column	Description
1:6	Station code (a6)
8:15	Latitude (f8.4)
17:25	Longitude (f9.4)
27:31	Elevation, m (i5)
33:37	Depth of burial, m (i5)
39:45	Date_on (i7)
47:53	Date_off (i7)

China Seismic Bureau Format (isstn = 4)

A format used by the China Seismic Bureau for their station lists. Uses a 3-character, lower-case station code. Only defined for positive latitude and longitude, but could be easily extended.

Column	Description
1:3	Station code (a3)
5:8	Elevation, m (i4)
10:11	Latitude degrees (i2)
14:15	Latitude minutes (i2)

18:21	Latitude seconds (f4.1)
25:27	Longitude degrees (i3)
30:31	Longitude minutes (i2)
34:37	Longitude seconds (f4.1)

NEIC Format (isstn = 5)

This format is for use with station coordinate data returned from the NEIC metadata server. If you use arrival time data from the NEIC ComCat server, especially for events within the U.S., you will probably need a supplemental station file to handle the many stations in U.S. regional networks that have not been registered with the International Registry at the ISC and also to solve the many station code conflicts that exist with registered stations.

The easiest method of building type 5 supplemental station files is to use the *mdget* utility program written and distributed by Dave Ketchum at NEIC <ketchum@usgs.gov>. The format used here is returned by the basic command *mdget -coord -s* (followed by a network/station regular expression and a *-b* and *-e* options to define a time interval, if desired).

Two forms of the command are especially useful for this purpose. It is necessary first to determine which network's version of a given station code is desired. There are cases of several regional networks using the same code and many cases where a network has used a code that is registered for a different station at the ISC. For example the station TCU is searched by

mdget -a *.*.TCU -b all

returning

```
FDSN.IR.TCU.--:24.1475 120.676 84.0
* <EOE>
FDSN.UU.TCU.--:41.1173 -111.4078 2269.0
* <EOE>
FDSN.UU.TCU.01:41.1173 -111.4078 2269.0
* <EOE>
HOLD.TAP.TCU.--:24.1475 120.676 84.0
* <EOE>
ISC.IR.TCU.--:24.1475 120.676 84.0
* <EOE>
NEIC.TAP.TCU.--:24.1475 120.676 84.0
* <EOE>
UUSS.UU.TCU.--:41.1173 -111.4078 2269.0
* <EOE>
UUSS.UU.TCU.01:41.1173 -111.4078 2269.0
* <EOE>
* <EOR>
```

In this case there is a conflict between TCU in the Utah network (network code UU) and TCU in the International Registry (FDSN.IR.TCU). To obtain the properly formatted version of FDSN.UU.TCU for the supplemental station file, and because if you need one UU station you will probably need others, it is convenient to use a different form of mdget to dump the entire UU network:

mdget -coord -s UU.....Z.. -b all > uu_stn.dat

where we redirect the output to a file. The regular expression has 12 characters:

1-2 Network code

3-7 Station code

8-10 Channel code

11-12 Location code

In the example the character "Z" in the 10th position selects only vertical components, which is usually adequate to obtain the station coordinates. The first few lines returned by this command are:

```
UU AIUT  -- EHZ:     0.0 -90.0    0.0:  40.8558 -112.1755 1334.0 1989-09-28 00:00 to 1990-05-10 23:59
* <EOE>
UU AIUT  -- EHZ:     0.0 -90.0    0.0:  40.8558 -112.1755 1334.0 1987-06-25 00:00 to 1989-09-27 23:59
* <EOE>
UU AIUT  -- EHZ:     0.0 -90.0    0.0:  40.8558 -112.1755 1334.0 1985-06-04 00:00 to 1987-06-24 23:59
* <EOE>
```

If you're going to save a full list for this network it is worthwhile to edit the returned list to remove "end of entry" lines, or the desired line can simply be copied and pasted into the type-5 supplemental station file. Although the output of the *mdget* utility includes operational epoch information, *mloc* does not use it.

Column	Description
4:8	Station code (a5)
40:47	Latitude (f8.4)
49:57	Longitude (f9.4)
58:62	Elevation, m (i5)

MSU Format (isstn = 6)

A format used by Kevin Mackey at Michigan State University, especially for data from the former Soviet Union.

Column	Description
1:5	Station code (a5)
6:7	Latitude degrees (i2)
9:10	Latitude minutes (i2)

12:15	Latitude seconds (f4.1)
16:16	“N” or “S” (a1)
17:19	Longitude degrees (i3)
21:22	Longitude minutes (i2)
24:27	Longitude seconds (f4.1)
28:28	“E” or “W” (a1)
30:33	Elevation, m (i4)

Managing Station Data in *mloc*

Last updated: November 24, 2017

This document describes how information about seismograph stations, i.e., their coordinates, is managed in the multiple event relocation program *mloc*. There is a separate document that describes the actual formats for station data that are supported in *mloc*. The strategy for managing station data presented here is specific to v10.4.0 and later versions of *mloc*, in which agency and deployment codes are supported.

The singular goal of this document is to explain how the various features of *mloc* can be used to correctly associate station coordinates (latitude, longitude and elevation) with the arrival time readings in a cluster of events being analyzed with *mloc*. Missing or incorrect station coordinates cause arrival time data to be lost to the relocation, or used in a biased manner, with possibly dire consequences for location accuracy. With experience it is possible to evaluate with considerable confidence whether a few missing or mis-located stations can be safely ignored, but it should be obvious that this is a slippery slope.

Because *mloc* is intended for studies that push the limits of hypocentral location accuracy, a casual attitude toward the complete and correct association of station coordinates is contraindicated.

The Short Version **67**

The New IASPEI Station Coding Standard **67**

NEIC Station Metadata **68**

Other Networks and Stations **68**

Station files in mloc **68**

Master Station File 69

Operational Epoch *69*

Supplemental Station Files 70

Order of Precedence 70

An Alternative to Supplemental Station Files 70

Station Code Conflicts **71**

Missing Station Codes **73**

Skipping Stations **73**

Using Deployment Codes to Resolve Station Code Conflicts **74**

The Short Version

- If you are working only with arrival time data from the ISC Bulletin or from stations that are registered with the International Registry of the Seismograph Stations (IR) at the ISC, you probably don't need to worry about managing station data. The default station data file (master station file) should cover almost everything.
- If *mloc* reports missing stations for ISC-sourced arrival time data, you need to update the master station file.
- If you have data from stations that are not registered at the IR you will need to provide station coordinate information in one or more supplemental station files, using the SSTN command.
- If your dataset includes readings from two or more distinct seismograph stations with the same station code and it is important to use all the data, you will need to tell *mloc* to use deployment codes to resolve the conflicts (the RADF command) and you will need to prepare your event data files and supplemental station files accordingly.

The rest of this document explains the gory details behind this guidance.

The New IASPEI Station Coding Standard

In 2013 IASPEI endorsed a new standard for identifying seismograph stations. A document describing that standard is available at the ISC website:

<http://www.isc.ac.uk/registries/download/IR_implementation.pdf>

This standard is often referred to as the ADSLC standard, in respect of the five defined fields agency, deployment, station, location and channel that take the place of the traditional station code. The MNF data format that is used for event data files in *mloc* has fields for all five elements of the ADSLC standard, but the deployment and station fields are of greatest importance for relocation analysis, and the agency field is useful for informational purposes.

It is important to understand that ADSLC codes are non-unique. That is, a given station can have multiple ADSLC codes, governed by a complex system of aliases and umbrella designations, some of which are mentioned below. This complexity is necessary to handle the multiple ownership and operational affiliations that exist for many seismograph stations.

Unfortunately, little progress has been made in actually supporting the ADSLC standard at the ISC or most other seismological agencies. Specifically, neither arrival time datasets downloaded from the ISC Bulletin nor station information datasets downloaded from the IR carry information concerning agency or deployment. On the other hand, by definition, any arrival time data downloaded from the ISC Bulletin may be legitimately characterized with agency="ISC" and deployment="IR", standing for the International Registry of Seismograph Stations, assuming that the station coordinates associated with the reading are those from the IR itself.

The ADSLC standard provides for a maximum of five characters for the agency and station fields, and eight characters for the deployment field. These field lengths are supported in the

MNF format. Although earlier versions of *mloc* supported 6-character station codes as a mechanism for resolving station code conflicts, with v10.4.0 station codes are limited to 5 characters and conflicts must be resolved with another mechanism, i.e., deployment codes.

It is also important to understand that, although MNF format supports the ADSLC formulation, it is usually best NOT to encode arrival time data files (.mnf files) with that information as a matter of course, even if it is available. *mloc* processes most arrival time datasets more efficiently if the agency and deployment fields in the phase lines of the .mnf files are left blank. In other words, situations requiring the use of agency and deployment fields are rare and matching stations solely by station code works fine most of the time. Therefore it is recommended to use agency and deployment codes only when they actually solve a problem and then they should be applied only to the phase readings that require them.

NEIC Station Metadata

One organization that does support the ADSLC, in a sense, is the NEIC, which maintains a database of station metadata that includes the FDSN network codes. As can be seen in the above-referenced defining document, stations belonging to FDSN-affiliated networks may be legitimately characterized in the ADSLC formulation with agency=”FDSN” and deployment=FDSN network code, a two character alphanumeric code. NEIC processing identifies stations by the combination of station code and FDSN network code.

Complexities arise, however, for several reasons:

- Some stations in the NEIC metadata database are not registered with the IR.
- Some of those unregistered stations in the NEIC metadata database have station codes that conflict with the codes of stations (in other places) that are registered at the IR.
- Some stations in the NEIC metadata database HAVE been registered at the IR, but with station codes that have been modified, typically by adding an extra character to the code.

Other Networks and Stations

With few exceptions, arrival time data obtained from other seismological agencies and organizations will be identified only with a station code. For simplicity of local usage, these codes are frequently three characters in length, which greatly enhances the odds of a conflict if such data are combined with datasets from other sources. Most major regional or national seismograph networks have registered (at the IR) at least some of their stations for international data exchange, but if one obtains datasets directly from the agency it is likely that there will be unregistered stations that lead to station code conflicts. The odds of station code conflicts go up dramatically with data from smaller local networks, civil engineering networks (e.g., dam monitoring networks) and temporary networks, all of which may be provide data of exceptional usefulness in carrying out calibrated relocations with *mloc*.

Station files in *mloc*

To handle station-related problems *mloc* supports multiple files of station information that may be provided in several formats (described in a separate document). There is a limit (23,000 at the time of this writing) to the number of stations that may be defined in mloc. This limit can be easily changed in the file "mloc.inc" by editing the value for the parameter *nmax1*.

Master Station File

The default set of station information is contained in the master station list, a file with the pathname /tables/stn/master_stn.dat, relative to the *mloc* working directory. The master list contains **only stations that have been registered at the IR**, but the coordinates in some cases have been revised, based on what is considered to be more reliable information.

When coordinates are modified, the original entry is not deleted. The new entry is placed above the original one in the list and *mloc* selects the first instance it encounters. The format used for the master station list includes the concept of an "author" for the station coordinates (eight characters) and this is used to annotate the source of the preferred coordinates.

New stations are regularly registered with the IR, so the master station file falls out of date and it must be updated manually when a dataset includes IR-registered stations that are not in the master list. This seldom involves more than a handful of stations for any one cluster.

At the time of this writing, the master station file contains 20,601 entries, leaving room for ~2400 station definitions in supplemental station files.

Operational Epoch

In addition to normal coordinate information, authorship information, and agency and deployment codes, the master station file format can carry information about operational epoch, beginning and/or ending dates. Interpretation of a blank field is obvious. These dates are each encoded in a seven-digit integer field consisting of a four-digit year and a three digit day-of-year.

The main use of operational epochs is to keep track of cases where a station has been moved but kept the same station code, but it may also serve to resolve station code conflicts if the conflicting stations have different operation epochs. In general that use requires that the station that conflicts with an IR-registered station would need to be specified with a format that carries operational epoch information. At present the only formats with this feature are the format of the master station file and the SEISAN format, to which the operational epoch fields have been added as an extension in order to better handle the frequent situation where the SEISAN format is used to define stations of a temporary network.

When comparing the station code of a phase arrival time reading with the list of stations to determine coordinates, a check against the operational epoch will be made if there is any information about it in the station entry. Readings that fail the test are listed in the ".stn" file, and the search continues with the next entry in the station file. This is an exception to the rule that the first-encountered matching station code will be assigned.

Use of operational epoch information can be useful in determining the correct station coordinates, but unfortunately, the data on operation epoch in the master station list is far from reliable. The original attempt to establish operational epochs for the master station file was made by Steve Myers and Bob Engdahl around 2011 or 2012, combining information compiled by Engdahl over many years of research in this field and the dates of first and last entries for stations in the seismological database at Lawrence Livermore National Laboratory. In some cases the ending date for operational epoch is simply the date of the latest entry for a station that continued to operate. Beginning dates tend to be more reliable, but are certainly not error-free. In most cases the correct way to deal with cases of failed date range is, assuming one is confident about the basic association of the station, to either edit the offending date, changing it to the date value of the actual reading in hand, or to delete the offending date in the master station file. Beyond these occasional "edits of opportunity", no systematic effort has been made to improve the accuracy of the operational epoch information in the master station file. It is probably best to leave these fields blank unless very reliable information in available on the subject.

Supplemental Station Files

If a dataset contains arrival time readings from stations not found in the master station file, and those stations are not simply recently-registered stations that need to be added to the master station list, the necessary information can be supplied to *mloc* in one or more supplemental station files. The SSTN command is used to designate supplemental station files. There is a limit of eight for the number of supplemental station files that can be defined. In practice it has never been necessary to use more than 3 or 4 supplemental stations, but the limit can be increased by editing the parameter *n_supp_stn_file_max* in the "mloc.inc" file and re-compiling *mloc*.

A supplemental station file can carry any number of entries, up to the limit imposed in *mloc* for the total number of defined stations, and this limit can be changed in the file "mloc.inc", after which *mloc* would need to be recompiled. At present there is space for about 2400 stations to be defined in supplemental station files. Supplemental station files can be in any of several formats, including the same format as the master station file.

Order of Precedence

If any supplemental station files have been declared with the SSTN command, they are read by *mloc* first, in the order declared. Then the master station list is read, completing a single list of station definitions against which the arrival time data will be compared to determine the associated station coordinates.

In the default mode of operation only station codes and operational epoch are compared and the first instance in which the station codes match (and the date of the reading satisfies whatever information exists concerning operational epoch) is selected as the source for station coordinate information. If necessary a more advanced mode can be selected, in which deployment code is also used to determine a match. This is discussed below.

An Alternative to Supplemental Station Files

When working with a dataset containing station codes that are not in the master station file (i.e., not registered at the IR), the standard procedure is to supply a supplemental station file with the missing station codes and coordinates. In certain cases there is an alternative approach that should be used.

It is not uncommon for local, regional and national-scale seismic networks to have been established with station codes that were selected for local convenience, with no consideration of conflicts with IR-registered stations. The processing software is written with these codes embedded at a low level. Then at some later date the network operators decide (or they are told) they would like to contribute data to the ISC, at which point they discover that many of their favorite station codes are already taken. So they register their stations at the IR with new codes, usually a variant of the old one, and write software to create the datasets that will be sent to the ISC that aliases the internal station codes to the registered ones. Changing the codes in their processing software would lead to massive headaches, breaking associations with their archives.

All is fine if you retrieve data from such networks from the ISC, but if you are working closely with the network and receive arrival time datasets directly from them, the readings will probably come with the unregistered codes, many of which will be conflicting with registered ones. What to do?

A supplemental station file could be used to solve this problem, although it might lead to having to resolve code conflicts. Since the problematic station codes have already been registered at the IR with different codes, it is possible to solve this problem more gracefully by using the feature of the MNF format that carries station code in two places. One station code field carries the original (unregistered) station code. This should not be altered, for forensic purposes. The other field is the one that is used by *mloc* when the data file is read, and this one can be edited to use the registered station code. Then the station code will be found in the master station file. This editing can be done by hand in the MNF files if the number of instances is small, but it is likely to be worth writing some code to do the aliasing when the native data files are converted to MNF for use in *mloc*.

It may still be necessary to provide a supplemental station file if some of the network's stations have never been registered at all.

Station Code Conflicts

If the arrival time dataset has been obtained from a single source such as the ISC or from a well-managed relational database, the association of station coordinates with phase readings should be trouble-free. In many cases, however, the type of analysis for which *mloc* is designed requires the use of data from multiple sources and therefore the use of supplemental station files, raising the possibility of having to process data from different stations with the same station code.

When supplemental station files are declared, a test for possible station code conflicts is performed and the results are logged in the .stn file. If more than one supplemental station file has been declared, each entry in the second (or later) file is compared to the entries of the previous supplemental station file(s) and cases of matching station code are noted. When the

master station file is read (after all supplemental station files) each entry is tested against all the supplemental station file entries.

For these comparisons, the differences in coordinates are calculated and logged. If the differences are small the issue may not be a station code conflict but rather a discrepancy concerning the coordinates of a station from different sources. In other cases the problem may be a typographical error in one database. Larger differences are almost certain to indicate a station code conflict.

For any cluster that requires one or more supplemental station files, it is strongly recommended to review the .stn file after the first run, to see if any station code conflicts have been reported. It is important to realize, however, that most station code conflicts reported in the .stn will not require any treatment. These are only *potential* station code conflicts, i.e., between two or more station files. The critical issue is whether the arrival time dataset actually contains readings from both of the stations with conflicting codes.

The easiest way to check for true conflicts is to search the .phase_data file for all instances of the station code in question and examine the phase lines closely, especially the travel time residuals. A true station code conflict will almost always result in some (or all) readings for the station having very large residuals. If they are large enough the phase will not be able to be associated and it will show up as "UNKNOWN" and the residual will be the observed travel time, i.e., the time between the event origin time and the observed arrival time of the reading.

If this investigation reveals no anomalies, the entry for the station in question that was found in the supplemental file is probably the correct one for the dataset, and the fact that there is an IR-registered station with the same code in the master station file is of no concern.

If, on the other hand, all the entries found by this search look suspicious, it may be that the entry in the master station file is the one that should be used and the conflicting entry from the supplemental station file should be deleted. This has to be tested.

The most difficult case is when there seem to be two or more subsets of readings with different results, one set with small residuals and another with large residuals. It is helpful to note the reading author fields for clues. For example, readings from the ISC should always use coordinates from the master station file. All station in the relocation are listed in the .stn file along with their coordinates and the source, whether a supplemental station file or the master station file. Another avenue for investigation is to go back to the relevant MNF file and check the epicentral distance and azimuth that is usually reported along with arrival time readings. Those values should be fairly consistent with the ones list in the .phase_data file if the station coordinates have been taken from the correct station file.

If the dataset does contain cases of true station conflict, i.e., arrival time data from different stations with the same code, there are two possible courses of action.

One possibility is to do nothing. This may be justifiable if the consequences of mishandling certain readings is judged to be negligible. If one of the conflicting stations provides data of great

importance in the relocation, especially if they are used for the hypocentroid in a direct calibration, that data should obviously be getting the correct station coordinates, usually from a supplemental station file. If the other conflicting station represents a far-teleseismic station that only provides a few readings (for example) for the largest event in the cluster and those readings are shared with only a few (or none) of the other events, such that they contribute almost nothing to the relocation, it may be justified to ignore the problem. The problematic readings can be flagged manually or flagged in bulk processing by the utility program XDAT. If nothing overt is done the windowing algorithm (command WIND) will still ensure that they are not used in the relocation.

If it is determined that the readings from both of the conflicting stations should be kept in the relocation, it will be necessary to take some extra steps to make use of deployment codes to distinguish between the two stations. This is described below.

Missing Station Codes

If a phase reading's station code is not found in the master station file or any supplemental station files, it is added to a list of missing station codes. That list is logged in the .stn file, along with the number of instances of each missing code. Missing station codes with many instances are obviously of greater importance than missing codes with only a few instances. In the extreme case of a single instance, that station would not contribute to the relocation at all unless it happens to be suitable for estimating the hypocentroid in a direct calibration. If that possibility can be dismissed, it is quite defensible to ignore the issue.

Missing codes for each event are also listed in the .phase_data file, in the "Bad Data" section. Missing codes for events with few readings or poor azimuthal coverage are of greater interest than those of large, well-recorded events.

The first place to search for missing codes is the International Registry of Seismograph Stations at the ISC. The master station file may simply need updating. The source of the arrival time dataset containing the unknown station code is the next place to look. Usually it will be possible to find a list of station codes and coordinates at the organization's website, but not always.

Skipping Stations

Occasionally it happens that a certain station appears to have some problem that causes the observed arrival times to be notably different from the theoretical arrival times. If the differences are systematic and too large to be explained reasonably by lateral heterogeneity or uncertainty in hypocentral location, one may question if the station coordinates that have been used are in error. Another possibility is a problem with the timing system or the code involved in basic recording. Data from older temporary seismograph deployments (before GPS timing and location capabilities) often suffer from such problems.

If the problem cannot be resolved, it may be best to keep data from the station in question out of the relocation. This can be done using the SKIP command. Phase readings that have been skipped with this command will be flagged with "s" in output files.

Using Deployment Codes to Resolve Station Code Conflicts

If the investigations discussed above reveal the presence in a cluster's arrival time dataset of readings from two or more distinct stations having the same station code, and further, that it is desirable to keep all those readings in the relocation, then an additional mechanism (beyond station codes) is needed to distinguish between stations. In *mloc*, this can be done using deployment codes to supplement station codes. This feature was introduced with v10.4.0 in November 2017.

Fortunately, cases in which deployment codes need to be invoked are rather rare. It is important to understand that *potential* conflicts, i.e. when the same code appears in more than one station file, do not necessarily require any action. It is only when the data set contains readings from two or more stations with the same code that the prospect of using deployment codes need be considered, and even then it can be reasonable to conclude that the extra trouble is not warranted.

In earlier versions of *mloc*, this situation could be handled by using the 6th character of the station code field to introduce the necessary distinctions, but arbitrary changes to station code files and arrival time datasets is a very poor practice and can easily lead to confusion and error in the future. This method of solving station code conflicts is not available as of v10.4.0 because station codes are now limited to 5 characters, the limit defined in the ADSLC formulation.

Internally, *mloc* uses only deployment codes to supplement station codes in order to resolve station code conflicts, but in most cases *mloc* carries along the agency code as well and agency codes appear in many *mloc* output files. Agency codes provide an extra layer of information about the identity of a station which can be useful for forensic purposes in *mloc*.

The most common agency codes in *mloc* will be "ISC" for station information derived from the International Registry and "FDSN", which is applicable to any seismograph station belonging to a network that participates in the FDSN. In particular, two-character FDSN network codes can be legally used as deployment codes, with "FDSN" as the corresponding agency code. The NEIC station metadata server is a prime source of information on FDSN network codes. There are still many seismic stations around the world that would not fall into either of these categories. In the absence of an accepted global standard for agency and deployment codes, the *mloc* user will have to invent agency and deployment codes for station lists and arrival time datasets if it is required. The master station file contains several examples in which this has been done, to make it easier to maintain information about sets of stations associated with a given network.

Regardless of other agency/deployment affiliations that may be permissible, all stations in the master station file can be considered to have the agency code "ISC" and deployment code "IR". When *mloc* reads the master station file it assigns "ISC" and "IR" as the agency and deployment codes for internal processing, regardless of what codes are used in the master station file. This uniformity aids in the use of deployment codes to resolve station code conflicts in *mloc*. Otherwise, a great deal of tedious and error-prone work would be required to ensure that the agency and deployment codes for specific readings in event data files match the ones that happen to be used in the master station file.

The key step to using deployment codes in *mloc* to resolve station code conflicts is the command RADF (Read Agency and Deployment Fields). It is used to set a logical variable that is false by default, meaning that the assignment of station coordinates will be based solely on matching station codes. If the command RADF is invoked to make that logical variable true, *mloc* will *try* to use deployment codes as well as station codes for matching readings to station coordinates. The emphasis on "*try*" is very important. If an arrival time reading does not have an entry in the deployment field, *mloc* falls back to matching station codes only. Therefore, even if RADF is invoked to turn on deployment code usage, nothing will change in processing station codes unless some phase readings have deployment codes.

In preparing datasets for use in *mloc*, therefore, it is highly recommended to leave agency and deployment fields blank, even if the appropriate entries for those fields are known. The only exception would be the case where it is known in advance that there will be a significant number of actual (as opposed to *potential* conflicts, discussed above) station code conflicts in the dataset. In that case only readings that will suffer from actual code conflicts need deployment codes.

If you use the command RADF to change the handling of deployment codes (in either direction) from a previous run, the entries for empirical reading errors in the .rderr file from the previous run will no longer match with the current run and *mloc* will use default values for reading errors. A second run with the same setting of RADF will restore the correct use of empirical reading errors.