

**Technical
Instructions**

IDC DOCUMENTATION

Event Location and Magnitude Software User Manual



Approved for public release;
distribution unlimited

Notice

This document was published December 2002 by the Monitoring Systems Operation of Science Applications International Corporation (SAIC) as part of the International Data Centre (IDC) Documentation. Every effort was made to ensure that the information in this document was accurate at the time of publication. However, information is subject to change.

Contributors

Douglas Brumbaugh, Science Applications International Corporation
Gordon Shields, Science Applications International Corporation

Trademarks

BEA TUXEDO is a registered trademark of BEA Systems, Inc.
ORACLE is a registered trademark of Oracle Corporation.
SAIC is a trademark of Science Applications International Corporation.
Solaris is a registered trademark of Sun Microsystems.
SPARC is a registered trademark of Sun Microsystems.
SQL*Plus is a registered trademark of Oracle Corporation.
Sun is a registered trademark of Sun Microsystems.
UltraSPARC is a registered trademark of Sun Microsystems.
UNIX is a registered trademark of UNIX System Labs, Inc.
X Window System is a registered trademark of The Open Group.

Ordering Information

The ordering number for this document is SAIC-02/3043.

This document is cited within other IDC documents as [IDC6.5.15].

Event Location and Magnitude Software User Manual

CONTENTS

About this Document	i
■ PURPOSE	ii
■ SCOPE	ii
■ AUDIENCE	ii
■ RELATED INFORMATION	ii
■ USING THIS DOCUMENT	iii
Conventions	iv
Chapter 1: Introduction	1
■ SOFTWARE OVERVIEW	2
■ STATUS OF DEVELOPMENT	7
■ FUNCTIONALITY	8
Features and Capabilities	8
Performance Characteristics	10
Location Mode	10
Magnitude Mode	11
Combined Mode	11
■ INVENTORY	11
Software	12
Database Tables	12
Location Mode Database Tables	12
Magnitude Mode Database Tables	13
Input Files	14
Location Input Files	14
Magnitude Input Files	15
■ ENVIRONMENT AND STATES OF OPERATION	16
Hardware Environment	16

Software Environment	16
Normal Operational State	16
Contingencies/Alternate States of Operation	17
Chapter 2: Operational Procedures	19
■ SOFTWARE STARTUP	20
Normal Operational Startup	20
Manual Startup	21
■ SOFTWARE SHUTDOWN	21
Automatic Application Shutdown	22
Manual (Shell) Shutdown	22
■ CONTROL PARAMETER SETUP	22
Creating an EvLoc Parameter File	22
General Control Parameters	23
Verbosity Control Parameters	25
Location Control Parameters	26
Magnitude Control Parameters	28
Example Parameter File	28
Obtaining Help	33
■ EARTH-MODEL FILE SETUP	33
Constructing Location Earth-model Files	34
Velocity Model Specification File	35
One-dimensional Travel-time Table	39
Radial 2-D Travel-time Tables	48
Ellipticity Correction Table	49
Source-specific Station Correction Table	49
Location Test-site Correction File	53
Long-period Grid Index File	55
Long-period Phase Velocity File	58
Slowness/Azimuth Station Correction File	60
Constructing Magnitude Earth-model Files	64
Magnitude Description File	64
Transmission Loss Specification File	69
Transmission Loss Model	76
Magnitude Test-site Correction File	85

■ MAINTENANCE	87
Purging Log Files	87
Maintaining Database Integrity	87
Integrating New Models	88
Adding New Stations	88
Updating Static Database Tables	88
Modifying Location Input Files	88
Modifying Magnitude Input Files	89
■ SECURITY	90
Chapter 3: Troubleshooting	91
■ MONITORING	92
Studying Processing Status	92
Querying the Database	93
Examining Log Files	93
■ INTERPRETING ERROR MESSAGES	93
Alphabetical Error Index	94
EvLoc Configuration-related Error Messages	101
libloc Configuration-related Error Messages	110
libmagnitude Configuration-related Error Messages	123
libloc Processing-related Error Messages	137
libmagnitude Processing-related Error Messages	143
■ SOLVING COMMON PROBLEMS	145
No Output	145
Error Recovery	146
■ REPORTING PROBLEMS	146
Chapter 4: Installation Procedures	147
■ PREPARATION	148
Obtaining Released Software	148
Hardware Mapping	148
■ EXECUTABLE FILES	148
■ INPUT FILES	149
EvLoc Parameter Files	149

Location Earth-model Files	149
Magnitude Earth-model Files	150
■ DATABASE	150
Accounts	151
Tables	151
Initialization of lastid	151
■ TUXEDO FILES	151
Configuring ubbconfig Template	152
Creating Queues	153
Creating Tuxshell Parameter Files	154
■ INITIATING OPERATIONS	155
■ VALIDATING INSTALLATION	155
References	157
Glossary	G1
Index	I1

Event Location and Magnitude Software User Manual

FIGURES

FIGURE 1.	IDC SOFTWARE CONFIGURATION HIERARCHY	3
FIGURE 2.	RELATIONSHIP OF THE EVENT LOCATION AND MAGNITUDE SOFTWARE TO IDC CSCs IN THE AUTOMATIC AND INTERACTIVE PROCESSING CSCs	5
FIGURE 3.	RELATIONSHIP OF EVLOC TO IDC OPERATIONAL SCRIPTS	7
FIGURE 4.	SAMPLE EVLOC PARAMETER FILE	31
FIGURE 5.	SAMPLE VMSF	37
FIGURE 6.	SAMPLE 1-D TRAVEL-TIME TABLE	40
FIGURE 7.	SAMPLE SSSC FILE	52
FIGURE 8.	SAMPLE LTSC FILE	55
FIGURE 9.	SAMPLE LPGI FILE	56
FIGURE 10.	SAMPLE LPPV FILE	59
FIGURE 11.	SAMPLE SASC FILE	62
FIGURE 12.	SAMPLE MDF	65
FIGURE 13.	SAMPLE TLSF	70
FIGURE 14.	SAMPLE TLM	77
FIGURE 15.	SAMPLE MTSC FILE	87
FIGURE 16.	SAMPLE TUXSHELL PARAMETER FILE	154

Event Location and Magnitude Software User Manual

TABLES

TABLE I:	DATA FLOW SYMBOLS	iv
TABLE II:	TYPOGRAPHICAL CONVENTIONS	v
TABLE 1:	OPERATIONAL IDC APPLICATIONS INTERFACING WITH LIBLOC AND LIBMAGNITUDE	6
TABLE 2:	OPERATIONAL IDC SCRIPTS INVOKING EVLOC	7
TABLE 3:	DATABASE TABLES USED BY EVLOC IN LOCATION MODE	12
TABLE 4:	DATABASE TABLES USED BY EVLOC IN MAGNITUDE MODE	13
TABLE 5:	LOCATION INPUT FILES	14
TABLE 6:	MAGNITUDE INPUT FILES	15
TABLE 7:	GENERAL EVLOC PARAMETERS	24
TABLE 8:	EVLOC VERBOSITY PARAMETERS	25
TABLE 9:	EVLOC LOCATION PARAMETERS	26
TABLE 10:	EVLOC MAGNITUDE PARAMETERS	29
TABLE 11:	FORMAT CODES	33
TABLE 12:	VMSF FORMAT	35
TABLE 13:	TRAVEL-TIME DATA FORMAT	39
TABLE 14:	SINGLE-VALUE TRAVEL-TIME MODELING ERROR DATA FORMAT	43
TABLE 15:	DISTANCE-DEPENDENT TRAVEL-TIME MODELING ERROR DATA FORMAT	44
TABLE 16:	DISTANCE/DEPTH-DEPENDENT TRAVEL-TIME MODELING ERROR DATA FORMAT	46
TABLE 17:	SSSC FILE FORMAT	51
TABLE 18:	LTSC FILE FORMAT	54
TABLE 19:	LPGI DATA FORMAT	56
TABLE 20:	LPPV DATA FORMAT	58
TABLE 21:	SASC FILE FORMAT	61
TABLE 22:	MAGNITUDE SPECIFICATION PARAMETERS	66

TABLE 23:	BULK MAGNITUDE STATION CORRECTION PARAMETERS	68
TABLE 24:	TLM PATHWAY PARAMETERS	71
TABLE 25:	DEFAULT TLM DESCRIPTION PARAMETERS	71
TABLE 26:	DEFAULT TLM PATHNAME EXAMPLES	73
TABLE 27:	STATION-SPECIFIC TLM DESCRIPTION PARAMETERS	74
TABLE 28:	STATION-SPECIFIC TLM PATHNAME EXAMPLES	76
TABLE 29:	TRANSMISSION LOSS DATA FORMAT	78
TABLE 30:	SINGLE-VALUE TRANSMISSION LOSS MODELING ERROR DATA FORMAT	81
TABLE 31:	DISTANCE-DEPENDENT TRANSMISSION LOSS MODELING ERROR DATA FORMAT	82
TABLE 32:	DISTANCE/DEPTH-DEPENDENT TRANSMISSION LOSS MODELING ERROR DATA FORMAT	83
TABLE 33:	MTSC FILE FORMAT	86

About this Document

This chapter describes the organization and content of the document and includes the following topics:

- Purpose
- Scope
- Audience
- Related Information
- Using this Document

About this Document

PURPOSE

This document describes how to use the Event Location and Magnitude software of the International Data Centre (IDC). The software is part of the Post-location Processing and Time-series Libraries computer software components (CSCs) of the Automatic Processing Computer Software Configuration Item (CSCI) and is identified as follows:

Title: Event Location and Magnitude

Abbreviations: *EvLoc*, *libloc*, and *libmagnitude*

SCOPE

The manual includes instructions for setting up the software, using its features, and basic troubleshooting. This document does not describe the software's design or requirements. These topics are described in sources cited in "Related Information."

AUDIENCE

This document is intended for the first-time or occasional user of the software. However, more experienced users may find certain sections useful as a reference.

RELATED INFORMATION

The following documents complement this document:

- *Event Location Software* [IDC7.1.5]
- *Event Magnitude Software* [IDC7.1.6]

- *New Travel-time Handling Facilities at the IDC; Functionality, Testing and Implementation Details* [Nag96]
- *IDC Processing of Seismic, Hydroacoustic, and Infrasonic Data* [IDC5.2.1Rev1]

See “References” on page 157 for a list of documents that supplement this document. The following UNIX manual (man) pages apply to the existing *EvLoc*, *libloc*, and *libmagnitude* software:

- *EvLoc*
- *libloc*
- *libmagnitude*

USING THIS DOCUMENT

This document is part of the overall documentation architecture for the IDC. It is part of the Technical Instructions category, which provides guidance for installing, operating, and maintaining the IDC systems. This document is organized as follows:

- Chapter 1: Introduction
This chapter provides an overview of the software’s capabilities, development, and operating environment.
- Chapter 2: Operational Procedures
This chapter describes how to use the software and includes detailed procedures for startup and shutdown, basic and advanced features, security, and maintenance.
- Chapter 3: Troubleshooting
This chapter describes how to identify and correct common problems related to the software.

▼ About this Document

- Chapter 4: Installation Procedures

This chapter describes first how to prepare for installing the software, then how to install the executable files, configuration data files, database elements, and any Tuxedo files. It also describes how to initiate operation and how to validate the installation.
- References

This section lists the sources cited in this document.
- Glossary

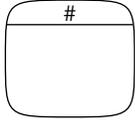
This section defines the terms, abbreviations, and acronyms used in this document.
- Index

This section lists topics and features provided in this document along with page numbers for reference.

Conventions

This document uses a variety of conventions, which are described in the following tables. Table I shows the conventions for data flow diagrams. Table II lists typographical conventions.

TABLE I: DATA FLOW SYMBOLS

Description	Symbol ¹
process	
data store (left), duplicated data store (right) Db = database store	
control flow	
data flow	

1. Most symbols in this table are based on Gane-Sarson conventions [Gan79].

TABLE II: TYPOGRAPHICAL CONVENTIONS

Element	Font	Example
database table	bold	event_control
database table and attribute, when written in the dot notation		origin.time
database attributes	<i>italics</i>	<i>delta</i>
processes, software units, and libraries		<i>EvLoc</i>
user-defined arguments and variables used in parameter (par) files or program command lines		<i>par=parfile</i>
titles of documents		<i>Event Magnitude Software</i>
variables used in error messages or filenames		<i><QUERY></i>
computer code and output	<code>courier</code>	<code>MDreadErr2: MDF incor- rectly formatted!</code>
filenames, directories, and websites		<code>iasp91.P</code>
text that should be typed exactly as shown		<code>edit-filter-dialog</code>
data types		<code>char</code>

Chapter 1: Introduction

This chapter provides a general description of the software and includes the following topics:

- Software Overview
- Status of Development
- Functionality
- Inventory
- Environment and States of Operation

Chapter 1: Introduction

SOFTWARE OVERVIEW

Figure 1 shows the logical organization of the IDC software. The Event Location and Magnitude software is part of the Post-location Processing and Time-series Libraries CSCs, which are part of the Automatic Processing CSCI.

The Event Location and Magnitude software is composed of three software units: *EvLoc*, *libloc*, and *libmagnitude*. *EvLoc* (Event Location and Magnitude) is an application in the Post-location Processing CSC that can be part of a processing pipeline or used as a standalone tool. *EvLoc* acquires data from an input database and input files, exchanges the data with functions in the *libloc* and *libmagnitude* software libraries, and stores the event location and magnitude results in an output database. In essence, *EvLoc* exchanges data between the input files, databases, and the critical processing units.

libloc and *libmagnitude* are common software libraries in the Time-series Libraries CSC. *libloc* provides software interfaces to read travel-time data from input files, compute event locations, and predict travel times. *libmagnitude* provides software interfaces to read transmission loss data from input files and estimate event magnitudes. *EvLoc* is based on *libloc* and *libmagnitude*.

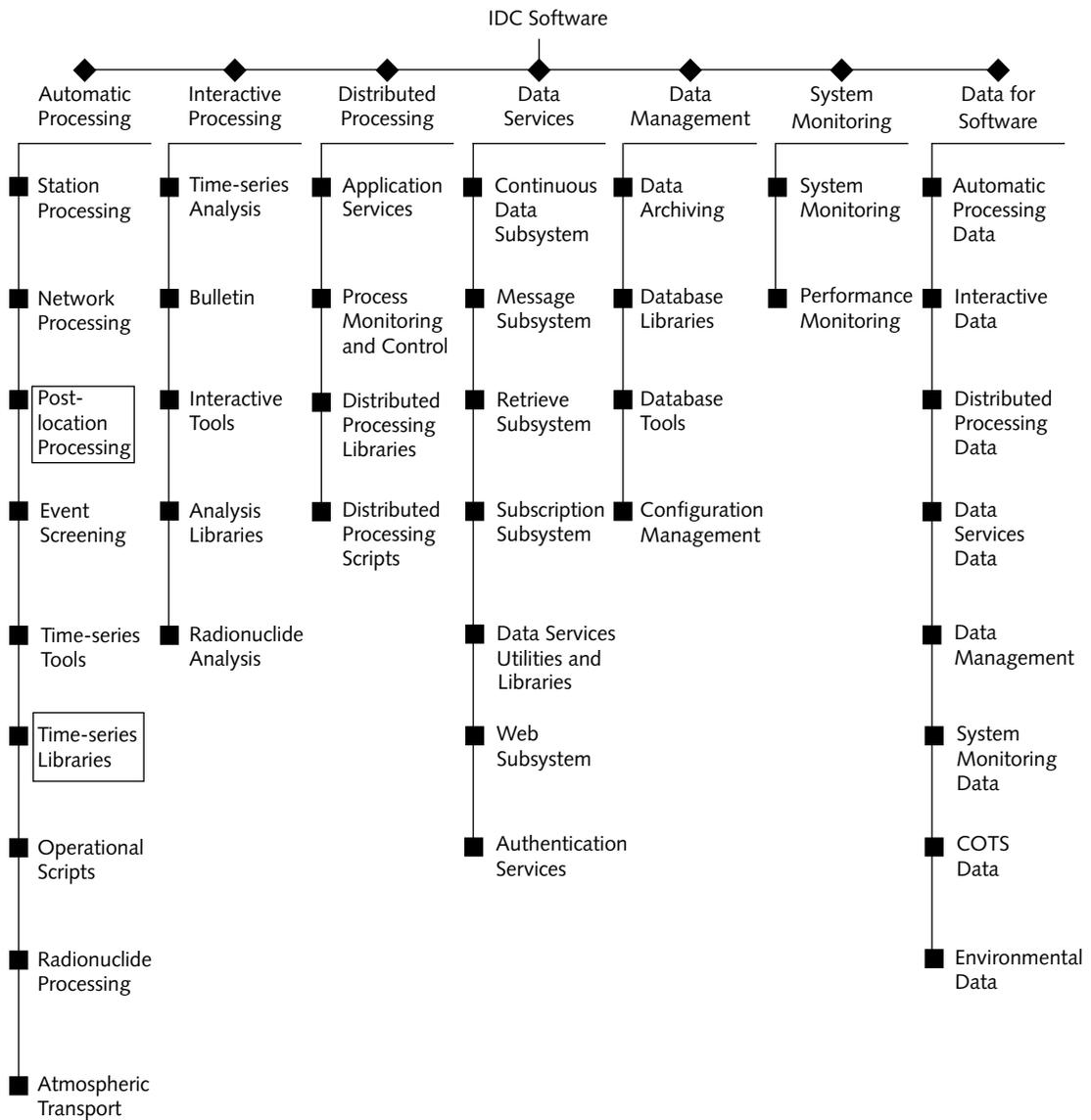


FIGURE 1. IDC SOFTWARE CONFIGURATION HIERARCHY

▼ Introduction

Figure 2 is a processing flow model of the relationship between the *libloc* and *libmagnitude* software libraries (processes 1 and 2, respectively) and other applications of the Automatic Processing and Interactive Processing CSCIs in the IDC operational system. Processes within the dashed box are part of the operational system whose execution is managed by a Distributed Application Control System (DACS) (not shown in Figure 2). IDC processing proceeds sequentially by process number (starting with process 3) from upper left to lower right in the figure. In addition to being managed by the DACS, all applications in Figure 2 exchange data with the operational database, as indicated by the line connecting the database to the dashed box. The *libloc* and *libmagnitude* libraries are outside of the dashed box because they are function archives. They are not managed by the DACS and do not interact with the operational database.

Processes 3–9 exchange information with *libloc* and/or *libmagnitude* functions to determine event locations and/or magnitudes. Event location processing occurs before magnitudes are computed in the processing flow because event locations are required to calculate magnitudes.

Table 1 identifies the applications associated with processes 3–9 in Figure 2. The values in the Process ID column correspond to process numbers used in Figure 2. Details of the data flow between *libloc*, *libmagnitude*, and their calling applications, are described in the *Event Location Software* and *Event Magnitude Software* design documents ([IDC7.1.5] and [IDC7.1.6], respectively).

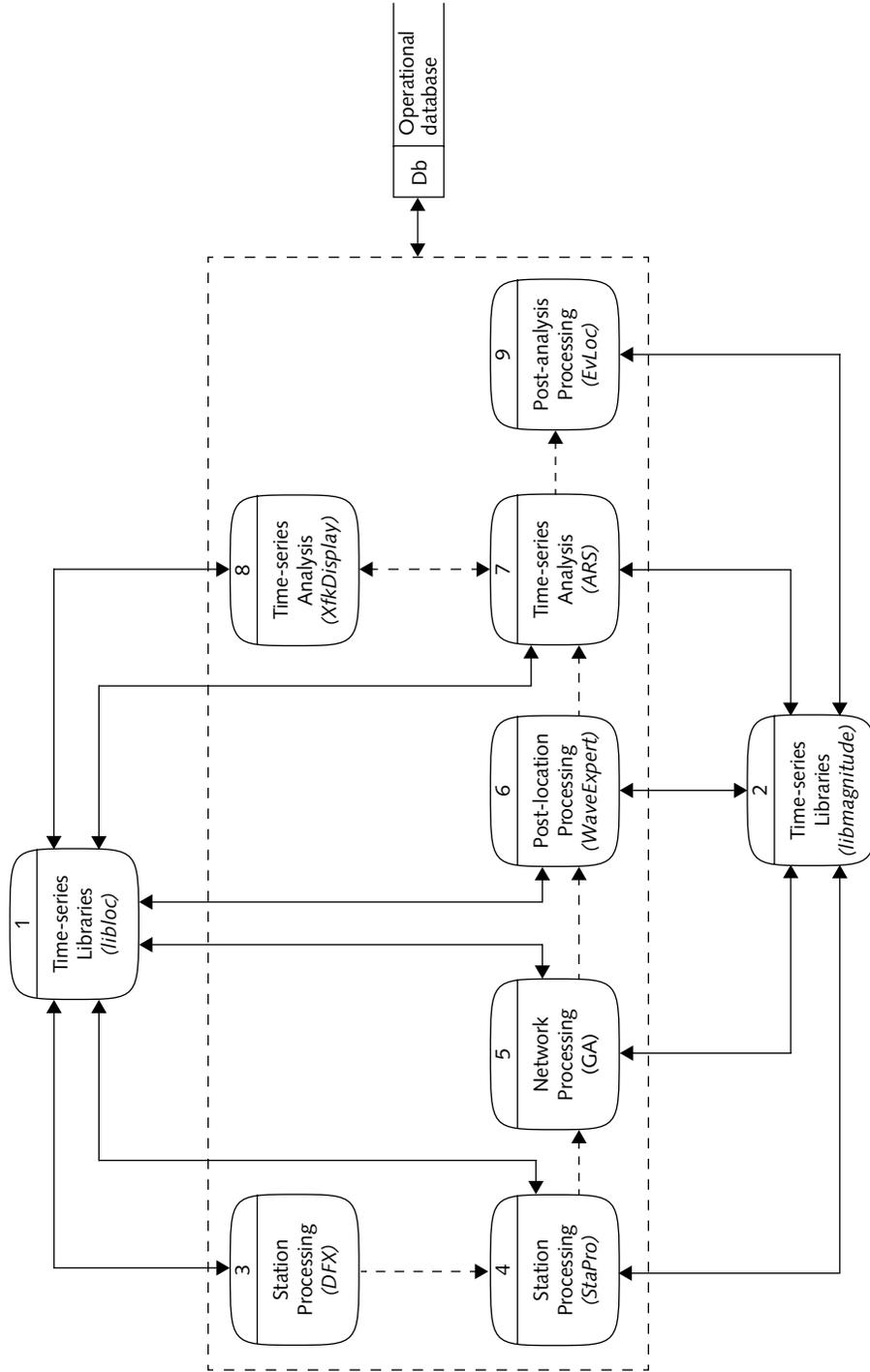


FIGURE 2. RELATIONSHIP OF THE EVENT LOCATION AND MAGNITUDE SOFTWARE TO IDC CSCS IN THE AUTOMATIC AND INTERACTIVE PROCESSING CSCIS

▼ Introduction

TABLE 1: OPERATIONAL IDC APPLICATIONS INTERFACING WITH LIBLOC AND LIBMAGNITUDE

Application	CSC	Libraries Used	Process ID
Detection and Feature Extraction (<i>DFX</i>)	Station Processing	<i>libloc</i>	3
Station Processing (<i>StaPro</i>)	Station Processing	<i>libloc</i> , <i>libmagnitude</i>	4
Global Association (GA) Subsystem	Network Processing	<i>libloc</i> , <i>libmagnitude</i>	5
<i>WaveExpert</i>	Post-location Processing	<i>libloc</i> , <i>libmagnitude</i>	6
Analyst Review Station (<i>ARS</i>)	Time-series Analysis	<i>libloc</i> , <i>libmagnitude</i>	7
F-K Analysis (<i>XfkDisplay</i>)	Time-series Analysis	<i>libloc</i>	8
Event Location and Magnitude (<i>EvLoc</i>)	Post-location Processing	<i>libmagnitude</i> ¹	9

1. *EvLoc* is operationally configured at the IDC to limit its functionality to magnitude estimation only. *EvLoc* is capable of using *libloc* functionality, but the IDC design does not utilize this capability at the operational level.

Figure 3 is a processing flow model of the relationship between *EvLoc* and two other components of the Automatic Processing CSCI in the IDC operational system. As in Figure 2, IDC processing proceeds sequentially by process number (starting with process 9). Only processes 9 and 10 are managed by the DACS.

Processes 10 and 11 in Figure 3 acquire arrival data from an operational database, modify arrival features, and write the modifications back to the database. Then Perl scripts composing these processes invoke *EvLoc* to recompute magnitudes based on the modifications. *EvLoc* extracts the adjusted data from the database, passes them to *libmagnitude*, and stores the returned magnitude results in the database.

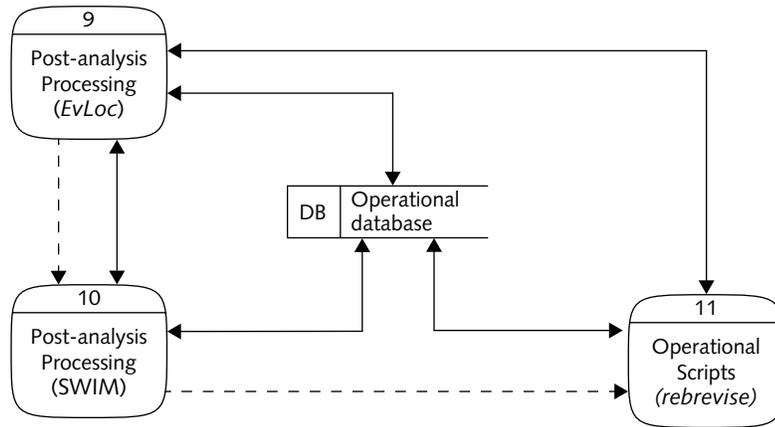


FIGURE 3. RELATIONSHIP OF EVLOC TO IDC OPERATIONAL SCRIPTS

Table 2 identifies the scripts that invoke *EvLoc*. The Software column identifies the software associated with the scripts (if any). The values in the Process ID column correspond to process numbers used in Figure 3.

TABLE 2: OPERATIONAL IDC SCRIPTS INVOKING EVLOC

Script	Software	CSC	Process ID
<i>MsOrid</i>	Surface Wave Identification and Measurement (SWIM)	Post-location Processing	10
<i>MsConflict</i>	SWIM	Post-location Processing	10
<i>rebrevise</i>	None	Operational Scripts	11

STATUS OF DEVELOPMENT

The Event Location and Magnitude software is mature and stable, although enhancements are made periodically. For example, Source-specific Station Corrections (SSSCs) were revised to include depth dependency in 2002 (see Table 5 on page 14). The version of the software described in this document is used at the Center for Monitoring Research (CMR) and is not available at the IDC.

FUNCTIONALITY

Features and Capabilities

EvLoc parses control parameter arguments from an input parameter file, reads station, event, detection, and optional control data from an input database account, interfaces with *libloc* and *libmagnitude* functions to compute event locations and magnitudes, and writes the resultant data to an output database account. *EvLoc* can be run in one of three functional modes:

- Location mode (compute event locations only).
- Magnitude mode (estimate event magnitudes only).
- Combined mode (determine both event locations and magnitudes).

EvLoc is only run in magnitude mode in the IDC operational system, including when it is invoked by SWIM software scripts and *rebrevis* (see “Normal Operational State” on page 16). Figure 3 shows the relationship between *EvLoc* and these scripts.

EvLoc offers the user significant flexibility regarding what and how data are exchanged with the database. The input event data are identified by a database query specified in the input parameter file. The input database can include optional location and magnitude processing control parameters. Results can be written to the input database tables or they can be written to a new set of output tables. To configure an input parameter file that utilizes one or more of these capabilities, see “Creating an *EvLoc* Parameter File” on page 22, particularly the “General Control Parameters” section.

The *libloc* software library provides functional interfaces for reading travel-time data from earth-model files, computing event locations with error estimates, and predicting travel times for a large range of seismic phases. *StaPro*, the GA Subsystem, and *ARS* use *libloc* functions to compute event locations. *EvLoc* can use *libloc* functionality, but is not configured to do so operationally at the IDC. *DFX*, *StaPro*, the GA Subsystem, *ARS*, *XfkDisplay*, and *WaveExpert* use *libloc* functions to predict travel times. All of these applications also interface with *libloc* functions to

read travel times from earth-model files. Figure 2 shows the relationship between these applications and *libloc*. See [IDC7.1.5] for additional details regarding *libloc* functionality used by these applications and the corresponding data flow.

The *libloc* software library uses a combination of one- and two-dimensional travel-time tables to compute event locations and can be configured to apply one or more of several available corrections to improve location accuracy and/or precision. Ellipticity, elevation, and bulk static corrections are automatically read and applied if available. Slowness/azimuth station corrections, based on empirical analyses of past events, are applied if available and requested. Two other mutually exclusive, region-specific travel-time corrections are available: source-specific station corrections and test-site corrections. When computing event locations, any of the four location parameters can be constrained, although latitude and longitude must be constrained as a pair. For each of the unconstrained parameters *libloc* propagates error values into an uncertainty estimate. Controls specifying the filesystem locations of the travel-time tables are stored in a configuration file. See “Constructing Location Earth-model Files” on page 34 for complete details.

The *libmagnitude* software library provides functional interfaces for reading transmission loss values from earth-model files and determining station and network magnitudes and uncertainties. *StaPro*, the GA Subsystem, and *WaveExpert* use *libmagnitude* functions to compute station magnitudes. *EvLoc* and *ARS* use *libmagnitude* functions to determine both station and network magnitudes. These applications also interface with *libmagnitude* functions to read transmission loss values from earth-model files. Figure 2 shows the relationship between these applications and *libmagnitude*. See [IDC7.1.6] for additional details regarding *libmagnitude* functionality used by these applications and the corresponding data flow.

The *libmagnitude* software library can apply a distance/depth-dependent transmission-loss correction, a bulk magnitude station correction, and/or a test-site correction to detection data (for example, amplitude, period, and duration) to compute a station magnitude. These corrections are read from the input files. *libmagnitude* can estimate standard normal, maximum-likelihood estimate (MLE), and bootstrap-resampled MLE network magnitudes. Controls specifying the filesystem loca-

▼ Introduction

tions of the transmission loss tables and how to compute magnitudes are stored in configuration files. See “Constructing Magnitude Earth-model Files” on page 64 for complete details.

Performance Characteristics

This section describes the performance characteristics of the Event Location and Magnitude software. Emphasis is given to *EvLoc* because it captures processing time information, handles error messages, and interfaces with the database.

The Event Location and Magnitude software is reliable and robust; it does not produce errors during run time unless there is a problem with data in the input database or the configuration of the input files.

The primary factor limiting the number of origins that can be processed during a single *EvLoc* invocation is the amount of available memory on the operational machine (see “Hardware Environment” on page 16). A secondary limitation is controlled by the input parameter `max_gdi_records`, which caps the number of records that can be read from the input database account during run time.

The remaining performance characteristics depend on the functional mode of the software and are provided for each functional mode in the remainder of this section. All timing statistics were calculated from runs on a Sun Ultra 60 computer.

Location Mode

When *EvLoc* operates in location mode, it reads data from six or seven input database tables and interfaces with *libloc* to read tens to hundreds of files (depending on configuration). *EvLoc* writes to three, four, or five output database tables (depending on configuration). It does not write any output files, unless standard out is redirected to a log file.

The single most time consuming part of *EvLoc* processing is reading the earth-model files associated with location mode, which occurs once at the beginning of a run. It takes approximately 10 s to read the earth-model files, most of which is spent reading the SSSC tables (page 49). Individual location computations only

take about 0.05 s. Accordingly, the time for a given *EvLoc* run in location mode can be estimated as 10 s plus an additional second for every 20 events processed. Reading the SSSC files can be suppressed if faster *EvLoc* run times are required. See “Velocity Model Specification File” on page 35 for details.

Magnitude Mode

When the *EvLoc* software operates in magnitude mode, it reads data from eight or nine input database tables and interfaces with *libmagnitude* to read three to tens of files (depending on configuration). *EvLoc* writes to two, three, or four output database tables (depending on configuration). It does not write any output files, unless standard out is redirected to a log file.

Reading the earth-model files associated with magnitude mode takes on the order of 1.0 s, which is done once at the beginning of the run. A single magnitude computation takes about 0.01 s. Therefore, the time for an *EvLoc* run in magnitude mode can be estimated as 1.0 seconds plus an additional 0.1 s for every 10 magnitude values computed. Because more than one magnitude type can be computed for each event, the product of the number of magnitude types and the number of events is used to estimate the run-time.

Combined Mode

The performance of *EvLoc* when it operates in combined mode (that is, both location and magnitude modes) is an aggregate of the performance of the individual modes.

INVENTORY

This section lists the software, database tables, and data files used by the Event Location and Magnitude software.

▼ Introduction

Software

EvLoc is the only compiled binary in the Event Location and Magnitude software. *libloc* and *libmagnitude* are both shared libraries.

When built, *EvLoc* links to the following shared libraries: *libgdi*, *libloc*, *libmagnitude*, *libinterp*, *libgeog*, *libLP*, *libstdtime*, *libdb30qa*, *libpar*, and *libaesir*. *EvLoc* also links to the following commercial off-the-shelf (COTS) libraries: *libF77*, *libtermcap*, *libsocket*, *libnsl*, *libelf*, *libm*, *libdl*, and *libsunmath*.

Database Tables

EvLoc reads input station, event, and control parameter data from database tables in an input database account and writes resultant event locations and magnitudes to an output database account. The input and output accounts can be the same. In the IDC operational system, the Late Event Bulletin (LEB) account is both the input and output account. *libloc* and *libmagnitude* do not directly interface with the database.

The functional mode dictates the database tables that are required.

Location Mode Database Tables

Table 3 lists the database tables used during an *EvLoc* run in location mode. The Name column identifies the database table. The Mode column is "R" if *EvLoc* reads from the table and "W" if *EvLoc* writes to the table.

TABLE 3: DATABASE TABLES USED BY EVLOC IN LOCATION MODE

Name	Mode	Description
affiliation	R	Station network information.
site	R	Station location information.
origin	R/W	Origin information for a particular event.
origerr	R/W	Error-estimate information for an origin.

**TABLE 3: DATABASE TABLES USED BY EVLOC
IN LOCATION MODE (CONTINUED)**

Name	Mode	Description
assoc	R/W	Arrival association information.
arrival	R	Event-independent arrival information.
event_control	R/W	Event location and magnitude control parameters (optional).
ar_info	W	Ancillary arrival-based information.

Magnitude Mode Database Tables

Table 4 lists the database tables used during an *EvLoc* run in magnitude mode. The column descriptions are the same as those for Table 3.

**TABLE 4: DATABASE TABLES USED BY EVLOC
IN MAGNITUDE MODE**

Name	Mode	Description
affiliation	R	Station network information.
site	R	Station location information.
origin	R/W	Origin information for a particular event (optional write).
assoc	R	Arrival association information.
parrival	R	Predicted arrivals and associations for origin-based amplitude measurements.
amplitude	R	Amplitude measurements.
stamag	R/W	Station magnitude determinations.
netmag	R/W	Network magnitude estimates.
event_control	R/W	Event location and magnitude control parameters (optional).

▼ Introduction

Input Files

At run time, the Event Location and Magnitude software reads a suite of input files. Input files are composed of control parameter files, configuration files, and earth-model files. Control parameter files contain adjustable *EvLoc* control parameter arguments. Configuration files store adjustable parameters accessed at run time. Earth-model files store empirical or theoretical models of physical earth processes. *EvLoc* only reads par files, and *libloc* and *libmagnitude* only read configuration and earth-model files.

The functional mode dictates which input files are read. The set of input files needed to locate events is disjoint from those needed to estimate magnitudes, although a par file is used in both cases to conveniently store and specify parameter arguments. “Creating an *EvLoc* Parameter File” on page 22 describes the format and content of par files.

Location Input Files

Table 5 lists the configuration and earth-model files read by the *libloc* software. “Constructing Location Earth-model Files” on page 34 describes the format and content of these files.

TABLE 5: LOCATION INPUT FILES

Name	Type	Description
Velocity Model Specification File (VMSF)	Configuration/ Earth Model	Stores travel-time earth-model files (configuration) and station/phase travel-time correction data (earth model).
One-dimensional Travel-time (TT) Table	Earth Model	Stores one-dimensional travel-time data for a seismic, hydroacoustic, or infrasonic (S/H/I) phase.
Radial Two-dimensional Travel-time Table	Earth Model	Stores two-dimensional travel-time data for a hydroacoustic or infrasonic phase.
Ellipticity Correction (EC) Table	Earth Model	Stores ellipticity travel-time correction data for a phase.

TABLE 5: LOCATION INPUT FILES (CONTINUED)

Name	Type	Description
Source-specific Station Correction (SSSC) Table	Earth Model	Stores 2-D regional travel-time corrections relative to a particular 1-D TT table for a station/phase pair.
Location Test-site Correction (LTSC) File	Earth Model	Stores travel-time correction data for station/phase pairs in a defined geographic region.
Long-period Grid Index (LPGI) File	Earth Model	Stores a global grid of phase velocity table indices for an LP phase.
Long-period Phase Velocity (LPPV) File	Earth Model	Stores phase-velocity tables for an LP phase.
Slowness/Azimuth Station Correction (SASC) Table	Earth Model	Stores slowness and azimuth corrections for a station.

Magnitude Input Files

Table 6 lists the configuration and earth-model files read by the *libmagnitude* software. “Constructing Magnitude Earth-model Files” on page 64 describes the format and content of these files.

TABLE 6: MAGNITUDE INPUT FILES

Name	Type	Description
Magnitude Description File (MDF)	Configuration/ Earth Model	Stores magnitude specification parameters (configuration) and bulk magnitude station corrections (earth model).
Transmission Loss Specification File (TLSF)	Configuration	Stores transmission loss models.
Transmission Loss Model (TLM)	Earth Model	Stores transmission loss values.
Magnitude Test-site Correction (MTSC) File	Earth Model	Stores magnitude corrections for a defined geographic region.

ENVIRONMENT AND STATES OF OPERATION

This section describes the hardware and software environments necessary to run *EvLoc*, its normal operational state, and an alternate operational state. Refer to the software user manuals of other applications that link to the *libloc* and *libmagnitude* software libraries for information about their environments and operational states (see Tables 1 and 2).

Hardware Environment

EvLoc is designed to run on a UNIX workstation with at least 256 MB of Random Access Memory (RAM) and a minimum of 2 GB of magnetic disk. However, better performance is realizable if *EvLoc* is invoked on workstations with 512 MB or more of RAM. This will avoid memory swapping, which becomes a problem when large files such as source-specific station correction files and travel-time tables are read into memory during run time.

Software Environment

EvLoc was built and tested using the Sun Microsystems' Solaris 7 operating system. It interfaces with an Oracle 8i database server to exchange event locations and magnitudes. TUXEDO interprocess communication software manages *EvLoc* processing in the IDC operational Post-location pipeline.

Normal Operational State

In normal IDC operations, *EvLoc* is automatically invoked in magnitude mode eight times in the Post-analysis Processing pipeline and up to five times by the *rebrevise* script for each event processed.

The Post-analysis Processing pipeline is controlled by the DACS. Four of the eight *EvLoc* instances within the pipeline are child processes managed by the *tuxshell* generalized processing server [IDC6.5.2Rev0.1]. These *EvLoc* processes estimate M_L , m_b network-average, m_b MLE, m_{b1} network-average, and m_{b1} MLE magnitudes (the latter two are estimated by the same child process). The four remaining

EvLoc instances are invoked by the *MsOrid* and *MsConflict* scripts of the SWIM software during Post-analysis Processing [IDC6.5.16]. These *EvLoc* runs estimate M_s network-average, M_s MLE, M_{s1} network-average, and M_{s1} MLE magnitudes.

rebrevis is used to correct problems discovered in the Reviewed Event Bulletin (REB) before publication. Recalculation functions within *rebrevis* invoke *EvLoc* to determine new m_b network-average, m_b MLE, M_L , M_s network-average, and M_s MLE magnitude estimates for a specified event.

Contingencies/Alternate States of Operation

EvLoc can be run independently of a processing pipeline by direct execution from a UNIX command line with appropriate parameter arguments. This alternate operational state is useful for:

- off-line testing of new or improved earth-model data or algorithms
- unit and integration testing
- tuning
- scientific research

Additionally, the Hypocenter Location Server (HLS) provides an alternate method of accessing the command-line instance of *EvLoc*. HLS allows users to run *EvLoc* remotely over the Internet on locally archived data. All of *EvLoc*'s functionality is available via the HLS interface, except Maximum Likelihood magnitude computation. [Kun02] provides more details on HLS.

Chapter 2: Operational Procedures

This chapter provides instructions for using the software and includes the following topics:

- Software Startup
- Software Shutdown
- Control Parameter Setup
- Earth-model File Setup
- Maintenance
- Security

Chapter 2: Operational Procedures

SOFTWARE STARTUP

This section describes how to invoke *EvLoc* both in an operational system and manually from a UNIX command line. The IDC operational system is used to illustrate an operational startup. Startup of *libloc* and *libmagnitude* is not possible because they are software libraries accessed via applications. Other than *EvLoc*, the applications that link to these software libraries are *DFX*, *StaPro*, the GA Subsystem, *WaveExpert*, *ARS*, and *XfkDisplay*. For descriptions of how to start these applications, refer to the *Analyst Instructions for Seismic, Hydroacoustic, and Infrasonic Data* [IDC6.2.5], *Station Processing (StaPro) Software User Manual* [IDC6.5.11], and *Global Association (GA) Subsystem Software User Manual* [IDC6.5.12].

Normal Operational Startup

In a normal IDC operational environment, *EvLoc* is invoked within the Post-analysis Processing pipeline, the SWIM software, and the *rebrevis* script. In the Post-analysis Processing pipeline, each *EvLoc* child process (there are four in the IDC configuration) is managed by a *tuxshell* application. The first *EvLoc* child process is invoked after *tuxshell* receives a "success" termination code from the preceding *DFX* child process in the pipeline. *Tuxshell* assembles an *EvLoc* command line and submits it to the operating system. When this *EvLoc* child process completes, it normally returns a success code to *tuxshell*, which then assembles a command line for the next *EvLoc* child process and submits that to the operating system. Processing continues in this fashion until all *EvLoc* child processes have successfully completed. See "Tuxedo Files" on page 151 for additional details. Information about *tuxshell*, the various services provided by the DACS, and the Post-analysis Processing pipeline may be found in [IDC6.5.2Rev0.1].

The *MsOrid* and *MsConflict* Perl scripts within the SWIM software also invoke *EvLoc*. *MsOrid*, whose processing is managed by *tuxshell* in the Post-analysis Processing pipeline, assembles one or more *EvLoc* command lines, submits them to the operating system, and interprets their return codes. The *EvLoc* child processes (there are two in the IDC configuration) are submitted sequentially, so a previous child process must complete before a subsequent child process is initiated. *MsConflict*, which is not managed by *tuxshell*, and may only be invoked by a command line submitted to an operating system, manages *EvLoc* child processes (there are two in the IDC configuration) the same way as *MsOrid*. Additional details about the relationship of *MsOrid* and *MsConflict* to *EvLoc* may be found in the “Functional Description” section of [IDC7.1.3].

Finally, the *rebrevise* Perl script invokes *EvLoc*. *rebrevise* is started manually by a command line submitted to the operating system by a senior analyst. When prompted, *rebrevise* manages *EvLoc* children (there are up to five possible in the IDC configuration) the same way as *MsOrid*, except that the return codes are not interpreted—control simply passes to the next command in the *rebrevise* script.

Manual Startup

EvLoc may be manually invoked by entering the following command line text:

```
EvLoc par=parfile
```

where *parfile* is a standard format par file containing *EvLoc* parameter arguments and optional nested par files. A useful addition to this command line is a redirect of standard out to a logfile (>& logfile). Further, command line overrides of any parameter argument can be achieved using the syntax:

```
EvLoc par=parfile arg1=arg arg2=arg ...
```

SOFTWARE SHUTDOWN

This section describes how to shut down *EvLoc* both automatically and manually from a UNIX command line. Shutdown of *libloc* and *libmagnitude* is not discussed since they are software libraries, and termination of the calling application will ter-

▼ Operational Procedures

minate processing within these libraries. For descriptions of how to shut down applications that link to *libloc* and *libmagnitude*, refer to their respective software user manuals.

Automatic Application Shutdown

EvLoc automatically terminates after processing all events. However, if an *EvLoc* child process invoked by *tuxshell* must be terminated, then use the DACS TUXEDO system to do so by following the shutdown procedures described in [IDC6.5.2Rev0.1].

Manual (Shell) Shutdown

A manually invoked *EvLoc* process may be terminated using the standard UNIX `kill` command.

CONTROL PARAMETER SETUP

Control parameters determine the application run-time behavior. This section describes the control parameters read by *EvLoc*. Control parameters are normally grouped together in a par file named on the command line, but can also be specified individually on the command line. The following sections describe the format of the par file

Creating an EvLoc Parameter File

An *EvLoc* par file contains control parameter arguments necessary for *EvLoc* processing. Parameters are specified in the form:

parameter=value

where *value* is the desired parameter value. Specify one parameter per line in the par file.

An *EvLoc* par file can also be configured so that system-level parameters are read prior to parsing the *EvLoc* parameters. In the *EvLoc* par file, include the following line before any *EvLoc* parameters are specified:

```
par=sysparfile
```

where *sysparfile* is the pathname of the nested par file containing the system-level parameters. This configuration is useful in an operational system to define environment variables, database accounts, and configuration file pathnames used by multiple applications. The example par file in Figure 4 on page 31 shows an *EvLoc* par file that references a system-level par file.

Four groups of control parameters can be specified in an *EvLoc* par file:

- general
- verbosity
- location
- magnitude

The remainder of this subsection classifies each *EvLoc* parameter into one of these four categories. Parameter definitions, data types, default values, and associated functionality are not static and may change over time. Refer to the *EvLoc* man page for up-to-date descriptions of the parameters rather than completely relying on this section.

General Control Parameters

General *EvLoc* control parameters can be specified regardless of functional mode. These parameters include input and output database accounts, database table data required by both location and magnitude processing, and station network. Table 7 shows the general *EvLoc* parameters. Parameters noted as required in this section are required in all operational modes.

▼ Operational Procedures

TABLE 7: GENERAL EVLOC PARAMETERS

Name	Data Type	Description	Default Value
<i>mode</i>	int	Functional mode setting: 0 = compute event locations only 1 = compute event magnitudes only 2 = compute both event locations and magnitudes	2
<i>db_vendor</i>	char	Database vendor (required).	NULL
<i>in_db_account</i>	char	Input database account name and password (required).	NULL
<i>out_db_account</i>	char	Output database account name and password (required).	NULL
<i>affiliation_table</i>	char	Input affiliation table name (required).	NULL
<i>site_table</i>	char	Input site table name (required).	NULL
<i>origin_table</i>	char	Input origin table name (required).	NULL
<i>origin_query</i>	char	Database query used to retrieve data from input origin table (required).	NULL
<i>assoc_table</i>	char	Input assoc table name (required).	NULL
<i>write_to_input_db_tables</i>	Bool	Write processing results to input database tables? Otherwise write to new tables.	FALSE
<i>new_origin_table</i>	char	Output origin table name (required if <i>write_to_input_db_tables</i> is FALSE).	NULL
<i>use_ev_cntrl_table</i>	Bool	Override control parameters with control settings in input event_control table?	FALSE
<i>event_control_table</i>	char	Input event_control table name (required if <i>use_ev_cntrl_table</i> is TRUE).	NULL
<i>write_ev_cntrl_table</i>	Bool	Write control settings to output event_control table?	FALSE

TABLE 7: GENERAL EVLOC PARAMETERS (CONTINUED)

Name	Data Type	Description	Default Value
<i>new_event_control_table</i>	char	Output event_control table name (required if <i>write_ev_cntrl_table</i> is TRUE).	NULL
<i>network</i>	char	Unique station network identifier (required).	NULL
<i>max_gdi_records</i>	int	Maximum number of records read from or written to database account.	30000
<i>triple_location</i>	Bool	Attempt to estimate event location and magnitude for three depth types (free, surface, and restrained)?	FALSE

Verbosity Control Parameters

The verbosity level parameters define the amount of processing result detail to be written to standard out during execution. Although much of the information written in the various verbose levels is also written to the database, the format of the verbose output provides context when looking for the cause of unexpected or suspect results. When the verbosity parameters are set to 0, some warning and error reporting is suppressed, but other levels only affect the output of processing results. Table 8 shows the *EvLoc* verbosity parameters.

TABLE 8: EVLOC VERBOSITY PARAMETERS

Name	Data Type	Description	Default Value
<i>verbose</i>	int	Verbosity level for locator output. Allowed range = 0–4 (less–more)	2
<i>mag_verbose</i>	int	Verbosity level for magnitude output. Allowed range = 0–2 (less–more)	0

▼ Operational Procedures

Location Control Parameters

EvLoc location control parameters control only event location processing and have no effect on event magnitude processing. These parameters include location-specific database table names, phase lists, configuration file names, travel-time correction controls, hypocenter constraints, outlier screening controls, error computation controls, and station subsets. Table 9 shows the *EvLoc* location parameters. Parameters labeled as 'required' in the description column are required in location modes.

TABLE 9: EVLOC LOCATION PARAMETERS

Name	Data Type	Description	Default Value
<i>origerr_table</i>	char	Input origerr table name (required).	NULL
<i>arrival_table</i>	char	Input arrival table name (required).	NULL
<i>new_origerr_table</i>	char	Output origerr table name (required if <i>write_to_input_db_tables</i> is FALSE).	NULL
<i>new_assoc_table</i>	char	Output assoc table name (required if <i>write_to_input_db_tables</i> is FALSE).	NULL
<i>list_of_phases</i>	char	Phases used to compute event locations (required).	NULL
<i>vmodel_spec_file</i>	char	VMSF pathname (required).	NULL
<i>sasc_dir_prefix</i>	char	SASC directory pathway and common filename prefix.	NULL
<i>write_ar_info_table</i>	Bool	Write ancillary arrival data to optional ar_info table?	FALSE
<i>ar_info_table</i>	char	ar_info output table name (required if <i>write_ar_info_table</i> is TRUE).	NULL
<i>ellip_cor_level</i>	int	Ellipticity correction type: 0 = none 1 = not used in IDC processing 2 = IASPEI 1991	2

TABLE 9: EVLOC LOCATION PARAMETERS (CONTINUED)

Name	Data Type	Description	Default Value
<i>sssc_level</i>	int	SSSC type: 0 = none 1 = regional 2 = local (not to be used)	0
<i>only_sta_w_corr</i>	Bool	Compute event locations using only data from stations possessing SSSCs or test-site corrections?	FALSE
<i>fix_ot</i>	Bool	Constrain the origin time to equal the input <i>origin.time</i> value?	FALSE
<i>fix_latlon</i>	Bool	Constrain the epicenter to equal the input <i>origin.lat</i> and <i>origin.lon</i> values?	FALSE
<i>fix_depth</i>	Bool	Constrain the origin depth to equal <i>depth_init</i> ?	TRUE
<i>depth_init</i>	double	Constrained origin depth, km (required if <i>fix_depth</i> is TRUE).	-999.0
<i>big_res_mult</i>	double	Time, azimuth, and slowness residual outlier screening multiplier.	3.0
<i>ignore_big_res</i>	Bool	Ignore data whose residuals are <i>big_res_mult</i> times greater than the standard error?	FALSE
<i>dist_var_wgt</i>	Bool	Apply <i>a priori</i> distance/depth modeling errors?	FALSE
<i>conf_level</i>	double	<i>F</i> -statistics confidence level. Must be 0.80, 0.90, 0.95, or 0.99.	0.90
<i>damp</i>	double	Percent damping applied to diagonal elements of sensitivity matrix. If default value set, then locator will determine optimal damping.	-1.0
<i>num_dof</i>	int	<i>A priori</i> number of degrees of freedom for computing error ellipse. Common settings: 0 = confidence ellipse 99999 = coverage ellipse	0

▼ Operational Procedures

TABLE 9: EVLOC LOCATION PARAMETERS (CONTINUED)

Name	Data Type	Description	Default Value
<i>max_iter</i>	int	Maximum number of iterations allowed in location loop.	20
<i>use_tscor</i> ¹	Bool	Apply location test-site corrections?	FALSE
<i>ts_region</i> ¹	char	Location test-site region name.	NULL
<i>sub_sta_list_only</i>	Bool	Compute event locations using only data from subset of stations?	FALSE
<i>sub_sta_list</i>	char	List of stations whose data can be used to compute event locations (required if <i>sub_sta_list_only</i> is TRUE).	NULL

1. This parameter is only applicable to CMR systems and is not used at the IDC.

Magnitude Control Parameters

EvLoc magnitude control parameters control only event magnitude processing and have no effect on event location processing. These parameters include magnitude-specific database table names, configuration file names, magnitude types (*mag-types*), magnitude estimation controls, outlier screening controls, and station subsets. Table 10 shows the *EvLoc* magnitude parameters. Parameters listed as 'required' in the description column are required in magnitude mode.

Example Parameter File

Figure 4 shows a listing of a sample *EvLoc* par file. Nested par file references are used at the top of the file $\$(IMSPAR)$ and $\$(AUTOMATIC)$. For nested par files to be used in this manner, the file name variables must be defined in the run-time environment or on the command line. The “#” character is a comment identifier. The standard functional mode at the IDC is magnitude-only, in which case no location parameters need be specified. In this example, some location parameters appear at the end of the file.

TABLE 10: EVLOC MAGNITUDE PARAMETERS

Name	Data Type	Description	Default Value
<i>parrival_table</i>	char	Input parrival table name (required).	NULL
<i>det_amplitude_table</i>	char	Input amplitude table name. Table contains arrival-based amplitude data. May be the same table as <i>ev_amplitude_table</i> (required).	NULL
<i>ev_amplitude_table</i>	char	Input amplitude table name. Table contains origin-based amplitude data. May be the same table as <i>det_amplitude_table</i> (required).	NULL
<i>stamag_table</i>	char	Input stamag table name (required).	NULL
<i>netmag_table</i>	char	Input netmag table name (required).	NULL
<i>new_stamag_table</i>	char	Output stamag table name (required if <i>write_to_input_db_tables</i> is FALSE).	NULL
<i>new_netmag_table</i>	char	Output netmag table name (required if <i>write_to_input_db_tables</i> is FALSE).	NULL
<i>mag_descrip_file</i>	char	MDF pathname (required).	NULL
<i>tl_spec_file</i>	char	TLSF pathname (required).	NULL
<i>list_of_magtypes</i>	char	Magnitude types to determine.	NULL
<i>list_of_mb_magtypes</i>	char	m_b magnitude types to estimate. The magnitude types must be a subset of <i>list_of_magtypes</i> .	NULL
<i>list_of_magtypes_to_timedef_restrict</i>	char	Magnitude types whose station magnitudes can be magnitude-defining only if the associated arrivals are time-defining.	NULL
<i>magtype_to_origin_mb</i>	char	m_b magnitude type whose event magnitudes are written to output origin.mb and <i>mbid</i> fields.	NULL

▼ Operational Procedures

TABLE 10: EVLOC MAGNITUDE PARAMETERS (CONTINUED)

Name	Data Type	Description	Default Value
<i>magtype_to_origin_ms</i>	char	M_s magnitude type whose event magnitudes are written to output origin.ms and <i>msid</i> fields.	NULL
<i>magtype_to_origin_ml</i>	char	M_L magnitude type whose event magnitudes are written to output origin.ml and <i>mlid</i> fields.	NULL
<i>use_prev_magdefs</i>	Bool	Use magnitude-defining states of input station magnitudes?	FALSE
<i>mag_num_boots</i>	int	Maximum number of bootstrap resamples.	20
<i>mag_compute_upper_bounds</i>	Bool	Write upper bound magnitudes to output tables?	FALSE
<i>mag_big_res_mult</i>	double	Station magnitude outlier screening residual multiplier.	3.0
<i>mag_ignore_big_res</i>	Bool	Ignore data whose residuals are <i>mag_big_res_mult</i> times greater than the standard error?	FALSE
<i>mag_sub_sta_list_only</i>	Bool	Compute event magnitudes using only data from subset of stations?	FALSE
<i>mag_sub_sta_list</i>	char	List of stations whose data may be used to compute event magnitudes (required if <i>mag_sub_sta_list_only</i> is TRUE).	NULL
<i>mag_use_tscor</i> ¹	Bool	Apply magnitude test-site corrections?	FALSE
<i>mag_ts_region</i> ¹	char	Magnitude test-site region name (required if <i>mag_use_tscor</i> is TRUE).	NULL
<i>mag_only_sta_w_corr</i>	Bool	If <i>mag_use_tscor</i> is TRUE, use only stations with magnitude test-site corrections to estimate network magnitude values?	FALSE

1. This parameter is only applicable to CMR systems and is not used at the IDC.

```

#
# EvLoc shell script following libpar conventions.  When available,
# internal EvLoc defaults are defined in braces [ ]
#

par=$(IMSPAR)
par=$(AUTOMATIC)

# --- Primary control parameters ---

mode=2                # Control mode [2]
                      # 0: Event location only
                      # 1: Magnitude only
                      # 2: Both location and magnitude (default)
#write_to_input_db_tables # Write to input DB tables?[FALSE]
                      # TRUE: Write to existing DB tables
                      # FALSE: Write to new DB tables (default)
#triple_location      # Calculate 3 locations? [FALSE]
                      # (1 restrained; 1 free depth; and 1 fixed
                      # depth at Earth's surface).  This control
                      # parameter is typically only used for
                      # archiving purposes.
#use_ev_cntrl_table   # Use input event_control information,
                      # if it exists?[FALSE]
write_ar_info_table   # Write output ar_info table.[FALSE]
write_ev_cntrl_table  # Write output control settings to
                      # event_control table.[FALSE]

# --- Specify these required parameters regardless of mode ---

#origin_query="where time between $(start-time) and $(end-time)"
origin_query="where time between 896486500.000 and 896504500.000"
#origin_query="where orid in (1447185, 1459600)"
in_db_account=$(IN_DB)
out_db_account=$(OUT_DB)
db_vendor=oracle
network=CUR_PRI

# --- Specify input database tables here ---

site_table=site
affiliation_table=affiliation
origin_table=in_origin
assoc_table=in_assoc
arrival_table=arrival

# --- If write_to_input_db_tables is FALSE,
# then you must specify following ---

new_origin_table=evloc_origin
new_assoc_table=evloc_assoc
new_origerr_table=evloc_origerr
new_netmag_table=evloc_netmag
new_stamag_table=evloc_stamag

```

FIGURE 4. SAMPLE EVLOC PARAMETER FILE

▼ Operational Procedures

```

# --- If use_ev_cntrl_table is TRUE,
#     specify the following 2 input DB tables ---

#event_control_table=in_event_control
#origerr_table=in_origerr
new_event_control_table=evloc_event_control

# --- If write_ar_info_table is TRUE,
#     you must specify the output table name ---

ar_info_table=evloc_ar_info

# --- Event magnitude control parameters ---

list_of_magtypes="mb_ave,mb_mle,ms_ave,mlppn"
list_of_magtypes_to_timedef_restrict="mb_ave"
mag_verbose=1                # Level of verbose magnitude output to be
                             # printed: 0=None; 1=Network info only;
                             # 2=Network and station mag info[0]
mag_compute_upper_bounds    # Compute/save upper-bound magnitudes
                             # [FALSE]
magtype_to_origin_mb="mb_ave" # If successful network mag computed
                             # for this magtype, write to
                             # origin.mb/mbid ["mb"]
magtype_to_origin_ms="ms"    # If successful network magnitude
                             # computed for this magtype, write to
                             # origin.ms/msid ["ms"]
magtype_to_origin_ml="mlppn" # If successful network magnitude
                             # computed for this magtype, write to
                             # origin.ml/mlid ["ml"]
mag_num_boots=20            # Number of bootstrap re-samples
                             # for MLE [0]
mag_ignore_big_res         # Do not include individual station mags
                             # where its residual > mag_big_res_mult *
                             # abs(net avg mag - sta mag)[FALSE]
mag_big_res_mult=3.0       # Large magnitude residual factor[3.0]
mag_only_sta_w_corr        # Use only amplitude data with valid
                             # test-site corrections[FALSE]

# --- Event location control parameters ---

vmodel_spec_file=$(TT_CONFIG)/vmsf/ims.defs
list_of_phases="P,PP,PKP,PKPab,PKPbc,PKPpdf,PcP,Pn,Pg,Sn,Lg"
verbose=3                  # Level of verbosity for printed locator output.
                             # Scaled from 0 (no printed output)
                             # to 4 (all output printed).[1]
#fix_depth=1               # Fix (constrain) event hypocentral depth [1=TRUE]
#depth_init=0.0           # Fixed (constrained) event hypocentral depth (km)
                             #[0.0]

```

FIGURE 4. (CONTINUED)

Obtaining Help

Help for using this software is provided in this user's manual, the *Event Location Software* and *Event Magnitude Software* design documents ([IDC7.1.5] and [IDC7.1.6], respectively), through the man pages, and in the document *Processing of Seismic, Hydroacoustic, and Infrasonic Data* [IDC5.2.1Rev1].

EARTH-MODEL FILE SETUP

This section describes the content and format of the configuration and earth-model files necessary for computing event locations and estimating event magnitudes. The files described in this section are grouped according to functional mode. The first subsection describes the files used for locating events. The second subsection describes the files used for estimating magnitudes.

File format tables define the formats of each line in the earth-model files. The data formats in the tables are represented by the numeric and character format codes listed in Table 11. These codes are identical to the conversion specifiers used in the C programming language [Ker88].

TABLE 11: FORMAT CODES

Format Code	Description of Data Type
%d	Integer.
%f	Floating-point number.
%lf	Double-precision, or long floating-point number.
%s	Character string.
%%	Any alphanumeric characters (free text) that may optionally follow input numeric data types. This is a shorthand notation of writing %*[^\\n].

Some format codes are multiplied by coefficients that defined in earlier rows (for example, $nz \times \%f$). These coefficients indicate the number of consecutive times a particular data type is read from the input file on a single line.

▼ **Operational Procedures**

Comments are included in input files to help users interpret the files. A comment is any character in a file that is not read by the parser. Any line starting with the "#" character is considered to be a comment. Some comments are required, and others are optional. For required comments, the parser either looks for the "#" character or simply skips a line. The positions of required comments are defined in the format tables. If the parser ignores characters at the end of a line, then the line format includes the free text "%*" format code. In some files, optional comment lines preceded by the "#" character are allowed and ignored, but in other files optional comment lines are not ignored by the parser and therefore not allowed. Most files follow a convention for comment usage. If comment usage in a particular file is unclear, follow the example in the sample file that accompanies the file description (for example, Figure 5).

Constructing Location Earth-model Files

Successful event location processing requires four input files:

- Velocity Model Specification file (VMSF)
- One-dimensional (1-D) Travel-time (TT) tables
- LP Grid Index (LPGI) file
- LP Phase Velocity (LPPV) file

A VMSF combines velocity model specification parameters with velocity earth-model data. A 1-D TT table tabulates travel-time earth-model data discretized by distance and depth but based on a 1-D velocity model. The LPGI and LPPV files tabulate LP earth-model data and are only used in LP processing.

More accurate event locations can be achieved by including five optional earth-model files in event location processing:

- Radial Two-dimensional (2-D) Travel-time tables
- Ellipticity Correction (EC) tables
- Source-specific Station Correction (SSSC) files
- Location Test-site Correction (LTSC) files

- Slowness/Azimuth Station Correction (SASC) files

A radial 2-D TT table tabulates travel-time earth-model data discretized by azimuth and distance, and based on a 2-D velocity model. An EC table tabulates ellipticity earth-model data. SSSC, LTSC, and SASC files tabulate source-specific earth-model correction data.

Velocity Model Specification File

The Velocity Model Specification File (VMSF) determines the velocity model that is used for computing locations via *libloc* functionality. This ASCII file contains two types of earth-model definitions: paths to travel-time table data directories, and Station/Phase/Model (SPM) specific information. The formats of the different travel-time data files referenced in the VMSF are covered later in this chapter.

The VMSF format is given in Table 12 and a sample VMSF is listed in Figure 5. [Nag96] is an additional source of information on the VMSF, but the file format has changed slightly since that document was published.

TABLE 12: VMSF FORMAT

Line Number(s)	Format	Description
1	%s	Relative directory for path-dependent LP information.
2	%s	Radial 2-D TT station file path/name.
3	%15s%s	Velocity model name and relative directory pathway containing default TT tables.

▼ Operational Procedures

TABLE 12: VMSF FORMAT (CONTINUED)

Line Number(s)	Format	Description
4 through <i>vmodels</i> + 2	%15s%s	Velocity model name and relative directory pathway containing TT tables assigned to a given model.
<i>vmodels</i> + 3	/n	A blank line indicating end of velocity model listing.
<i>vmodels</i> + 4 through EOF	%s%s%s%lf%lf	Station name, phase name, velocity model name, sedimentary velocity (km/s), and bulk station correction (s) (not required).

```

#
# The first line of this file defines the directory locations for the
# path-dependent long-period info (namely, LR and LQ phases).
../LP
# The second line defines the path/name of a radial_2D station file.
# radial_2D uses 2-D tables to return hydro-acoustic travel times.
../radial_2D/station_lists
#
# The following lines indicate directory locations for specific velocity
# models defined below. The very first line defines the model name and
# directory location for the default travel-time table set. A blank
# line must follow the list of models immediately below. A '#' can
# occur in the first column anywhere within the list of models for
# comments.
#
# Velocity Model      Directory location relative to this directory
# -----
iasp91                ../iasp91
baltic                ../baltic

# Station/phase specific knowledge. This includes station/phase-
# dependent velocity model, sedimentary velocity, travel-time modeling
# error, and a bulk station correction term.
#
# Note that each line of station/phase-dependent knowledge includes, in
# order, the station name, phase type, velocity model, sedimentary
# velocity (km/sec) for making elevation corrections, and a bulk static
# station correction term (sec.). The bulk static station correction
# will be added to the overall travel-time relative to the velocity
# model specified on the same line. If no station/phase info resides
# here, then the default phase-dependent model specified above will be
# used. Individual line entries only need be space delimited. Any line
# with '#' in its first position will be ignored throughout this file.
# Please be aware that any SSSC files found within these travel-time
# areas will be read as well, regardless of whether or not they are to
# be employed. Also realize that an SSSC's applied will be relative to
# ellipticity, elevation and bulk static station correction terms.
#
# Sta      Phase      Velocity      Sed.      Bulk
# Name     Type       Model         Vel.      Station   Comments
#          (km/s)    Corr (s)
# -----
ARCES     *P       iasp91       5.8       0.0
ARCES     *S       iasp91       3.35      0.0
ARCES     Pn       iasp91       5.8       0.0
ARCES     Pg       iasp91       5.8       0.0
ARCES     Sn       iasp91       3.35      0.0
ARCES     Lg       iasp91       3.35      0.0
FINES     *P       iasp91       5.8       0.0
FINES     *S       iasp91       3.35      0.0
FINES     Pn       iasp91       5.8       0.0
FINES     Pg       iasp91       5.8       0.0
...

```

FIGURE 5. SAMPLE VMSF

▼ Operational Procedures

In the first section of the VMSF the lines containing the travel-time directory paths must be specified in a certain order and the paths are relative to the directory containing the VMSF. Comment lines beginning with “#” may be placed anywhere in the file and will be ignored by the program. The first non-comment line of the VMSF must contain a path to the Long-period Grid Index (LPGI) and Long-period Phase Velocity (LPPV) files. The second non-comment line specifies the path to the radial 2-D station file, which in turn lists the paths to 2-D hydroacoustic travel-time tables. While this field is required, “NONE” is an acceptable entry. In this case, the default 1-D hydroacoustic travel-time tables are used, if available, as needed by *libloc*.

Starting with the third non-comment line, the paths to seismic travel-time tables based on 1-D velocity models are specified by two fields, name and path. The 1-D velocity model name can be up to 15 characters, and the paths are relative the VMSF directory. Multiple velocity models may be specified, each on a separate line. The first model listed is the default model. One blank line must follow the last model.

The last VMSF section, which is not required, provides a means for incorporating station/phase-dependent knowledge in the travel-time calculations. This section specifies, among other things, the velocity model used for each station/phase combination listed. The lines of this section contain five unformatted fields: station name, phase name, velocity model, sediment velocity (km/s), and bulk station correction (s). The velocity model names must match one of the 1-D velocity model names assigned to a directory earlier in the file. If this last section is omitted, then the default 1-D model is used for all stations and phases. This list of station/phase pairs determines the SSSC files read by *libloc*. Therefore, to make use of an SSSC for a particular station/phase, there must be a corresponding entry in this section of the VMSF file. Equivalently, individual station/phase pairs can be suppressed by preceding the line with the “#” comment character. Figure 5 shows a listing of an example VMSF file, containing comments that restate some of the formatting guidelines. (Most of the station/phase section was truncated for this example.)

One-dimensional Travel-time Table

A 1-D Travel-time (TT) table is an ASCII file that contains travel-times and optional modeling errors for a given phase-dependent travel-time curve, discretized by distance and depth, and based on a 1-D velocity model. This section does not discuss the generation of travel-time values; it describes only the format of a 1-D TT table so that a correctly formatted table can be created.

A 1-D TT table has two sections. The first section contains travel-times and the second section contains optional modeling errors. Lines of comments are allowed before, after, and within the body of each section, but only in specific quantities and locations. In most cases, comment lines may begin with and contain any character. However, in some instances comment lines must begin with the “#” character. Blank lines are not permitted in a 1-D TT table.

The travel-times consist of sample depths, sample distances, and distance/depth-dependent estimates of travel time, as estimated from a phase-dependent travel-time curve. They are used as the initial, uncorrected estimates for a source-to-receiver travel time. Table 13 describes the format of the travel-time data section of a 1-D TT table. Capital letters in the Line Number(s) column indicate variable line numbers and are defined in the detailed description of the travel-time data.

TABLE 13: TRAVEL-TIME DATA FORMAT

Line Number(s)	Format	Description
1	%s%s	Velocity model name.
2	%d%*	Number of sample depths (=nz).
3 through A	nz × %f	Sample depths (km below sea level).
A + 1	%d%*	Number of sample distances (=nx).
A + 2 through B	nx* %f	Sample distances (arc degrees).
B + 1	%s	Comment line that begins with “#”.
B + 2 through C	nz × nx × %f	Travel-time estimates (s).

▼ Operational Procedures

An example 1-D TT table is shown in Figure 6 for a PKPab phase.

```
# IASPEI travel-time table for phase: PKPab
14  Number of depth samples at the following depths (km):
    0.00 15.00 35.00 50.00 75.00 100.00 150.00 200.00 250.00 300.00
400.00 500.00 600.00 700.00
32  Number of distance samples at the following distances (deg):
145.00 146.00 147.00 148.00 149.00 150.00 151.00 152.00 153.00 154.00
155.00 156.00 157.00 158.00 159.00 160.00 161.00 162.00 163.00 164.00
165.00 166.00 167.00 168.00 169.00 170.00 171.00 172.00 173.00 174.00
175.00 176.00
# Travel time at depth = 0.00 km.
1177.7375
1181.5366
1185.4521
1189.4470
1193.5034
1197.6097
1201.7579
1205.9418
1210.1566
1214.3987
1218.6647
1222.9520
1227.2583
1231.5814
1235.9197
1240.2715
1244.6354
1249.0101
1253.3944
1257.7875
1262.1881
1266.5955
1271.0088
1275.4272
1279.8502
1284.2771
1288.7072
1293.1400
1297.5748
1302.0115
1306.4492
1310.8877
```

... Travel times at other depths ...

```
# Modelling error(s)
  2  2
144.0 180.0
  0.0 200.0
#
  1.4
  1.4
#
  1.1
  1.1
```

FIGURE 6. SAMPLE 1-D TRAVEL-TIME TABLE

Line 1 of the travel-time data section in a 1-D TT table contains a single line whose first or second space-delimited string identifies the 1-D velocity model used to create the travel-time curve. The first 15 characters in the velocity model name will be written into the `assoc.vmodel` and the `ar_info.vmodel` fields (if applicable). If the first space-delimited string is the character "#", then the second string should be a name identifying the velocity model. If the first string is not the character "#", then this string should identify the velocity model. The remainder of this comment line may contain any character. A common practice is to start the comment with the "#" character, identify the velocity model, and specify the phase for which the table is valid, as in Figure 6.

Line 2 specifies the integer number of sample depths made on the travel-time curve. The number of sample depths (nz in Table 13 on page 39) may be followed by a short comment on the same line. This comment may begin with and contain any character.

Line 3 begins a list of depths at which the travel-time curve was sampled. All depths must be positive numbers. The nz sample depths may all be listed on one line, or spread across multiple lines. If the depths are all listed on one line, then the last line containing sample depths (A in Table 13) is 3. If the depths are spread across m lines (where $1 < m \leq nz$), then A is $2 + m$. If multiple depths are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the travel-time data.

Line $A + 1$ specifies the integer number of sample distances made on the travel-time curve. The number of sample distances (nx in Table 13) may be followed by a short comment on the same line. This comment may begin with and contain any character.

Line $A + 2$ begins a list of distances at which the travel-time curve was sampled for all nz depths. The nx sample distances may all be listed on one line, or spread across multiple lines. If the distances are all listed on one line, then the last line containing sample distances (B in Table 13) is $A + 2$. If the distances are spread across n lines (where $1 < n \leq nx$), then B is $A + 1 + n$. If multiple distances are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the travel-time data.

▼ Operational Procedures

Line $B + 1$ is a required comment line that separates the sampling data from the travel-times. This line precedes the travel-times estimated at all n_x distances for the first sampled depth on the travel-time curve. This comment line must begin with the “#” character. A common practice is to indicate this depth in the comment line.

Line $B + 2$ begins the n_z sets of travel times; one set for each sampled depth. Each set of travel times is composed of n_x estimates of the travel-time curve; one estimate for each sampled distance. The travel times must be listed in the same order as the sample distances. The times for a particular sample depth may all be listed on one line, or spread across multiple lines. If multiple times are present on a given line, then they must be separated by at least one space. The alignment of the travel times should mimic that of the sample distances (lines $A + 2$ through B). Although not a requirement, alignment makes the association between distance and travel time more readable in the file.

Multiple sets of travel times must be listed in the same order as the corresponding sample depths. Each set must be separated by a comment line that begins with the “#” character. A common practice is to indicate the sample depth for the upcoming set of travel times in the comment line. No other comment lines are allowed in this portion of the travel-time data. The last line in a 1-D TT table containing travel times is line C .

The travel-time data are completely specified at this point in a 1-D TT table. The remainder of the 1-D TT table specifies any modeling error data. Modeling error data are optional. If there are no modeling error data for the travel-time curve, then the 1-D TT table is complete.

The modeling error data consist of sample depths, sample distances, and distance/depth-dependent estimates of the standard error (standard deviation) associated with the travel-time estimates from a phase-dependent travel-time curve. The modeling errors specified in a 1-D TT table may be either a single-value, distance-dependent values, or distance/depth-dependent values. This choice dictates the format of the modeling error data. Any of these formats may be employed regardless of whether the travel times themselves are distance/depth-dependent or distance-dependent only. Modeling errors may not be depth-dependent only. The

modeling error data immediately follow the travel-time data in a 1-D TT table without interruption by a blank line or text. See Figure 6 for an example of a 1-D TT table with distance/depth-dependent modeling error data.

The first format for specifying modeling error data in a 1-D TT table is the single-valued format. This format is employed when the modeling error is single-valued across all sample distances and depths of the travel-time curve. Table 14 describes the single-valued format.

TABLE 14: SINGLE-VALUE TRAVEL-TIME MODELING ERROR DATA FORMAT

Line Number(s)	Format	Description
C + 1	%s	Comment line.
C + 2	%d%d%*	Number of sample distances (=1) and number of sample depths (=1).
C + 3	%s	Comment line.
C + 4	%f%*	Single-value travel-time modeling error (standard error estimate of travel time).

Line C + 1 of the modeling error data section contains a single comment line. This comment line must be placed immediately following the last set of travel-time estimates. It may begin with and contain any character. A common practice is to identify the start of the modeling error section or to indicate that the modeling error is single-valued.

Line C + 2 consists of two "1" indicators separated by spaces. These indicators signify that a single modeling error will be associated with all travel-times read from the TT table. The indicators may be followed by a short comment on the same line. The comment may begin with and contain any character.

Line C + 3 is a comment line that separates the single-valued modeling error indicators from the single modeling error itself. This comment line may begin with and contain any character. Common practice is to only include the "#" character in this comment line.

▼ Operational Procedures

Line C + 4 specifies the single modeling error. This modeling error will be associated with any travel time read from the travel-time data section, regardless of distance and depth.

The second format for specifying modeling error data in a 1-D TT table is the distance-dependent format. This format is employed when the modeling error is distance-dependent only over the entire travel-time curve. Table 15 describes the distance-dependent format. The acronym "EOF" in Table 15 stands for "end-of-file".

TABLE 15: DISTANCE-DEPENDENT TRAVEL-TIME MODELING ERROR DATA FORMAT

Line Number(s)	Format	Description
C + 1	%s	Comment line.
C + 2	%d%d%*	Number of sample distances (=mx) and number of sample depths (=1).
C + 3 through D	mx × %f	Sample distances (arc deg).
D + 1	%s	Comment line.
D + 2 through EOF	mx × %f	Travel-time modeling errors (standard error estimates of travel time).

Line C + 1 of the modeling error section contains a single comment line. The comment line must be placed immediately following the last set of travel-time estimates. It may begin with and contain any character. A common practice is to identify the start of the modeling error section or to indicate that the modeling error is distance-dependent only.

Line C + 2 specifies the integer number of sample distances used to estimate the modeling error of the travel-time curve. The number of sample distances should be followed by a space and a "1". The "1" signifies that the modeling errors are independent of depth. These two values may be followed by a short comment on the same line. The comment may begin and end with any character.

Line $C + 3$ begins a list of distances at which the modeling errors were estimated. The mx sample distances may differ from those used to estimate travel times that have a distance dependency. The sample distances may all be listed on one line, or spread across multiple lines. If the distances are all listed on one line, then the last line containing sample distances (D in Table 15) is $C + 3$. If the distances are spread across m lines (where $1 < m \leq mx$), then D is $A + 2 + m$. If multiple distances are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the modeling error data.

Line $D + 1$ is a comment line that separates the sample distances from the distance-dependent modeling errors. This comment line may begin with and contain any character. A common practice is to indicate that the modeling errors are distance-dependent.

Line $D + 2$ begins the list of modeling errors. The mx modeling errors must be listed in the same order as the corresponding sample distances. The modeling errors may be listed on one line, or spread across multiple lines. If multiple modeling errors are present on a given line, then they must be separated by at least one space. The alignment of the modeling errors should mimic that of the sample distances (lines $C + 3$ through D). Although not a requirement, alignment makes the association between distance and modeling error more readable in the file. Comment lines are not allowed in this portion of the modeling error data.

The third format for specifying modeling error data in a 1-D TT table is the distance/depth-dependent format. This format is employed when the modeling error is both distance- and depth-dependent over the entire travel-time curve. Table 16 describes the distance/depth-dependent format.

▼ Operational Procedures

TABLE 16: DISTANCE/DEPTH-DEPENDENT TRAVEL-TIME MODELING ERROR DATA FORMAT

Line Number(s)	Format	Description
C + 1	%s	Comment line.
C + 2	%d%d%*	Number of sample distances (=mx) and number of sample depths (=mz).
C + 3 through D	mx × %f	Sample distances (arc degrees).
D + 1 through E	mz × %f	Sample depths (km below sea level).
E + 1	%s	Comment line that begins with "#".
E + 2 through EOF	mx × mz × %f	Travel-time modeling errors (standard error estimates of travel time).

Line C + 1 of the modeling error section contains a single comment line. The comment line must be placed immediately following the last set of travel-time estimates. It may begin with and contain any character. A common practice is to identify the start of the modeling error section or to indicate that the modeling error is distance/depth-dependent.

Line C + 2 specifies the integer number of sample distances and number of sample depths used to estimate the modeling error of the travel-time curve. These two values may be followed by a short comment on the same line. This comment may begin with and contain any character.

Line C + 3 begins a list of distances at which the modeling errors were estimated for all *mz* depths. The *mx* sample distances may differ from those used to estimate travel times that have a distance dependency. The sample distances may all be listed on one line, or spread across multiple lines. If the distances are all listed on one line, then the last line containing sample distances (*D* in Table 16) is C + 3. If the distances are spread across *m* lines (where $1 < m \leq mx$), then *D* is C + 2 + *m*. If multiple distances are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the modeling error data.

Line $D + 1$ begins a list of depths at which the modeling errors were estimated. All depths must be positive numbers. The mz sample depths may differ from those used to estimate travel times that have a depth dependency. The sample depths may all be listed on one line, or spread across multiple lines. If the depths are all listed on one line, then the last line containing sample depths (E in Table 16) is $D + 1$. If the depths are spread across n lines (where $1 < n \leq mz$), then E is $D + n$. If multiple depths are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the modeling error data.

Line $E + 1$ is a comment line that separates the sample distances and depths from the distance/depth-dependent modeling errors. This comment line must begin with the “#” character. A common practice is to indicate the first sample depth in the comment line.

Line $E + 2$ begins the mz sets of modeling errors; one set for each sampled depth. Each set is composed of mx modeling errors; one modeling error for each sampled distance. The modeling errors must be listed in the same order as the corresponding sample distances. The modeling errors for a particular sample depth may all be listed on one line, or spread across multiple lines. If multiple modeling errors are present on a given line, then they must be separated by at least one space. The alignment of the modeling errors should mimic that of the sample distances (lines $C + 3$ through D). Although not a requirement, it makes the association between distance and modeling error more readable in the file.

Multiple sets of modeling errors must be listed in the same order as the sample depths they correspond to. Each set must be separated by a comment line that begins with the “#” character. A common practice is to indicate the sample depth for the upcoming set of modeling errors in the comment line. No other comment lines are allowed in this portion of the modeling error data. The 1-D TT table file should terminate after the last line of modeling error data.

▼ Operational Procedures

Radial 2-D Travel-time Tables

Radial 2-D travel-time (TT) tables are used for storing hydroacoustic and infrasonic travel-times. This section describes the contents of the binary-format radial 2-D TT files, the contents of the associated ASCII control files, and the directory structure; it does not describe the creation of the binary files. The top-level file that controls the data that are utilized by *libloc* is the station file, the path of which is specified in the Velocity Model Specification File (VMSF). The station file must contain two lines, one for the path to the file containing the hydroacoustic station names, and another for the path to the file containing the infrasonic station names. As in the VMSF, "NONE" is an acceptable entry in the station file. In that case, the default 1-D TT table is used as needed by *libloc*.

Within the top-level radial 2-D directory, the hydroacoustic and infrasonic data must reside in separate subdirectories. In addition to the station file that lists station names, a file named `time_guide` must be present in each of these subdirectories to specify the time periods for which separate travel-time tables exist. A separate directory of TT tables exists for each time period (WINTER, SPRING, SUMMER, and AUTUMN).

The first line of the `time_guide` file specifies an additional modeling error term for hydroacoustic arrivals from seismic events. These types of arrivals have increased uncertainties because of the uncertainty in the travel time due to earth-to-ocean coupling. Subsequent lines in the `time_guide` file specify the time periods for which TT tables exist. The lines are ordered chronologically starting with the period containing Julian day 1. Each line contains two fields: the period name, which must correspond to the directory name containing the associated TT tables, and the Julian day of the end of the time period.

The binary 2-D TT files reside in directories with the same names as the time period names in the `time_guide` file. The individual TT filenames correspond to station names, and must match those listed in the station file.

At the top of each 2-D TT file the station latitude and longitude, the time period name length, and the time period name are listed. Following these values, the travel times are specified by azimuth. For each azimuth, the file specifies the azimuth, the number of TT values, the distance between travel times (deg), followed by the travel times and the modeling errors for each point along that azimuth.

Ellipticity Correction Table

Ellipticity correction (EC) tables are a set of ASCII files associated with a particular 1-D velocity model that provide travel-time corrections by phase to compensate for the deviations of the earth from a perfect sphere. If ellipticity correction data exist for a given velocity model, the path to the correction files is specified by the file `<vmodel>.elcor_dir` located in the corresponding 1-D velocity-model directory. Appendix C.2 of [Nag96] provides a detailed description of these files, including a discussion of their derivation and a formal description of the file format.

Source-specific Station Correction Table

Source-specific Station Correction (SSSC) tables contain 2- or 3-D regional travel-time corrections relative to a particular 1-D velocity model. If they exist for a given velocity model, the ASCII SSSC files reside in a directory named `SSSC/` under the corresponding 1-D velocity model directory. SSSC files exist for the IASP91 velocity model.

Each SSSC file specifies corrections for a distinct station/phase pair. Within each file, the corrections are defined over a rectangular geographic region that may or may not include the station. For most of the SSSC files currently in use, this rectangular region is approximately centered on the station. Each SSSC file defines an orthogonal latitude and longitude grid, and an optional depth grid. The files specify a correction and a modeling error at each grid node. The actual travel-time correction applied by *libloc* is based on a bi-cubic interpolation of the correction values at the grid nodes surrounding the event hypocenter. In the case of 2-D SSSC files, the correction is applied regardless of the event depth. For depth-

▼ Operational Procedures

dependent SSSC files, the event must lie within the depth grid for an arrival to receive a correction. In this case, the final correction is a linear interpolation of the two bi-cubic interpolations at the depth samples bounding the event depth.

To yield correction interpolations that smoothly approach zero at the edges of the correction regions, the correction values should be zero along all grid boundaries. For depth-dependent SSSC files, which define a correction volume, this boundary condition includes the bottom surface of the correction volume. The simplest way to satisfy this requirement is to define a separate four-point latitude/longitude grid for the bottom surface of the correction grid. In location modes, *EvLoc/libloc* writes a single warning message if one or more of the SSSC files violate the zero-boundary condition. To see the files that violate the condition and the number of occurrences in each file, run *EvLoc* with the *mode* parameter set to 3, which outputs only that information. (This minor SSSC file-check functionality is controlled by the high-level *mode* parameter because mode 3 may be expanded in the future to perform a general check of all required *EvLoc* configuration files, independent of any location or magnitude calculation.) Violating the zero-boundary condition is not a fatal error. However, violations of this condition should be avoided to reduce the possibility of oscillatory, non-convergent solutions when an event location is very close to a SSSC region boundary.

This remainder of this section describes the SSSC file format in detail. Table 17 lists the SSSC file format and Figure 7 shows an abridged sample SSSC file to augment the written description. In practice, each data line is preceded by one or more comment lines that describe the succeeding data line, but the comments are not required by the file format so they are excluded from the format description in Table 17. The first column in Table 17 specifies the number of lines corresponding to each data section instead of the line number. Additionally, some of the lines are only used in depth-dependent SSSC files. The column one values for these lines are enclosed in parentheses and noted in the description field in Table 17. Comment lines, starting with the “#” character, are allowed to separate any of the data sections but are not allowed within the range of lines corresponding to a single data section.

TABLE 17: SSSC FILE FORMAT

Number of Lines per Data Section	Format	Description
(1)	%s%s	Optional version line used to flag depth-dependent SSSC files. Currently, the only recognized string is "Version 3D".
1	%s%s%s%s%s%lf%lf%lf	Station name, phase name, source region, velocity model, region level, bulk station correction, sedimentary velocity, and modeling error factor (in order).
1	%d%d(%d)	Number of nodes in the latitude and longitude (and optionally depth) directions (<i>nlats</i> , <i>nlons</i> , <i>ndep</i>).
1	<i>nlats</i> × %10.3f	Latitude samples (deg). North is positive.
1	<i>nlons</i> × %10.3f	Longitude samples (deg). East of 0 is positive.
(1)	<i>ndep</i> × %10.3f	Depth samples (km). (Depth-dependent SSSC files only.)
<i>nlats</i>	<i>nlons</i> × %8.2f	Travel-time correction values (s). There are <i>nlats</i> samples consecutive lines, where each line contains <i>nlons</i> samples per line.
<i>nlats</i>	<i>nlons</i> × %8.2f	Modeling error values (s). Same row/col format as correction values.
$(2 \times (ndep - 1) \times nlats)$	<i>nlons</i> × %8.2f	Repeat correction and modelling error lines for each additional depth sample. (Depth-dependent SSSC files only.)
(*)	multiple formats, see above	Repeat all lines from the number of grid nodes through the modelling errors for each additional grid. (Depth-dependent SSSC files only.)

▼ Operational Procedures

```

# Station Phase Source Region Velocity Model Level StaCor SedVel MEFac
# =====
ARCES Pn fenno iasp91 reg 0.000 5.800 1.0
# Number of Lat/Lon samples, respectively:
51 176
# Latitude samples (deg) from north to south:
89.000 88.000 87.000 86.000 85.000 84.000 83.000
82.000 81.000 80.000 79.000 78.000 77.000 76.000
...
42.000 41.000 40.000 39.000
# Longitude samples (deg) from west to east:
-50.000 -49.000 -48.000 -47.000 -46.000 -45.000 -44.000
-43.000 -42.000 -41.000 -40.000 -39.000 -38.000 -37.000
...
117.000 118.000 119.000 120.000 121.000 122.000 123.000
124.000 125.000
# SSSC's in map view (origin is NW corner):
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
...
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
...
# Modeling errors in map view (origin is NW corner):
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
...
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
...

```

FIGURE 7. SAMPLE SSSC FILE

The first line of the SSSC file format is an optional version identifier used to flag depth-dependent SSSC files. The only currently-recognized value is "version 3D", which allows *libloc* to parse depth-dependent SSSC files differently than 2-D SSSC files.

The second data line consists of several general SSSC attributes such as the station and phase name. To maintain consistency of station/phase-specific information, the bulk correction and sedimentary velocity values that appear in the VMSF are also included in this line. This allows the code to adjust the SSSC's if these values have been updated in the VMSF. The values in the VMSF have precedence.

The third data line specifies the number of latitude and longitude samples used to define the SSSC region grid. In the case of depth-dependent files, this line contains a third number that specifies the number of depth samples to which the horizontal grid applies.

The next two lines list the latitude and longitude values in degrees, using a positive north and east convention. They should be listed starting from the northwest corner of the grid. If the file contains depth-dependent corrections, then the depth samples follow on the next line. These values should start at 0 and increase downward.

The next group of lines list the correction values themselves, followed by the modeling error values, both in seconds. The corrections and modeling errors are specified in a separate line for each latitude; each line contains the values for each longitude sample. After all the correction value lines, the modeling errors are listed in exactly the same format.

In depth-dependent SSSC files the correction and modeling error lines are repeated for each depth sample. Additionally, the depth-dependent SSSC format accommodates multiple latitude/longitude grids, each with its own range of applicable depth samples. The additional latitude/longitude grids may specify sparser or denser sampling, but must all have the same bounds. To specify additional grids, the entire range of lines from the grid specification through the modelling errors are repeated as necessary. The SSSC file format does not limit the number of separate grids. However, the combined size of all the SSSC files in use is limited because the SSSC data are held in memory during *EvLoc* execution.

Location Test-site Correction File

Location Test-site Correction (LTSC) ASCII files contain travel-time corrections to arrivals originating in specific regions for specific station/phase pairs relative to a particular 1-D velocity model. The term “test-site” refers to nuclear test site locations, where a history of past events has provided the data for developing these corrections.

▼ Operational Procedures

One test-site file contains all the correction data for a given 1-D model. If present for a particular velocity model, this file is named `<vmodel>.ts_cor` and is located in the corresponding 1-D velocity-model directory. Like the SSSC's, test-site corrections are region-specific additive corrections to travel times generated through a background 1-D velocity model and are intended to account for the lateral variations of the actual earth from the 1-D model. A key difference though, is that test-site corrections are limited to a specific latitude/longitude. SSSC's and test-site corrections are mutually exclusive; only one or the other can be applied in a single location computation.

LTSC files contain correction data organized by region/phase and station. Table 18 describes the LTSC file format. No comments are allowed on separate lines in the main body of the file. However, comments may be placed at the end of a given line after all the data fields, or at the end of the file.

TABLE 18: LTSC FILE FORMAT

Line Number(s)	Format	Description
1	%d%*	Number of test-site region/phase groups.
2	%d%s%s%d%*	Region/phase group sequence number, region name (up to 9 chars), phase name, number of stations with corrections for this region/phase.
3 thru 3*, where 3* = #stations + 2	%6s%f	Station name, correction (s)
4 thru EOF	multiple formats	Repeat line 2 and corresponding station lines as in 3 thru 3* for each region/phase group corresponding to the number in line 1.

Line 1 of the file contains the total number of region/phase groups for which one or more station corrections are specified. The second line specifies region/phase group information in four fields: the region/phase group number (that is, the sequence number as it appears in the file), the station name, the phase name, and the number of stations for which corrections are specified.

Following this line, the corrections are listed on separate lines by station. The region/phase group line and corresponding station lines are repeated for each region/phase group, where the total number of these groups matches the number in the first line of the file. Figure 8 shows a sample test-site file for region names "AA" and "BB".

```

 2
 1 AA      P          7
WRA      0.50
ASAR     1.00
MAW     -1.00
VNDA    -1.20
CMAR    -1.00
KSAR     0.60
LPAZ     1.20
 2 BB      P          5
BDFB    -1.00
ROSC     1.00
LPAZ     0.50
PTI      0.20
PPT      0.50
#@(##)   iasp91.ts_cor    2.1    06/06/96

```

FIGURE 8. SAMPLE LTSC FILE

During location processing with *libloc* the test-site corrections are accessed via the region name value. For example, in *EvLoc* the *ts_region* parameter controls the corrections that are applied by *libloc*.

Long-period Grid Index File

A Long-period Grid Index (LPGI) file is an ASCII file that contains phase velocity table indices discretized by co-latitude and longitude for a particular LP phase. This section does not discuss how to generate LP velocity models (dispersion curves). However, once the models have been generated, this section is useful for converting the models into a format readable by functions in the *libLP* shared library, which are called by *libloc* functions.

▼ Operational Procedures

The phase velocity table indices in a LPGI file are integers identifying which LP phase velocity tables, or dispersion curves, are to be used in computing an LP travel time. The LP phase velocity tables are stored in the Long-period Phase Velocity (LPPV) file. The grid represented by the LPGI file encompasses the entire globe.

Table 19 describes the format of the LPGI file. Separate LPGI files must exist for the LP Rayleigh phase (LR) and Love phase (LQ). If either of these two files are not present, then the *libLP* software returns an error message and the calling application exits (see “libloc Configuration-related Error Messages” on page 110). Figure 9 shows a sample LPGI file for an LR phase.

TABLE 19: LPGI DATA FORMAT

Line Number(s)	Format	Description
1	%s	Comment line.
2	%d%d%lf%*	Number of grid cells by co-latitude (=nlat), number of grid cells by longitude (=nlon), and spacing (in arc degrees) between adjacent grid centers (=space).
A through EOF	nlat × ('\n','#','\n',nlon × %d)	Phase velocity indices.

```
# #Lats  #Lons  Spacing (deg.) for grid indices
   36     72     5.0

# Co-Latitude Range:  0.00 to  5.00 deg.
150 150 150 150 150 150 150 150 150 150 150 150 150 150 150 150 150
150 150 151 151 151 151 151 151 151 150 150 150 150 150 151 151 151 151
151 151 151 151 151 129 129 129 129 129 151 151 151 151 151 151 151 151
151 151 151 151 151 151 151 151 151 151 151 151 151 151 151 151 151 151

# Co-Latitude Range:  5.00 to 10.00 deg.
150 150 151 151 151 151 151 113 113 113 113 113 113 151 151 151 151
151 151 151 151 150 150 150 151 151 151 151 151 151 139 139 139 151
151 151 151 151 151 151 151 151 151 139 139 139 139 139 139 139 139
124 43 43 43 43 43 44 44 44 44 44 45 44 44 43 23 151 151

... Phase velocity indices for other grid cells ...
```

FIGURE 9. SAMPLE LPGI FILE

Line 1 of a LPGI file is a required comment line. The line must be present, but its contents are ignored. Commonly, this line is used to identify the grid index file and specify the phase to which the file applies.

Line 2 specifies the integer number of grid cells along the co-latitude axis (*nlat* in Table 19), the integer number of grid cells along the longitude axis (*nlon* in Table 19), and the spacing between adjacent grid cell centers (*space* in Table 19). The same spacing is applied in both the co-latitude and longitude directions. These values are constrained by the following equations: $nlon * space = 360 \text{ deg}$ and $nlat * space = 180 \text{ deg}$. The grid cell spacing may be followed by a short comment on the same line. This comment may begin with and contain any character.

Line A begins the *nlat* tables of phase velocity indices. Each index represents a cell in the phase velocity grid and provides a mapping to the phase velocities in the LPPV file. The *nlon* phase velocity indices in each table may all be listed on a single line or spread across multiple lines. If multiple indices are present on a given line, then they must be separated by at least one space. The first index in each table represents the grid cell whose western boundary lies on the Prime Meridian. The last index in each table represents the grid cell whose eastern boundary lies on the Prime Meridian. Each table represents a complete strip of the global grid along a single co-latitude range. The first table of indices represents the grid cells whose northern boundary is the North Pole. The last table of indices in the LPGI file represents the grid cells whose southern boundary is the South Pole. The LPGI file should terminate after the last table of indices.

Comments are not allowed within the tables of phase velocity indices. However, two lines must precede each latitude table, the second of which must be a comment line with a “#” character as described in Table 19. A common practice is to indicate the co-latitude of the grid cells represented by the indices in the comment line. The LPGI file shown in Figure 9 demonstrates this format.

▼ Operational Procedures

Long-period Phase Velocity File

A Long-period Phase Velocity (LPPV) file is an ASCII file that stores dispersion information for multiple grid indices. The dispersion data are phase velocities specified for a number of period samples. This section does not discuss how to generate the dispersion curves. However, once they have been generated, this section is useful for converting them into a format readable by functions in the *libLP* shared library, which are called by *libloc* functions.

Table 20 describes the format of the LPPV file. Separate LPPV files must exist for the long-period Rayleigh (LR) phase and Love phase (LQ). If either of these two files are not present, then the *libLP* software returns an error message and the calling application exits (see “libloc Configuration-related Error Messages” on page 110). Figure 10 shows an example of a LPPV file for an LR phase.

TABLE 20: LPPV DATA FORMAT

Line Number(s)	Format	Description
1	%s	Comment line.
2	%d%*	Number of phase velocity indices (=nindex).
3	%d%*	Number of sample periods (=nper).
4 through A	nper × %lf	Sample periods (sec).
A+1 through EOF	nindex × ('#', '\n', nper × %lf)	Phase velocities (km/s).

Line 1 of a LPPV file is ignored by *EvLoc*, but must be present. A common practice is to use this line to identify the phase velocity file and specify the phase to which the file applies.

Line 2 specifies the integer number of phase velocity indices (*nindex* in Table 20). Each index represents a distinct table of phase velocities. The number of phase velocity indices may be followed by a short comment on the same line. This comment may begin with and contain any character.

```

# LR parameter table discretized as index/period
154 # number of index samples
100 # number of period samples
5.000 5.051 5.102 5.155 5.208 5.263 5.319 5.376
5.435 5.495 5.556 5.618 5.682 5.747 5.814 5.882
5.952 6.024 6.098 6.173 6.250 6.329 6.410 6.494
6.579 6.667 6.757 6.849 6.944 7.042 7.143 7.246
7.353 7.463 7.576 7.692 7.812 7.937 8.065 8.197
8.333 8.475 8.621 8.772 8.929 9.091 9.259 9.434
9.615 9.804 10.000 10.204 10.417 10.638 10.870 11.111
11.364 11.628 11.905 12.195 12.500 12.821 13.158 13.514
13.889 14.286 14.706 15.152 15.625 16.129 16.667 17.241
17.857 18.519 19.231 20.000 20.833 21.739 22.727 23.810
25.000 26.316 27.778 29.412 31.250 33.333 35.714 38.462
41.667 45.455 50.000 55.556 62.500 71.429 83.333 100.000
125.000 166.667 250.000 500.000
# A0 normal oceanic, 0.15 km sed.
1.285697
1.282573
1.279342
1.276001
1.272544
1.268967
1.265263
1.261428
1.257454
1.253336
1.249067
1.244639
1.240045
1.235277
1.230327
1.225184
1.219840
1.214284
1.208506
1.202493
1.196234
1.189716
1.182924
1.175845
1.168463
1.160762
1.152725
1.144334
1.135570
1.126414
1.116847
1.106847
1.096396
1.085474
1.074064

```

... Phase velocities for additional sample periods of first index ...

... Phase velocities for remaining indices ...

FIGURE 10. SAMPLE LPPV FILE

▼ Operational Procedures

Line 3 specifies the integer number of sample periods (*nper* in Table 20) in each table of phase velocities. The number of sample periods may be followed by a short comment on the same line. This comment may begin with and contain any character.

Line 4 begins the list of *nper* sample periods. The periods may all be listed on one line, or spread across multiple lines. If more than one period is specified on a given line, then the periods must be separated by at least one space. Line A in Table 20 represents the last line containing sample periods in the LPPV file. Comment lines are not allowed within this list.

Line A + 1 begins the *nindex* phase velocity tables; one table for each phase velocity index. Each table is composed of *nper* phase velocity estimates; one estimate for each sample period. The phase velocities must be listed in the same order as the corresponding sample periods. The phase velocities for a particular index may all be listed on one line, or spread across multiple lines. If multiple phase velocities are present on a given line, then they must be separated by at least one space. The alignment of the phase velocities should mimic that of the sample periods (lines 4 through A). Although not a requirement, this alignment makes the association between period and phase velocity more readable in the file.

Comments are not allowed within the phase velocity tables. However, a comment line containing the "#" as the first character must precede each table as described in Table 20. A common practice is to indicate the associated phase velocity index and a brief description of the geology represented in the comment line. The LPPV file shown in Figure 10 demonstrates this format.

Slowness/Azimuth Station Correction File

A Slowness/Azimuth Station Correction (SASC) file is an ASCII file that contains slowness specification data, optional rotational coefficients, and binned slowness/azimuth corrections and modeling errors for a particular station. Each station may be associated with only one SASC file. This document does not discuss how to

determine such data. However, this section does describe the format of an SASC file so that a readable one may be created when slowness/azimuth correction data are available for a particular station.

Table 21 shows the format of an SASC file. Capital letters in the Line Number(s) column indicate variable line numbers and are defined in the detailed description following the table. Lines of comments are allowed anywhere within an SASC file. Comment lines must begin with the “#” character. Blank lines may also be inserted anywhere in an SASC file. An example SASC file is shown in Figure 11 for station BGCA.

TABLE 21: SASC FILE FORMAT

Line Number(s)	Format	Description
A	%lf%lf%lf%lf%lf%lf%lf%lf%lf	Default slowness vector corrections (x- and y-components), default slowness modeling error, and four optional rotation coefficients.
B	%d	Number of slowness/azimuth bins ($=nbin$).
C through C + $nbin - 1$	%lf%lf%lf%lf%lf%lf%lf%lf%lf %lf%lf%lf	Lower and upper bounds for slowness and azimuth bins, slowness and azimuth corrections, and slowness and azimuth modeling errors.

Line A of a SASC file contains both slowness specification data and optional rotation coefficients. This line may be preceded by any number of comments or blank lines. The slowness specification data are composed of default slowness vector corrections (x- and y-components) and a default slowness modeling error. These data may be optionally followed by four rotation coefficients that compose a rotation matrix. The coefficients are listed row-by-row in the order: r_{11} , r_{12} , r_{21} , r_{22} , where r is the rotation matrix.

▼ Operational Procedures

```

# Slowness-azimuth station correction table
#
# BGCA, 20000105 21:08:59
#
# time period: 1996-01-01 - 1999-12-16
# 8240 input slowness-azimuth readings
# 269 bins populated with observations
# 3004 readings contributed to the corrections
# provisional default corrections: sx: -0.281 , sy: 0.125
# azimuthal variance of default corrections: 69.768
# no global trend has been detected
# overall sdev of slowness residuals: 1.676
# overall sdev of azimuth residuals: 21.872
# uncorrected bins, slowness residual sdev: 2.108
# uncorrected bins, azimuth residual sdev: 29.027
# corrected bins, slowness residual sdev: 0.745
# corrected bins, azimuth residual sdev: 8.333
#
# Structure of the slowness-azimuth correction (SASC) table:
#   comment lines begin with '#'
#   line 1: Contains global slowness vector AFFINE transformation and
#           the slowness modeling error (azimuth modeling error is derived)
#   column 1: global slowness correction, x-component
#   column 2: global slowness correction, y-component
#   column 3: default slowness modeling error (sec/deg)
#   columns 4-7 are global rotation coefficients if the station has
#           orientation error. They are derived from a separate program and
#           they are optional.
#   line 2: number of bins with correction (ncor)
#           (if only defaults are given the rest of this file remains empty)
#   line 3 - line ncor+2: bin corrections and modeling errors
#   column 1: bottom of slowness interval
#   column 2: top of slowness interval
#   column 3: bottom of azimuth interval
#   column 4: top of azimuth interval
#   column 5: slowness correction
#   column 6: azimuth correction
#   column 7: slowness modeling error
#   column 8: azimuth modeling error
#
# def_sx_corr def_sy_corr def_mdl_err, all a12 a21 a22
# True_azi = Obs_azi + 10 (degrees)
0.0000 0.0000 2.1080 0.9848 0.1736 -0.1736 0.9848
# Number of user-specified SASC bins:
40
# User-specified bin defs, S-A corrections and S-A modeling errors:
4.000 6.000 15.000 30.000 -0.711 -9.714 0.920 11.222
4.000 6.000 30.000 45.000 -0.592 -3.590 0.604 7.577
4.000 6.000 45.000 60.000 -0.104 2.393 0.716 9.313
4.000 6.000 60.000 75.000 -0.033 1.782 0.753 11.872
4.000 6.000 75.000 90.000 -0.095 0.891 0.832 8.915
4.000 6.000 105.000 120.000 0.886 -4.840 0.997 12.159
4.000 6.000 120.000 135.000 0.722 2.310 0.840 10.775
4.000 6.000 225.000 240.000 0.440 0.987 0.926 11.765
4.000 6.000 270.000 285.000 0.385 3.948 0.836 14.472
6.000 8.000 0.000 10.000 -0.276 -0.329 0.809 7.281

```

Remaining binned slowness/azimuth correction data ...

FIGURE 11. SAMPLE SASC FILE

The default slowness vector corrections are subtracted from the observed slowness vector components to form corrected slowness vector components. If rotation coefficients are specified in the SASC file, then the observed horizontal slowness vector components are rotated by applying an affine transform that uses the rotation coefficients. This rotation occurs prior to applying the default slowness vector corrections. The corrected azimuth is computed as the \tan^{-1} of the corrected horizontal slowness vector components.

The default slowness modeling error and the observed slowness are used to estimate the default azimuth modeling error. The total slowness and azimuth errors are estimated from the default slowness and azimuth modeling errors if binned slowness/azimuth modeling errors are not present in the SASC file.

Line *B* specifies the integer number of slowness/azimuth bins (*nbin* in Table 21) listed in the remainder of the SASC file. This line may be preceded by any number of comment or blank lines.

Lines *C* through *C+nbin-1* contain binned slowness/azimuth corrections and modeling errors. These lines may be preceded by any number of comment or blank lines. Each line specifies a lower and upper bound on a particular slowness and azimuth bin. The slowness correction, azimuth correction, slowness modeling error, and azimuth modeling error associated with the slowness/azimuth bin follow the bin specification.

If the slowness and azimuth of an arrival at a station with a SASC file fall within one of the slowness/azimuth bins, then the associated slowness and azimuth corrections are subtracted from the observed slowness and azimuth. This correction occurs prior to rotating the horizontal slowness vector components. Also, the slowness and azimuth modeling errors are added to observed slowness and azimuth measurement errors (*delslo* and *delaz*, respectively) in a root-mean-square (rms) sense to produce a total slowness error. The binned modeling errors are used instead of the default modeling errors if the former modeling errors exist.

▼ Operational Procedures

Constructing Magnitude Earth-model Files

Successful magnitude processing requires three input files:

- Magnitude Description File (MDF)
- Transmission Loss Specification File (TLSF)
- Transmission Loss Models (TLM)

A MDF combines magnitude configuration parameters with earth-model correction data. A TLSF configures transmission loss data. A TLM tabulates a unique transmission loss earth model.

More accurate event magnitudes can be achieved by applying magnitude test-site corrections read from one or more Magnitude Test-site Correction (MTSC) files.

Magnitude Description File

A Magnitude Description File (MDF) defines configurations in the form of magnitude specification (control) parameters and specifies earth-models in the form of bulk magnitude station corrections. These parameters are used by several applications (including *EvLoc*) in conjunction with a TLSF to determine station and/or network magnitudes.

A MDF has two sections optionally preceded or followed by an unlimited number of comment lines. The first section contains lines of magnitude specification parameters. The second section contains lines of bulk magnitude station corrections. Lines of comments may also be placed within the body of each section. All comment lines must begin with the “#” character. An example MDF is shown in Figure 12.

The lines of magnitude parameters specify magnitude control settings. Table 22 identifies the format for a given line of magnitude specification parameters. A blank line must follow the complete list of magnitude specification parameters. This blank line informs the MDF parser that the magnitude specification section is complete.

```

#
# This is a Magnitude Description File (MDF). This file isolates
# high-level magnitude specification and control settings. It also is
# used to describe all mappings to the transmission loss (TL)
# information stored within the Transmission Loss Specification File
# (TLSF). A '#' can occur in the first column anywhere within this
# file for comments.
##
# Mag TL detect event Algo Dist. Range SDlimits Wgt
# Type Type amptype amptype Code min max LB UB BL ave Comments
#-----
mb mb A5/2 - 0 20.0 100.0 0.0 2.0 .35 0 Net Avg mb
mb_ave mb A5/2 - 0 20.0 100.0 0.0 2.0 .35 0 Net Avg mb
mb_mle mb A5/2 ANP/2 2 20.0 100.0 .35 .35 .35 0 MLE mb
ms ms ALR/2 - 0 2.0 100.0 0.0 2.0 .25 0 Net Avg Ms
ms_ave ms ALR/2 - 0 2.0 100.0 0.0 2.0 .25 0 Net Avg Ms
ms_mle ms ALR/2 ANL/2 2 2.0 100.0 .25 .25 .25 0 MLE Ms
ml ml SBSNR - 0 2.0 20.0 0.0 2.0 .35 1 SBSNR ml
mlppn ml SBSNR - 0 2.0 20.0 0.0 2.0 .35 1 SBSNR ml
mb1 mb1 A5/2 - 0 2.0 100.0 0.0 2.0 .30 1 Net Wgt Avg mb
mb1mle mb1 A5/2 ANP/2 2 2.0 100.0 .30 .30 .30 1 MLE mb1
ms1 ms1 ALR/2 - 0 2.0 100.0 0.0 2.0 .25 1 Net Wgt Avg Ms
ms1mle ms1 ALR/2 ANL/2 2 2.0 100.0 .25 .25 .25 1 MLE Ms1

# Specify station/TLtype-dependent bulk static station corrections.
#
# Sta Station Station Comments
# Name TLType Cor Cor Err (# of data used to get station corr)
#-----
DFAULT ml 0.000 0.000 Default value of station corr. error
DFAULT mb1 0.000 0.200 Default value of station corr. error
DFAULT ms1 0.000 0.250 Default value of station corr. error
ABKT mb1 -0.114 0.000 348
AFI mb1 -0.185 0.000 195
ALQ mb1 0.347 0.000 654
AQU mb1 -0.007 0.000 25
ARCES mb1 0.102 0.000 3160
ARU mb1 -0.246 0.000 1063
ASAR mb1 0.172 0.000 6498
BBB mb1 -0.438 0.000 77
BDFB mb1 -0.003 0.000 907

mb1 bulk magnitude station corrections for other stations

ARCES ms1 0.016 0.2 383 from stacor_all
BDFB ms1 -0.092 0.2 263
BGCA ms1 0.037 0.2 268
BJT ms1 -0.062 0.2 437
BOSA ms1 -0.110 0.2 198
BRAR ms1 0.148 0.2 184
CMAR ms1 0.202 0.2 659
CPUP ms1 0.008 0.2 118
DBIC ms1 -0.019 0.2 179

ms1 bulk magnitude station corrections for other stations

```

FIGURE 12. SAMPLE MDF

▼ Operational Procedures

TABLE 22: MAGNITUDE SPECIFICATION PARAMETERS

Name	Format	Description	Example
<i>magtype</i>	%s	Magnitude descriptor.	mb_m1e
<i>TLtype</i>	%s	Transmission loss descriptor.	mb
<i>det_amptype</i>	%s	Arrival-based amplitude measure descriptor.	A5/2
<i>ev_amptype</i>	%s	Origin-based amplitude measure descriptor.	ANP/2
<i>algo_code</i>	%d	Network magnitude algorithm code: 0 = network-average. 1 = MLE without bootstrap resampling. 2 = MLE with bootstrap resampling.	2
<i>dist_min</i>	%f	Minimum valid origin-to-station distance for a station magnitude to be defining (deg).	20.0
<i>dist_max</i>	%f	Maximum valid origin-to-station distance for a station magnitude to be defining (deg).	100.0
<i>LB</i>	%f	Lower-bound standard deviation.	0.35
<i>UB</i>	%f	Upper-bound standard deviation.	0.35
<i>BL</i>	%f	Baseline standard deviation.	0.35
<i>wgt_ave</i>	%d	Estimate weighted-average network magnitudes? 0 = no, 1 = yes.	0
<i>comments</i>	%*	Comments.	mb MLE

Each line of magnitude specification parameters corresponds to a unique magnitude type (*magtype*). *Magtype* is linked to a transmission loss type (*TLtype*) and two amplitude measure types (*det_amptype* and *ev_amptype*). The *magtype-TLtype* linkage helps to define the earth-model parameters and station corrections that will be used to compute station magnitudes for the magnitude type. Examples of *TLtypes* are "mb", "ms", "m1", "mb1", and "ms1". The two *magtype-amptype* linkages define the amplitude data that are used to compute station magnitudes. Examples of *det_amptypes* are "A5/2", "ALR/2", "SBSNR", and "-". Examples of *ev_amptypes* are "ANP/2", "ANL/2", and "-". To compute station magnitudes

using only arrival-based amplitudes, specify *det_amptype* and set *ev_amptype* to “-”. Similarly, to compute station magnitudes based only on origin-based amplitudes, set *det_amptype* to “-” and specify *ev_amptype*. To compute station magnitudes based on both arrival-based and origin-based amplitudes, specify both *det_amptype* and *ev_amptype*.

Only defining station magnitudes are used to estimate network magnitudes associated with *magtype*. One of the criteria for identifying station magnitudes as defining is a source-receiver distance check. The allowable source-receiver distance must be between the values of the *dist_min* and *dist_max*.

The *algo_code* and *wgt_ave* magnitude specification parameters control how the network magnitudes associated with *magtype* will be estimated from defining station magnitudes. Network-average magnitudes are estimated when *algo_code* = 0, maximum-likelihood estimates (MLEs) are made when *algo_code* = 1, and MLEs with bootstrap resampling of the station magnitudes are estimated when *algo_code* = 2. Standard, unweighted network magnitudes are estimated when *wgt_ave* = 0, and weighted network magnitudes are estimated when *wgt_ave* = 1. Network-average and MLE magnitudes may be weighted or unweighted. The network average and MLE algorithms are discussed in [IDC5.2.1Rev1], while weighted averages and bootstrap resampling are outlined in [IDC7.1.6].

The *LB*, *UB*, and *BL* parameters control limits and default values for the network magnitude standard deviations associated with *magtype*. The network magnitude standard deviations are determined according to *algo_code* and *wgt_ave*, but are limited by *LB* and *UB*. These latter parameters specify the lower and upper bounds constraining the standard deviation of the network magnitude, respectively. *BL* specifies a default (baseline) value for the standard deviation. The baseline value is used as the network magnitude standard deviation and uncertainty when only one defining station magnitude exists for *magtype*. The baseline value is also used to equally weight the station magnitudes when *wgt_ave* = 1 and *LB* = *UB*.

Comments may be written at the end of a line of magnitude specification parameters. A common practice is to describe the algorithm used to estimate the network magnitudes for *magtype*. A “#” does not need to precede end-of-the-line comments.

▼ Operational Procedures

The lines of bulk magnitude station correction parameters specify static station magnitude corrections and error estimates for the corrections. Table 23 identifies the format for a given line of bulk magnitude station correction parameters. The end of the MDF indicates to the parser that this section is complete. Comment lines may be inserted after these parameters.

TABLE 23: BULK MAGNITUDE STATION CORRECTION PARAMETERS

Name	Format	Description	Example
<i>sta</i>	%s	Station code.	ZAL
<i>TLtype</i>	%s	Transmission loss descriptor.	mb1
<i>sta_corr</i>	%f	Bulk magnitude station correction.	0.183
<i>sta_corr_err</i>	%f	Bulk magnitude station correction error.	0.000
<i>comments</i>	%*	Comments.	3753

Each line of bulk magnitude station correction parameters corresponds to a unique station (*sta*) and *TLtype* combination. The *sta_corr* and *sta_corr_err* parameters specify the bulk magnitude station correction and bulk magnitude station correction error, respectively, for a unique *sta-TLtype* combination. In addition, this section must contain default values for the correction and error for each unique *TLtype*. The default values are specified by setting *sta* to "DEFAULT" and then specifying appropriate default values in *sta_corr* and *sta_corr_err*. If default bulk magnitude station correction parameters are not specified for a particular *TLtype* whose associated *magtype* specifies the weighted average algorithm (*wgt_ave* = 1), then an error message will be written to `STDERR` (see "libmagnitude Configuration-related Error Messages" on page 123).

Comments may be written at the end of a line of bulk magnitude station correction parameters. A common practice is to list the number of detections used to estimate the bulk magnitude station correction parameters. A "#" does not need to precede end-of-the-line comments.

Transmission Loss Specification File

A Transmission Loss Specification File (TLSF) is the central control point for specifying regional and global transmission loss information. This file specifies which transmission loss model to use for a given *Tltype*, station, phase, and channel (frequency) combination. The transmission loss information is used by several applications (including *EvLoc*) in conjunction with a MDF to determine station and/or network magnitudes.

A TLSF has three sections separated by a blank line and optionally preceded or followed by an unlimited number of comment lines. The first section contains lines of transmission loss model (TLM) pathway parameters. The second section contains lines of default TLM description parameters. The third section contains lines of station-specific TLM description parameters. Lines of comments may also be placed within the body of each section. All comment lines must begin with the “#” character. An example TLSF is shown in Figure 13.

The lines of TLM pathway parameters specify partial TLM filenames and relative pathways. Table 24 identifies the format for a given line of TLM pathway parameters.

Each line of TLM pathway parameters corresponds to a unique *Tltype*. The link between a *Tltype* and TLM pathway parameters is established in the default TLM description section of the TLSF. A line of TLM pathway parameters partially identifies the default TLM filename and completely identifies its directory location. The *root_name* parameter specifies the root name, or the first term, of the complete TLM filename. The *pathway* parameter specifies the directory location of the TLM relative to the directory location of the TLSF.

▼ Operational Procedures

```

#
# This is a Transmission Loss Specification File (TLSF).  This single
# file is a central control point for defining all regionalized
# transmission loss (TL) knowledge.  This generic TL information is
# deliberately separated from the specification of regionalized
# magnitude information store in the Magnitude Description File (MDF)
# and other application-specific information (e.g., event
# characterization, discrimination, etc.).  A '#' can occur in the first
# column anywhere within this file for comments.
#
# The following lines indicate directory locations for specific TL
# models defined below.
#
# TL Model          Directory location relative to this directory
# -----
qfvc                ../global          (example files, qfvc.mb, etc.)
qfvcl               ../global
global              ../global
rez_pearce          ../global
mar_bas             ../global
# Following models are more localized representations
mcoefs_defl         ../regional/mag_coefs

# Next, we define a list of valid TLtypes (one per line) along with
# their associated default TL model, indication of phase usage, and
# list of acceptable phase types.
#
# TLType           Default      Phase   List of
#                  TL Model     Depend  Phases   Comments
# -----
mb                 qfvc                0        P        Veith/Clawson
mb1                qfvcl               0        P,Pn     Veith/Clawson
ms                 rez_pearce          0        LR       Rezapour/Pearce
ms1                rez_pearce          0        LR       Rezapour/Pearce
ml                 mcoefs_defl        0        P,Pn     Local Mag (Only P/Pn)

# Station/TLtype/model-specific knowledge.
#
# Sta   Phase
# Name  TLType      TL Model          Type      Chan      Comments
# -----
ABKT    ml           mcoefs_ABKT       -         -
AFI     ml           mcoefs_AFI        -         -
ALQ     ml           mcoefs_ALQ        -         -
AQU     ml           mcoefs_AQU        -         -
ARCES   ml           mcoefs_ARCES      -         -
ARU     ml           mcoefs_ARU        -         -
ASAR    ml           mcoefs_ASAR       -         -
ATTU    ml           mcoefs_ATTU       -         -
BBB     ml           mcoefs_BBB        -         -
BDFB    ml           mcoefs_BDFB       -         -
BGCA    ml           mcoefs_BGCA       -         -
BJT     ml           mcoefs_BJT        -         -
BORG    ml           mcoefs_defl       -         -
BOSA    ml           mcoefs_BOSA       -         -
BRAR    ml           mcoefs_BRAR       -         -

```

station-specific TLM description data for other station-*TLtype* combinations

FIGURE 13. SAMPLE TLSF

TABLE 24: TLM PATHWAY PARAMETERS

Name	Format	Description	Example
<i>root_name</i>	%s	Root name of default TLM filename	qfvc
<i>pathway</i>	%s	Directory location of TLM relative to TLSF	../global

The lines of default TLM description parameters specify model and phase data needed to construct default TLM filenames for a given *Tltype*. Table 25 identifies the format for a given line of default TLM description parameters. A blank line must follow the complete list of default TLM description parameters. This blank line informs the TLSF parser that the default TLM description section is complete.

TABLE 25: DEFAULT TLM DESCRIPTION PARAMETERS

Name	Format	Description	Example
<i>Tltype</i>	%s	Transmission loss descriptor.	mb
<i>root_name</i>	%s	Root name of default TLM filename.	qfvc
<i>phase_depend</i>	%d	Is default TLM phase-dependent? 0 = no; use same TLM for all phases. 1 = yes; use unique TLM for each phase.	0
<i>list_of_phases</i>	%s	List of phase names.	P
<i>comments</i>	%*	Comments.	[Vei72]

Each line of default TLM description parameters corresponds to a unique *Tltype*. A list of phase names (*list_of_phases*) is associated with each *Tltype*. This list specifies those phases whose amplitude measurements will be used to compute station magnitudes. In addition, the amptypes of such amplitude measurements must also be identical to the *det_amptype* or *ev_amptype* settings in the magnitude specification section for the *magtype-Tltype* linkage under consideration (see "Magnitude Description File" on page 64).

▼ Operational Procedures

The default transmission loss models may be phase-dependent. In other words, a unique TLM may be used for each phase in *list_of_phases* for a given *TLtype*. Use of phase-dependent transmission loss data for individual phases is indicated by setting the *phase_depend* parameter to "1". A *phase_depend* setting of "0" indicates that the same TLM will be used for all phases associated to *TLtype*.

The filename of a default TLM is constructed from the *root_name*, *TLtype*, and *list_of_phases* parameters. *root_name* specifies the root name, or first term, of the default TLM filename. The format of the default TLM filename is:

root_name.TLtype.phase

where *phase* is one of the phase names in *list_of_phases*. The *phase* suffix is only required if a unique TLM is needed for each phase in *list_of_phases* for a given *TLtype*. In such cases, the *phase_depend* parameter should be set to "1". If the same default TLM is used for all phases in *list_of_phases* (*phase_depend* = "0"), then the *phase* suffix is not applicable.

The complete pathname for a default TLM is constructed from TLM pathway and default TLM description parameters. The pathway to the default TLM is specified by the *pathway* parameter in the TLM pathway sections (Table 24 on page 71). The default TLM filename is specified by the *root_name*, *TLtype*, and optionally the *list_of_phases* parameters in the default TLM description section (Table 25 on page 71). The *root_name* parameter, which is present in both the TLM pathway data and default TLM description section, links the relative pathway and default TLM filename together to form the complete default TLM pathname for a given *TLtype*.

Comments may be written at the end of a line of default TLM description parameters. A common practice is to identify the algorithm used to create the default TLM. A "#" does not need to precede end-of-the-line comments.

Table 26 lists examples of default TLM pathnames constructed from TLM pathway and default TLM description parameters. The Pathway column identifies the setting of the *pathway* parameter in a line of TLM pathway data. The Root Name, *TLtype*, Phase Depend, and Phase(s) columns identify the settings of the *root_name*, *TLtype*, *phase_depend*, and *list_of_phases* parameters, respectively, in a

line of default TLM description parameters. The Pathname(s) column identifies the complete pathname of the default TLM constructed from the TLM pathway and default TLM description sections.

TABLE 26: DEFAULT TLM PATHNAME EXAMPLES

Pathway	Root Name	TLtype	Phase Depend	Phase(s)	Pathname(s)
../global	qfvc	mb	0	P	../global/ qfvc.mb
../global	qfvc1	mb1	0	P, Pn	../global/ qfvc1.mb1
../regional/ mag_coefs	mcoefs_ def1	ml	0	P, Pn	../regional/ mag_coefs/ mcoefs_def1.ml
../global	global	ml	1	Pn, Pg, Sn, Lg	../global/ global.ml.Pn, ../global/ global.ml.Pg, ../global/ global.ml.Sn, ../global/ global.ml.Lg

The lines of station-specific TLM description parameters specify model, phase, and channel/frequency parameters needed to construct station-specific TLM filenames for a given *TLtype*. Table 27 identifies the format for a given line of station-specific TLM description parameters. Multiple TLM sections must be separated by an empty line. The end of the TLSF also indicates to the parser that the station-specific TLM description section is complete. The TLM description section may be appended with comment lines.

▼ Operational Procedures

TABLE 27: STATION-SPECIFIC TLM DESCRIPTION PARAMETERS

Name	Format	Description	Example
<i>sta</i>	%s	Station code.	BDFB
<i>Tltype</i>	%s	Transmission loss descriptor.	m1
<i>root_name</i>	%s	Root name of station-specific TLM filename.	mcoef_s_BDFB
<i>phase</i>	%s	Phase name.	-
<i>chan</i>	%s	Channel or frequency identifier.	-
<i>comments</i>	%*	Comments.	

A station-specific TLM takes precedence over a default TLM for the same *Tltype*. Each line of station-specific TLM description parameters corresponds to a unique station (*sta*) and *Tltype* combination. The parameters set in each line are used to construct the filename of a TLM specific to *sta* for the given *Tltype*. The station-specific transmission loss models may be phase- and channel/frequency-dependent. That is, a unique TLM may be specified for a particular phase (*phase*) and channel/frequency (*chan*) for a given *sta-Tltype* pair. The *chan* parameter may be used to identify a particular channel or a particular frequency band, whichever is more convenient.

The filename of a station-specific TLM is constructed from the *root_name*, *Tltype*, *phase*, and *chan* parameters. *root_name* specifies the root name, or first term, of the station-specific TLM filename. The format of the station-specific TLM filename is:

root_name.Tltype.phase.chan

The *phase* and *chan* suffixes are optional; they are only applied when station-specific TLMs are not required to be phase- and channel/frequency-dependent. If the station-specific TLM is independent of phase and channel/frequency, then set *phase* and *chan* to "-" in the corresponding line of the station-specific TLM description section. A station-specific TLM may be dependent on phase but independent of channel/frequency. In this case, specify *phase* and set *chan* to "-".

Finally, a station-specific TLM may be both phase- and channel/frequency-dependent. In this case, specify *phase* and *chan*. Station-specific TLMs are not allowed to be independent of phase and dependent on channel/frequency.

The complete pathname for a station-specific TLM is constructed from TLM pathway and station-specific TLM description parameters. The pathway to the station-specific TLM is specified by the *pathway* parameter in the TLM pathway section (Table 24 on page 71). The station-specific TLM filename is specified by the *root_name*, *Tltype*, and optionally *phase* and *chan* parameters in the station-specific TLM description section (Table 27 on page 74). The *Tltype* parameter links the station-specific TLM description with the default TLM description sections. The default *root_name* parameter links the default TLM description with the TLM pathway sections. These linkages connect the relative pathway with the station-specific TLM filename to form the complete station-specific TLM pathname for a given *Tltype*.

Comments may be written at the end of a line of station-specific TLM description parameters. A “#” need not precede end-of-the-line comments.

Table 28 lists examples of station-specific TLM pathnames constructed from TLM pathway and station-specific TLM description parameters. The Pathway column identifies the setting of the *pathway* parameter in a line of TLM pathway parameters. The Root Name, Tltype, Phase, and Chan columns identify the settings of the *root_name*, *Tltype*, *phase*, and *chan* parameters, respectively, in a line of station-specific TLM description parameters. The Pathname column identifies the complete pathname of the station-specific TLM constructed from the TLM pathway and station-specific TLM description sections.

▼ Operational Procedures

TABLE 28: STATION-SPECIFIC TLM PATHNAME EXAMPLES

Pathway	Root Name	Tltype	Phase	Chan	Pathname
../ regional/ mag_coefs	mcoefs_ILAR	ml	-	-	../regional/ mag_coefs/ mcoefs_ILAR.ml
../global	geress	ml	Pn	-	../global/ geress.ml.Pn
../global	geress	ml	Lg	inc2-4	../global /geress.ml. Lg.inc2-4

Transmission Loss Model

A Transmission Loss Model (TLM) is an ASCII file that contains earth-model data in the form of transmission loss and optional modeling error data for a given phase-dependent attenuation curve discretized by distance and depth. This document does not discuss how to create or discretize attenuation models or how to estimate modeling errors. However, this section does describe the format of a TLM so that a readable file may be created.

TLMs are probably more correctly termed transmission loss tables rather than transmission loss models, because a table is a physical parameterization used to represent a model. Nevertheless, in this document the names “transmission loss model” and “transmission loss table” are synonymous. In general, the term “transmission loss model” will be used to describe the physical file containing the transmission loss information.

A TLM has two sections. The first section contains transmission loss values and the second section contains optional modeling errors. Lines of comments are allowed before, after, and within the body of each section, but only in specific quantities and locations. In most cases, comment lines may begin with and contain any character. However, in some instances comment lines must begin with the “#” character. Blank lines are not permitted in a TLM. An example TLM is shown in Figure 14 for a P phase attenuation curve.

```
# Q-factor of P waves according to Veith-
Clawson (1972), revised by Maxwell, 02/99.
11 # number of depth samples
  0.00 15.00 40.00 100.00 200.00 300.00 400.00 500.00 600.00 700.00
800.00
181 # number of distance samples
  0.00 1.00 2.00 3.00 4.00 5.00 6.00 7.00 8.00 9.00
 10.00 11.00 12.00 13.00 14.00 15.00 16.00 17.00 18.00 19.00
 20.00 21.00 22.00 23.00 24.00 25.00 26.00 27.00 28.00 29.00
 30.00 31.00 32.00 33.00 34.00 35.00 36.00 37.00 38.00 39.00
 40.00 41.00 42.00 43.00 44.00 45.00 46.00 47.00 48.00 49.00
 50.00 51.00 52.00 53.00 54.00 55.00 56.00 57.00 58.00 59.00
 60.00 61.00 62.00 63.00 64.00 65.00 66.00 67.00 68.00 69.00
 70.00 71.00 72.00 73.00 74.00 75.00 76.00 77.00 78.00 79.00
 80.00 81.00 82.00 83.00 84.00 85.00 86.00 87.00 88.00 89.00
 90.00 91.00 92.00 93.00 94.00 95.00 96.00 97.00 98.00 99.00
100.00 101.00 102.00 103.00 104.00 105.00 106.00 107.00 108.00 109.00
110.00 111.00 112.00 113.00 114.00 115.00 116.00 117.00 118.00 119.00
120.00 121.00 122.00 123.00 124.00 125.00 126.00 127.00 128.00 129.00
130.00 131.00 132.00 133.00 134.00 135.00 136.00 137.00 138.00 139.00
140.00 141.00 142.00 143.00 144.00 145.00 146.00 147.00 148.00 149.00
150.00 151.00 152.00 153.00 154.00 155.00 156.00 157.00 158.00 159.00
160.00 161.00 162.00 163.00 164.00 165.00 166.00 167.00 168.00 169.00
170.00 171.00 172.00 173.00 174.00 175.00 176.00 177.00 178.00 179.00
180.00
```

... Transmission loss estimates at other depths ...

```
# Distance samples for all depths
101 1
  0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0
 10.0 11.0 12.0 13.0 14.0 15.0 16.0 17.0 18.0 19.0
 20.0 21.0 22.0 23.0 24.0 25.0 26.0 27.0 28.0 29.0
 30.0 31.0 32.0 33.0 34.0 35.0 36.0 37.0 38.0 39.0
 40.0 41.0 42.0 43.0 44.0 45.0 46.0 47.0 48.0 49.0
 50.0 51.0 52.0 53.0 54.0 55.0 56.0 57.0 58.0 59.0
 60.0 61.0 62.0 63.0 64.0 65.0 66.0 67.0 68.0 69.0
 70.0 71.0 72.0 73.0 74.0 75.0 76.0 77.0 78.0 79.0
 80.0 81.0 82.0 83.0 84.0 85.0 86.0 87.0 88.0 89.0
 90.0 91.0 92.0 93.0 94.0 95.0 96.0 97.0 98.0 99.0
100.0
# Distance-dependent modelling error for all depths
0.5000 0.5000 0.5050 0.4530 0.4140 0.3940 0.3780 0.3570 0.3400 0.3300
0.3120 0.2980 0.3000 0.3080 0.3080 0.3050 0.3000 0.2910 0.2790 0.2780
0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810
0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810
0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810
0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810
0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810
0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810
0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810 0.2810
0.2810
```

FIGURE 14. SAMPLE TLM

▼ Operational Procedures

The transmission loss data consist of sample depths, sample distances, and distance/depth-dependent estimates of transmission loss, as estimated from a phase-dependent attenuation curve. They are used to correct station magnitudes. Table 29 describes the format of the transmission loss data section of a TLM. Capital letters in the Line Number(s) column indicate variable line numbers and are defined in the detailed description of the transmission loss data.

TABLE 29: TRANSMISSION LOSS DATA FORMAT

Line Number(s)	Format	Description
1	%s	Comment line.
2	%d%*	Number of sample depths (=nz).
3 through A	$nz \times \%f$	Sample depths (km below sea level).
A+1	%d%*	Number of sample distances (=nx).
A+2 through B	$nx \times \%f$	Sample distances (arc degrees)
B+1	%s	Comment line that begins with "#".
B+2 through C	$nx \times nz \times \%f$	Transmission loss estimates (magnitude units).

Line 1 is ignored. A common practice is to identify the algorithm and any pertinent coefficients used to create the model and to specify the phase for which the model is valid.

Line 2 specifies the integer number of sample depths made on the attenuation curve. The number of sample depths (nz in Table 29 on page 78) may be followed by a short comment on the same line. This comment may begin with and contain any character.

Line 3 begins a list of depths at which the attenuation curve was sampled. All depths must be positive numbers. The nz sample depths may all be listed on one line, or spread across multiple lines. If the depths are all listed on one line, then the last line containing sample depths (A in Table 29) is 3. If the depths are spread

across m lines (where $1 < m \leq nz$), then A is $2 + m$. If multiple depths are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the transmission loss file.

Line $A + 1$ specifies the integer number of sample distances made on the attenuation curve. The number of sample distances (nx in Table 29) may be followed by a short comment on the same line. This comment may begin with and contain any character.

Line $A + 2$ begins a list of distances at which the attenuation curve was sampled for all nz depths. The nx sample distances may all be listed on one line or spread across multiple lines. If the distances are all listed on one line, then the last line containing sample distances (B in Table 29) is $A + 2$. If the distances are spread across n lines (where $1 < n \leq nx$), then B is $A + 1 + n$. If multiple distances are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the transmission loss data.

Line $B + 1$ is a comment line that separates the sample distances and depths from the transmission loss estimates. This line precedes the transmission loss values estimated at all nx distances for the first sampled depth on the attenuation curve. The comment line must begin with the “#” character. A common practice is to indicate this depth in the comment line.

Line $B + 2$ begins the nz sets of transmission loss estimates—one set for each sampled depth. Each set of transmission loss estimates (or magnitude corrections) is composed of nx estimates of the attenuation curve—one estimate for each sampled distance. The transmission loss estimates must be listed in the same order as the sample distances. The estimates for a particular sample depth may all be listed on one line or spread across multiple lines. If multiple estimates are present on a given line, then they must be separated by at least one space. The alignment of the transmission loss estimates should mimic that of the sample distances (lines $A + 2$ through B). Although not a requirement, alignment makes the association between distance and transmission loss more readable in the file.

▼ Operational Procedures

Multiple sets of transmission loss estimates must be listed in the same order as the corresponding sample depths. Each set must be separated by a comment line that begins with the “#” character. A common practice is to indicate the sample depth for the upcoming set of transmission loss estimates in the comment line. No other comment lines are allowed in this portion of the transmission loss file. The last line in a TLM containing transmission loss estimates is line C.

The transmission loss data are completely specified at this point in a TLM. The remainder of the TLM specifies any modeling error data. Modeling error data are optional. If there are no modeling error data for the attenuation curve in a TLM, then the TLM is complete.

The modeling error data consist of sample depths, sample distances, and distance/depth-dependent estimates of the standard error (standard deviation) associated with the transmission loss estimates from a phase-dependent attenuation curve. The modeling errors specified in a TLM may be either a single-value, distance-dependent values, or distance/depth-dependent values. This choice dictates the format of the modeling error data. Any of these formats may be employed regardless of whether the transmission loss estimates themselves are distance/depth-dependent or distance-dependent only. Modeling errors may not be depth-dependent only. The modeling error data immediately follow the transmission loss data in a TLM without interruption by a blank line or text. See Figure 14 for an example of a TLM with distance-dependent modeling error data.

The first format for specifying modeling error data in a TLM is the single-valued format. This format is employed when the modeling error is single-valued across all sample distances and depths of the attenuation curve. Table 30 describes the single-valued format.

TABLE 30: SINGLE-VALUE TRANSMISSION LOSS MODELING ERROR DATA FORMAT

Line Number(s)	Format	Description
C + 1	%s	Comment line.
C + 2	%d%d%*	Number of sample distances (=1) and number of sample depths (=1).
C + 3	%s	Comment line.
C + 4	%f%*	Single-value transmission loss modeling error (standard error estimate of transmission loss).

Line C + 1 of the modeling error section contains a single comment line. This comment line must be placed immediately following the last set of transmission loss estimates. It may begin with and contain any character. A common practice is to identify the start of the modeling error section or to indicate that the modeling error is single-valued.

Line C + 2 consists of two "1" indicators separated by spaces. These indicators signify that a single modeling error will be associated with all transmission loss estimates read from the TLM. The indicators may be followed by a short comment on the same line. The comment may begin with and contain any character.

Line C + 3 is a comment line that separates the single-valued modeling error indicators from the single modeling error itself. This comment line may begin with and contain any character. A common practice is to only include the "#" character in this comment line.

Line C + 4 specifies the single modeling error. This modeling error will be associated with any transmission loss estimate read from the transmission loss data section, regardless of distance and depth.

▼ Operational Procedures

The second format for specifying modeling error data in a TLM is the distance-dependent format. This format is employed when the modeling error is distance-dependent only over the entire attenuation curve. Table 31 describes the distance-dependent format. The acronym “EOF” in Table 31 stands for “end-of-file”.

TABLE 31: DISTANCE-DEPENDENT TRANSMISSION LOSS MODELING ERROR DATA FORMAT

Line Number(s)	Format	Description
C + 1	%s	Comment line.
C + 2	%d%d%*	Number of sample distances (=mx) and number of sample depths (=1).
C + 3 through D	mx × %f	Sample distances (arc deg).
D + 1	%s	Comment line.
D + 2 through EOF	mx × %f	Transmission loss modeling errors (standard error estimates of transmission loss).

Line C + 1 of the modeling error section contains a single comment line. The comment line must be placed immediately following the last set of transmission loss estimates. It may begin with and contain any character. A common practice is to identify the start of the modeling error section or to indicate that the modeling error is distance-dependent only.

Line C + 2 specifies the integer number of sample distances used to estimate the modeling error of the attenuation curve. The number of sample distances should be followed by a space and a “1”. The “1” signifies that the modeling errors are independent of depth. These two values may be followed by a short comment on the same line. The comment may begin and end with any character.

Line C + 3 marks the beginning of a list of distances at which the modeling errors were estimated. The mx sample distances may differ from those used to produce transmission loss estimates that have a distance dependency. The sample distances may all be listed on one line, or spread across multiple lines. If the distances are all

listed on one line, then the last line containing sample distances (D in Table 31) is $C + 3$. If the distances are spread across m lines (where $1 < m \leq mx$), then D is $A + 2 + m$. If multiple distances are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the modeling error file.

Line $D + 1$ is a comment line that separates the sample distances from the distance-dependent modeling errors. This comment line may begin with and contain any character. A common practice is to indicate that the modeling errors are distance-dependent.

Line $D + 2$ begins the list of modeling errors. The mx modeling errors must be listed in the same order as the corresponding sample distances. The modeling errors may be listed on one line, or spread across multiple lines. If multiple modeling errors are present on a given line, then they must be separated by at least one space. The alignment of the modeling errors should mimic that of the sample distances (lines $C + 3$ through D). Although not a requirement, alignment makes the association between distance and modeling error more readable in the file. Comment lines are not allowed in this portion of the modeling error file.

The third format for specifying modeling error data in a TLM is the distance/depth-dependent format. This format is employed when the modeling error is both distance- and depth-dependent over the entire attenuation curve. Table 32 describes the distance/depth-dependent format.

TABLE 32: DISTANCE/DEPTH-DEPENDENT TRANSMISSION LOSS MODELING ERROR DATA FORMAT

Line Number(s)	Format	Description
$C + 1$	%s	Comment line.
$C + 2$	%d%d%*	Number of sample distances ($=mx$) and number of sample depths ($=mz$).
$C + 3$ through D	$mx \times \%f$	Sample distances (arc deg).

▼ Operational Procedures

TABLE 32: DISTANCE/DEPTH-DEPENDENT TRANSMISSION LOSS MODELING ERROR DATA FORMAT (CONTINUED)

Line Number(s)	Format	Description
$D+1$ through E	$mz \times \%f$	Sample depths (km below sea level).
$E+1$	$\%s$	Comment line that begins with "#".
$E+2$ through EOF	$mx \times mz \times \%f$	Transmission loss modeling errors (standard error estimates of transmission loss).

Line $C + 1$ of the modeling error section contains a single comment line. The comment line must be placed immediately following the last set of transmission loss estimates. It may begin with and contain any character. A common practice is to identify the start of the modeling error section or to indicate that the modeling error is distance/depth-dependent.

Line $C + 2$ specifies the integer number of sample distances and number of sample depths used to estimate the modeling error of the attenuation curve. These two values may be followed by a short comment on the same line. This comment may begin and contain any character.

Line $C + 3$ begins a list of distances at which the modeling errors were estimated for all mz depths. The mx sample distances may differ from those used to produce transmission loss estimates that have a distance dependency. The sample distances may all be listed on one line, or spread across multiple lines. If the distances are all listed on one line, then the last line containing sample distances (D in Table 32) is $C + 3$. If the distances are spread across m lines (where $1 < m \leq mx$), then D is $C + 2 + m$. If multiple distances are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the modeling error file.

Line $D + 1$ begins a list of depths at which the modeling errors were estimated. All depths must be positive numbers. The mz sample depths may differ from those used to produce transmission loss estimates that have a depth dependency. The sample depths may all be listed on one line, or spread across multiple lines. If the

depths are all listed on one line, then the last line containing sample depths (E in Table 32) is $D + 1$. If the depths are spread across n lines (where $1 < n \leq mz$), then E is $D + n$. If multiple depths are present on a given line, then they must be separated by at least one space. Comment lines are not allowed in this portion of the modeling error file.

Line $E + 1$ is a comment line that separates the sample distances and depths from the distance/depth-dependent modeling errors. This comment line must begin with the “#” character. A common practice is to indicate the first sample depth in the comment line.

Line $E + 2$ begins the mz sets of modeling errors; one set for each sampled depth. Each set is composed of mx modeling errors; one modeling error for each sampled distance. The modeling errors must be listed in the same order as the corresponding sample distances. The modeling errors for a particular sample depth may all be listed on one line, or spread across multiple lines. If multiple modeling errors are present on a given line, then they must be separated by at least one space. The alignment of the modeling errors should mimic that of the sample distances (lines $C + 3$ through D). This is not a requirement, but alignment makes the association between distance and modeling error more readable in the file.

Multiple sets of modeling errors must be listed in the same order as the corresponding sample depths. Each set must be separated by a comment line that begins with the “#” character. A common practice is to indicate the sample depth for the upcoming set of modeling errors in the comment line. No other comment lines are allowed in this portion of the modeling error data. The TLM file should terminate after the last line of modeling error data.

Magnitude Test-site Correction File

Like their travel-time counterparts, Magnitude Test-site Correction (MTSC) files are ASCII files that contain corrections to computed station magnitude values for events located in certain localized (nuclear) test-site regions. Magnitude test-site files always accompany a corresponding TLM file. As individual TLM files are *magtype*-specific, so are MTSC files. If a MTSC file exists for a particular TLM and

▼ Operational Procedures

phase, its full path and filename must be specified in a file named `<TLM>.<magtype>.ts_dir`, located in the TLM directory. As such, the location and name of the actual test-site file are not strictly controlled, but the `<TLM>.<magtype>.ts_cor` naming convention should be followed for consistency.

The correction data in a particular MTSC file are organized by region and station in a format nearly identical to the LTSC file. Table 33 lists the MTSC file format. No comments are allowed on separate lines in the main body of the file. Comments may be placed at the end of a given line, after all the data fields, or at the end of the file.

TABLE 33: MTSC FILE FORMAT

Line Number(s)	Format	Description
1	%d%*	Number of test-site regions in this file.
2	%d%s%s%d%*	Region sequence number, Region name (up to 9 chars), <i>Magtype</i> , Number of stations with corrections for this region/ <i>magtype</i> .
3 thru 3*, where 3* = #stations + 2	%6s%lf	Station name, correction (in magnitude units).
4 thru EOF	multiple formats	Repeat line 2 and corresponding station lines as in 3 thru 3* for each region, corresponding to the number in line 1.

Line 1 of the file is the total number of test-site regions for which one or more station corrections are specified. Line 2 specifies region information in four fields: the region number (that is, the sequence number as it appears in the file), the station name, the *magtype*, and the number of stations for which corrections are specified.

Following this line, the corrections are listed on individual lines by station. The region line and corresponding station lines are repeated for each region, where the total number of these groups matches the number in the first line of the file. Figure 15 shows a sample MTSC file for region names "AA" and "BB".

```

2      # Number of distinct regions with magnitude test-site corrections
1      AA      mb      5      # region num, name ID, TLtype, num stas
      PDY      2.0
      ARU      2.1
      CMAR     2.2
      GBA      2.3
      KSAR     2.4
2      BB      mb      3      # region num, name ID, TLtype, num stas
      FINES    2.5
      ARCES    2.6
      HFS      2.7

```

FIGURE 15. SAMPLE MTSC FILE

During magnitude processing with *libmagnitude*, the magnitude test-site corrections are accessed via the test-site region name value. For example in *EvLoc*, the *mag_ts_region* parameter controls the corrections that are applied by *libmagnitude*.

MAINTENANCE

The following maintenance activities may have to be performed: purging *EvLoc* log files, maintaining database integrity, integrating new model data into the operational filesystem, and adding new stations to the station network.

Purging Log Files

EvLoc diagnostic output is commonly redirected to a log file. The IDC operational system stores these log files in a directory along with log files from other applications. All log files, including those created by *EvLoc*, are purged regularly.

Maintaining Database Integrity

At the IDC, the lead analyst maintains the integrity of the operational database through regular execution of the *rebrevis* Perl script, which invokes *EvLoc* to recompute suspect magnitudes.

▼ **Operational Procedures****Integrating New Models**

Occasionally, new models such as regional travel-time or transmission loss data must be integrated into the operational system. “Earth-model File Setup” on page 33 describes how to construct and format configuration and earth-model files, and “Input Files” on page 149 describes how to configure and install these files. These new files should be stored in the filesystem with other files of the same type. The creation and modification of input files necessitated by the addition of new stations to a network are discussed in the next section.

Adding New Stations

The addition of a new station to an operational network requires updates to static database tables and optional modification of several location and magnitude input files.

Updating Static Database Tables

Insert a row into the **site** and **siteaux** tables for the new station. Insert a row into the **affiliation** table for each network containing the new station. See [IDC5.1.1Rev3] for the database schema.

Modifying Location Input Files

The VMSF may be optionally modified to include station-specific information for the new station. SSSC tables, an SASC file, and radial 2-D travel-time tables may be created to store travel-time and slowness/azimuth correction data for the new station. In addition, LTSC files may be modified to include station-specific corrections. However, it is very unlikely that test-site corrections would be available for a new station. Unless nuclear tests are recorded at a station, no data exist to generate the LTSC entry.

One or more lines may be added to the station/phase/model-specific travel-time correction portion of the VMSF. If at least one such line is added for the new station, then 2-D regional travel-time corrections may be applied for that station by

adding one or more station/phase SSSC tables to the SSSC directory. See “Velocity Model Specification File” on page 35 and “Source-specific Station Correction Table” on page 49 for format and content information about these files.

A single SASC file may be created to store slowness/azimuth correction knowledge for the new station. See “Slowness/Azimuth Station Correction File” on page 60 for format and content information about SASC files.

One or more 2-D travel-time tables may be created if the new station is a hydroacoustic or infrasonic station. These tables store 2-D hydroacoustic or infrasonic travel-time data. A number of steps are necessary to configure the system so that the new 2-D travel-time tables may be used operationally. See “Radial 2-D Travel-time Tables” on page 48 for configuration details as well as for descriptions of the format and content of the tables.

Modifying Magnitude Input Files

The MDF and TLSF may be optionally modified to include station-specific information for the new station. One or more TLMs may be created to store transmission loss data for the new station. In addition, MTSC files also may be updated to include station-specific correction data.

One line of bulk magnitude station correction parameters may be added to the MDF for the new station for each distinct *magtype*. See “Magnitude Description File” on page 64 for format and content information about the MDF.

One line of station-specific TLM description parameters may be added to the TLSF for the new station for each distinct *Tltype*. This is particularly useful for estimating local or regional magnitudes. If one such line is added for the new station, then a properly-named TLM file must be created and installed in the appropriate directory. See “Transmission Loss Specification File” on page 69 and “Transmission Loss Model” on page 76 for format and content information about the TLSF and TLMs as well as for configuration details related to the two types of files.

▼ Operational Procedures

SECURITY

EvLoc requires an ORACLE user identification and password to establish a connection to an ORACLE database account. Passwords are created and modified by the ORACLE database administrator.

Chapter 3: Troubleshooting

This chapter describes how to identify and correct problems related to the Event Location and Magnitude and includes the following topics:

- Monitoring
- Interpreting Error Messages
- Solving Common Problems
- Reporting Problems

Chapter 3: Troubleshooting

MONITORING

This section describes techniques for validating proper installation and operation of the *EvLoc* software:

- studying the *WorkFlow* processing status display
- querying the operational database
- examining *EvLoc* log files

Studying Processing Status

The *WorkFlow* monitoring tool allows processing engineers and managers to visualize the progress of pipeline processing. At the IDC, *WorkFlow* is used to monitor the progress of Station Processing, Network Processing, Post-location Processing, and Post-analysis Processing intervals. See [IDC6.2.1] and [IDC6.5.2Rev0.1] for more information about *WorkFlow*.

The *WorkFlow* display uses rectangular 'bricks' to represent individual processing time intervals. The color of each brick indicates the processing state of a given time interval. Four processing states are used at the IDC to monitor *EvLoc* processing: *EvLoc-started*, *EvLoc-retry*, *EvLoc-done*, and *EvLoc-failed*. If *EvLoc* is installed and operating properly, a *WorkFlow* display should consistently show green bricks for each time interval processed. These green bricks represent an *EvLoc-done* state and indicate that *EvLoc* successfully completed processing of the corresponding interval. Red bricks represent an *EvLoc-failed* state and indicate a problem with *EvLoc* processing. The operator should report any occurrences of these failed states and attempt to manually reprocess the failed intervals.

Querying the Database

The location and magnitude estimates made by the Event Location and Magnitude software may be monitored by querying the operational database after *EvLoc* has finished processing one or more time intervals.

If using *EvLoc* to compute event locations, query the **origin**, **origerr**, and **assoc** tables from the operational database account for location, error, and association statistics. If using *EvLoc* to estimate event magnitudes, query the **origin**, **stamag**, and **netmag** tables from the operational database account for magnitude and uncertainty statistics.

Examining Log Files

Messages written by *EvLoc* may be redirected to a log file, which is located in an appropriate directory in the operational file system. The log file records the progress of the processing and any warnings or errors that occur during processing. Monitor the log file to ensure that *EvLoc* operates as intended. The log file is useful for obtaining additional information about a problem identified in *Workflow*. Information about logged error messages are given in the section below.

INTERPRETING ERROR MESSAGES

The more commonly encountered *EvLoc* error messages are usually descriptive enough to diagnose a problem without the aid of a separate reference. For completeness however, this section includes descriptions and solutions for all the error and warning messages related to parameter, configuration, and data inconsistencies that may be remedied by the user. There are other errors indicative of system-level problems, such as those related to memory allocation, that are not included in this document. If memory allocation errors do occur, run the process on a machine with more memory, or close other applications running at the same time. Other system-level errors will likely not originate in *EvLoc*, and may not have simple solutions, but still may be diagnosable based on the message printed.

▼ Troubleshooting

The next section is an index of all *EvLoc* errors and warning messages, ordered alphabetically by the first line of the output message. Following that, the full descriptions of the errors are organized in two main categories, those related to configuration problems, and those related to computational or processing problems. Within each of these sections, the errors are listed by the part of the code from which they originate; that is, in *EvLoc* itself, or in *libloc* or *libmagnitude*. Within each error listing section, the ordering of the errors is based on their sequence in the flow of processing.

Some error messages contain variables of the form <VAR>. These variables represent one of two or more possible values. The possible values are given in the "Description" section of the error listings.

Alphabetical Error Index

This section lists all *Evloc* error and warning messages, ordered alphabetically, and shows the page number of the detailed description

Message	Page
best_guess: Too few data to get an initial location	139
Bogus event_control->src_dpnt_corr level: <LEVEL> specified for orid: <ORID>! Exiting!!!	108
EM ESTIMATOR HAS NOT CONVERGED AFTER <N> ITERATIONS!	145
Error 26: Requested test-site region not available!	120
ERROR! Acoustic travel time requested without epoch time set. Will use <SEASON> tables as default!	138
Error encountered while attempting to read mag. test-site corr. file:	134
Error: problem with number of <DATA> values in SSSC file for sta <STA>, phase <PHASE>	116
Error: problem with number of <ATTRIBUTE> samples in SSSC file for sta <STA>, phase <PHASE>	116
Error: read_LP_info: File: <FILE> is missing!	112

Message	Page
ERROR: set_sta_pt: site table not specified or empty!	117
Error: setup_tttables: Null phase list!	111
ERROR: Travel time requested for station not in 2-D acoustic tables.	120
Fatal error: <PHASE/STATION>: <VALUE> in SSSC file is inconsistent with filename!!	114
Fatal error: While trying to read current SSSC file: <FILE>	114
get_LP_velocity: Input period of <PERIOD> sec. is out-of-range!	120
If you wish to write results to input database then input and output database accounts must be identical! They are not!! Exiting!!!	104
Incorrect LB, UB, or BL value for magtype <MAGTYPE> in MDF:	124
Incorrect number of arguments per magtype in MDF: expecting <N1>, found <N2>	124
Incorrect number of arguments per sta/TLtype in MDF file: expecting <N1>, found <N2>	124
interp_for_tl_value: tl_index is too large!!!	143
Locator: Bad input assoc data!	121
Locator: Bad input origerr data!	122
Locator: Bad input origin data!	121
Locator: Divergent solution encountered!	142
Locator: Insufficient defining data before location is even attempted!	139
Locator: Insufficient defining data remains due to hole in T-T table(s)!	141
Locator: Insufficient defining data remains due to large residual restriction!	140

▼ Troubleshooting

Message	Page
Locator: Insufficient defining data remains due to TT correction restriction!	141
Locator: Insufficient defining data remains due to T-T table extrapolation!	142
Locator: Insufficient defining data remains while attempting to locate event!	140
Locator: Lat/Lon value(s) out of range given a fixed lat/lon!	122
Locator: Maximum number of iterations exceeded!	143
Locator: Mismatch between input arrival/assoc data!	138
Locator: SVD routine cannot decompose given matrix!	142
Locator: Warning - No observations to process!	121
<MAGTYPE> weighted average desired, but no <STA> station weights available!	137
mag_boot_strap: Cannot generate epoch time for random number generation!	144
Magtype: <MAGTYPE> is not specified within MDF	135
MDF is empty, so no magnitude info can be computed	123
Message: No SASC tables can be read since no directory/prefix is specified!	118
Message: No SASC tables can be read since no / in SASC directory name!	118
mstpar: must specify value for '<PAR>', expecting <TYPE>	105
Number of stations in sub-station list > NUM_SUB_STA!! Must re-declare getpar size in read_evloc_par()!!!	105
Problems encountered reading travel-time info. Exiting!!	110
Problems encountered while trying to read magnitude info. Exiting!!	123

Message	Page
Problems encountered while trying to read SASctables. Exiting!!	118
read_evloc_db_tables: Error <ERR_CODE>: '<ERR_STRING>' No site data available from table <SITE_TABLE>	107
read_evloc_db_tables: Error <ERR_CODE>: '<ERR_STRING>' while trying to access <FEATURE> table with query: <QUERY>	106
read_evloc_db_tables: Error <ERR_CODE>: '<ERR_STRING>' while querying db using query=<QUERY>	107
read_evloc_db_tables: gdi_open() failed!! Error <ERR_CODE>: '<ERR_STRING>'	106
read_evloc_db_tables: No <TABLE_TYPE> data available from table '<TABLE>' for orid <ORID>	108
read_LP_info: Error reading <DATA> in file: <FILE>!	112
read_mdf: MDF: <MDF> will not open!	123
read_sasc: No . in SASC file name!	119
read_sasc: Number of az/slow corr bins found: <B1> dis- agrees with the <B2> specified in file: <FILE> Will continue, but this should be fixed!!!!	119
read_sasc: Number of az/slow corr bins found is > spec- ified in file: <FILE> This will overrun memory, so bailing out!!!!	119
read_tlsf: Error reading station/TLtype/model info in file:	127
read_tlsf: Error! STM: <STA>/<TLTYPE>/<ROOT> line not associated with any tltype_model_descrip TLtype defi- nition!	128
read_tlsf: No / in TL model filename!	126
read_tlsf: No TL tables could be opened!	129
read_tlsf: site table not specified or empty!	125
read_tlsf: TL model not specified for: <TLM> for TLtype:<TLTYPE>	126

▼ Troubleshooting

Message	Page
read_tlsf: TLSF: <TLSF> will not open!	126
read_tlsf: Warning! STM: <STA>/<TLTYPE>/<ROOT> line found to be a duplicate with another STM record!	128
read_tlsf: Warning! STM: <STA>/<TLTYPE>/<ROOT> line found to be redundant with info specified in tlttype_model_descrip!	127
read_tl_table: Error reading depth sample value in file: <TLM>	130
read_tl_table: Error reading distance-dependent modelling error in file: <TLM>	133
read_tl_table: Error reading distance/depth modelling error in file: <TLM>	134
read_tl_table: Error reading distance sample value in file: <TLM>	130
read_tl_table: Error reading modelling error depth sample value in file: <TLM>	133
read_tl_table: Error reading modelling error distance sample value in file: <TLM>	132
read_tl_table: Error reading number of depth samples in file: <TLM>	129
read_tl_table: Error reading number of distance samples in file: <TLM>	130
read_tl_table: Error reading premature EOF in file: <TLM>	131
read_tl_table: Error reading single bulk modelling error in file: <TLM>	132
read_tl_table: Error reading transmission-loss value in file: <TLM>	131
read_tt_tables: Error reading <DATA> in file <FILE>	112
read_tt_tables: No / in <DATA> directory name!	111
read_tt_tables: No / in ellipticity correction directory name!	113

Message	Page
read_tt_tables: No tables could be opened!	113
read_tt_tables: Problems encountered while trying to read <DATA> tables!	111
read_tt_tables: Velocity model not specified for: <VMODEL> in station/phase/model listing	113
read_tt_tables: VMSF: <FILE> will not open!	111
Requested mag. test-site region: <REG>, not available!	135
Station DFAULT not found in MDF file station correction list for TLtype <TLTYPE>.	125
trv_time_w_ellip_elev: phase_index is too large!!!	138
Usage: EvLoc par=par_filename	101
User cannot request triple locations and magnitudes only!!! Re-set mode and triple_location par file arguments accordingly! Exiting !!!	105
User CANNOT request using event_control tables and not write a new one! Please adjust par file appropriately!! Exiting!!!	104
User CANNOT write ar_info table if only magnitudes are desired! Please adjust par file appropriately!! Exiting!!!	101
User CANNOT write to input database tables if output ar_info table is requested! Please adjust par file appropriately!! Exiting!!!	102
User CANNOT write to input database tables if synthetic data is requested! This is simply too dangerous!! Exiting!!!	103
User CANNOT write to input database tables if triple locations are requested, since this is a many-to-one relationship! Please adjust par file appropriately!! Exiting!!!	102
Warning: Bulk station correction in SSSC file is inconsistent with velocity model spec. file value for station: <STA> phase: <PHASE>	115

▼ Troubleshooting

Message	Page
Warning: File <TLM> will not open!	129
Warning: Invalid magtype, <MAGTYPE>, specified in attempt to reset	136
Warning: Invalid magtype, <MAGTYPE>, specified in attempt to revert	136
Warning: Network stdev = <STDEV> < lower bound in mdf file = <LB> ->	144
Warning: Network stdev = <STDEV> > upper bound in mdf file = <UB> ->	144
Warning: Sed. velocity in SSSC file is inconsistent with velocity model spec. file value for station: <STA> phase: <PHASE>	115
Warning: SSSC file zero boundary value violation. For more details run EvLoc in mode 3	117
Warning: Sum of individual LP segments differ by > 0.1% from original distance!	137
write_evloc_db_tables: Error <ERR_CODE>: '<ERR_STRING>'. DB error found while trying to delete <FEATURE> records from <TABLE> using query: <QUERY>	109
write_evloc_db_tables: Error <ERR_CODE>: '<ERR_STRING>' while attempting to add <FEATURE> records to <TABLE>	110
write_evloc_db_tables: gdi_get_counter failed trying to get <ID>s!! Error <ERR_CODE>: '<ERR_STRING>'	109
write_evloc_db_tables: gdi_open() failed!! Error <ERR_CODE>: '<ERR_STRING>'	108
You CANNOT calculate magnitudes when creating synthetic data, so, EvLoc will automatically ignore magnitudes!!	103
You CANNOT create synthetic data in triple location mode!! EvLoc will automatically reset to single location mode!!	103

EvLoc Configuration-related Error Messages

Message: Usage: EvLoc par=par_filename

Description: No input control parameters specified. Commonly, input control parameters are set in a par file, which is specified on the command line.

Action: Run *EvLoc* again, but this time add the command line argument `par=par_filename`, where *par_filename* is the filename of the par file that contains the input control parameter settings.

Message: User CANNOT write ar_info table if only magnitudes are desired! Please adjust par file appropriately!!
Exiting!!!

Description: Inconsistency between *write_ar_info_table* and *mode* parameters. **ar_info** records can only be written when *EvLoc* is run in location mode.

Action: If computing magnitudes only, then comment out the *write_ar_info_table* parameter in the input par file. If computing event locations, then set the *mode* parameter to "0" (compute event locations only) or "1" (compute both event locations and magnitudes).

▼ Troubleshooting

Message: User CANNOT write to input database tables if output *ar_info* table is requested! Please adjust par file appropriately!! Exiting!!!

Description: Inconsistency between the *write_to_input_db_tables* and *write_ar_info_table* parameters. *ar_info* records can not be written to an input database table because the *ar_info* table is an output table only.

Action: If creating output *ar_info* records, then comment out the *write_to_input_db_tables* parameter in the input par file. Also ensure that the output database tables exist in the output database account and that their table names are specified correctly in the par file.

If not creating output *ar_info* records, then comment out the *write_ar_info_table* parameter.

Message: User CANNOT write to input database tables if triple locations are requested, since this is a many-to-one relationship! Please adjust par file appropriately!! Exiting!!!

Description: Inconsistency between *write_to_input_db_tables* and *triple_location* parameters. Multiple event locations can not be written to input database tables.

Action: If computing triple locations, then comment out the *write_to_input_db_tables* parameter in the input par file. Also ensure that the output database tables exist in the output database account and that their table names are specified correctly in the par file.

If not attempting to compute triple locations, then comment out the *triple_location* parameter.

Message: User CANNOT write to input database tables if synthetic data is requested! This is simply too dangerous!! Exiting!!!

Description: Inconsistency between the *write_to_input_db_tables* and *create_syn_data_only* parameters. Synthetic event and detection data can only be written to output database tables to avoid contaminating or overwriting the source data.

Action: Set the parameter *write_to_input_db_tables* to false, and specify values for output table parameters. Be aware that *SynGen* should be used instead of *EvLoc* for computing synthetic travel-times.

Message: You CANNOT create synthetic data in triple location mode!! EvLoc will automatically reset to single location mode!!

Description: Inconsistency between the *create_syn_data_only* and *triple_location* parameters.

Action: No action is necessary. *EvLoc* automatically resets to single location mode. Be aware that *SynGen* should be used instead of *EvLoc* for computing synthetic travel-times.

Message: You CANNOT calculate magnitudes when creating synthetic data, so, EvLoc will automatically ignore magnitudes!!

Description: Inconsistency between the *mode* and *create_syn_data_only* parameters.

Action: No action is necessary. *EvLoc* automatically does not compute synthetic amplitudes or magnitudes. Be aware that *SynGen* should be used instead of *EvLoc* for computing synthetic travel-times.

▼ Troubleshooting

Message: User CANNOT request using `event_control` tables and not write a new one! Please adjust par file appropriately!! Exiting!!!

Description: Parameter inconsistency. If the `use_ev_cntrl_table` parameter is set, then the `write_ev_cntrl_table` must also be set.

Action: If intending to use `event_control` attributes to compute event locations and magnitudes, then uncomment the `write_ev_cntrl_table` parameter in the input par file. Also ensure that the `new_event_control_table` parameter specifies an existing `event_control` table in the output database account.

If not intending to use `event_control` attributes, then comment out the `use_ev_cntrl_table` parameter.

Message: If you wish to write results to input database then input and output database accounts must be identical! They are not!! Exiting!!!

Description: Inconsistency between the `write_to_input_db_tables`, `in_db_account`, and `out_db_account` parameters.

Action: If intending to write output records to the input database tables, then set the `in_db_account` and `out_db_account` parameters to the same database account in the input par file.

If intending to write output records to output database tables that have different names from the input database tables, then comment out the `write_to_input_db_tables` parameter and ensure that the `in_db_account` and `out_db_account` parameters are set to the proper database accounts. Also ensure that the output database tables exist in the output database account and that their table names are specified correctly in the par file.

Message: Number of stations in sub-station list >
NUM_SUB_STA!! Must re-declare getpar size in
read_evloc_par()!!!

Description: The list of stations in the *sub_sta_list* or *mag_sub_sta_list* parameter is too long.

Action: If fewer stations may be used to compute event locations or magnitudes, then remove unnecessary stations from the *sub_sta_list* or *mag_sub_sta_list* parameter in the input par file. If all stations in the *sub_sta_list* or *mag_sub_sta_list* parameter must be used to compute event locations or magnitudes, then contact the maintainer of the software.

Message: User cannot request triple locations and magnitudes only!!! Re-set mode and triple_location par file arguments accordingly! Exiting !!!

Description: Inconsistency between the *triple_location* and *mode* parameters.

Action: If intending to compute magnitudes, then comment out the *triple_location* parameter in the input par file. If intending to compute multiple event locations, then set the *mode* parameter to "0" (compute event locations only) or "1" (compute both event locations and magnitudes).

Message: mstpar: must specify value for '<PAR>', expecting
<TYPE>

Description: The required parameter <PAR> of variable type <TYPE> was not set in the input par file. <TYPE> is either "an integer", "a string", "a float", "a double", or "a boolean".

Action: Set the required parameter in the input par file.

▼ Troubleshooting

Message: read_evloc_db_tables: gdi_open() failed!! Error
<ERR_CODE>: '<ERR_STRING>'

Description: The *EvLoc* function that reads from the database could not establish a connection with the input database account. Error code <ERR_CODE> and message text <ERR_STRING> returned from the database interface provide additional information about the error.

Action: Ensure that the account, password, and database strings are set correctly in the *in_db_account* parameter in the *EvLoc* par file. If the settings are correct, then talk to the database administrator.

Message: read_evloc_db_tables: Error <ERR_CODE>:
'<ERR_STRING>' while trying to access <FEATURE>
table with query: <QUERY>

Description: Data could not be read from the input <FEATURE> database table using the query <QUERY>. <FEATURE> is either "origin", "assoc", "arrival", "parrival", "det_amplitude", "ev_amplitude", "netmag", "stamag", or "event_control". Error code <ERR_CODE> and message text <ERR_STRING> returned from the database interface provide additional information about the error.

Action: Verify that the <FEATURE> table exists in the input database account. Also verify that <QUERY> includes the correct database table name(s).

If an incorrect table name is identified in the error message, then change the corresponding input table name parameter setting in the input par file.

Message: read_evloc_db_tables: Error <ERR_CODE>:
'<ERR_STRING>' while querying db using query=<QUERY>

Description: Station information could not be read from the input **site** and **affiliation** tables using query <QUERY>. Error code <ERR_CODE> and message text <ERR_STRING> returned from the database interface provide additional information about the error.

Action: Verify that the **site** and **affiliation** table names listed in <QUERY> are correct. Also verify that the station network specified in <QUERY> is correct.

If an incorrect table name or station network is identified in the error message, then change the setting of the *site_table*, *affiliation_table*, or *network* parameters in the input par file.

Message: read_evloc_db_tables: Error <ERR_CODE>:
'<ERR_STRING>' No site data available from table
<SITE_TABLE>

Description: **site** records do not exist in the **site** table <SITE_TABLE> for the requested station network and time period. The time period is defined as the time difference between the latest and earliest origins read from the input **origin** table. Error code <ERR_CODE> and message text <ERR_STRING> returned from the database interface provide additional information about the error.

Action: If the **site** table does not contain the desired station sites, then add **site** records to the table specified in the *site_table* parameter in the input par file, or change the *site_table* setting to point to one that does contain the desired **site** records.

If an incorrect network is identified, then change the setting of the *network* parameter.

If an incorrect time period is identified, then change the *origin_query* parameter so that it specifies the correct start and end times.

▼ Troubleshooting

Message: read_evloc_db_tables: No <TABLE_TYPE> data available from table '<TABLE>' for orid <ORID>

Description: Records do not exist in the input <TABLE_TYPE> (either ASSOC or ARRIVAL) database table <TABLE> for the orid <ORID>.

Action: Add the missing **assoc/arrival** records to the table specified in the corresponding input table name parameter in the input par file, or change the setting of the parameter to point to a **assoc/arrival** table that does contain the missing **assoc/arrival** records for the orid.

If the missing **assoc/arrival** records can not be found, then rewrite the query specified in the *origin_query* parameter to ignore processing of the orid.

Message: Bogus event_control->src_dpnt_corr level: <LEVEL> specified for orid: <ORID>! Exiting!!!

Description: The **event_control.src_dpnt_corr** value is out of range for the specified orid.

Action: Set the *src_dpnt_corr* level in the **event_control** record associated with <ORID> so that <LEVEL> is between "0" and "4".

Message: write_evloc_db_tables: gdi_open() failed!! Error <ERR_CODE>: '<ERR_STRING>'

Description: The *EvLoc* function that writes to the database could not establish a connection with the output database account. Error code <ERR_CODE> and message text <ERR_STRING> returned from the database interface provide additional information about the error.

Action: Ensure that the account, password, and database strings are set correctly in the *out_db_account* parameter in the *EvLoc* par file.

If the settings are correct, then talk to the database administrator.

Message: write_evloc_db_tables: gdi_get_counter failed trying to get <ID>s!! Error <ERR_CODE>: '<ERR_STRING>'

Description: An *orid* or *magid* value (<ID>) could not be retrieved from the **lastid** table. Error code <ERR_CODE> and message text <ERR_STRING> returned from the database interface provide additional information about the error.

Action: Add a record to the **lastid** table in the output database account for the <ID>.

Message: write_evloc_db_tables: Error <ERR_CODE>: '<ERR_STRING>'. DB error found while trying to delete <FEATURE> records from <TABLE> using query: <QUERY>

Description: Records could not be deleted from the input <FEATURE> database table <TABLE> using the query <QUERY>. <FEATURE> describes the data contained in <TABLE>, and is either "origin", "origerr", "assoc", "stamag", "netmag", or "event_control". Error code <ERR_CODE> and message text <ERR_STRING> returned from the database interface provide additional information about the error.

Action: Confirm that no uncommitted changes have been made to the database table <TABLE>. If the problem is not solved, ask the database administrator for help.

A work-around solution is to set the *write_to_input_db_tables* parameter to **FALSE** and write to separate output tables.

▼ Troubleshooting

Message: write_evloc_db_tables: Error <ERR_CODE>:
'<ERR_STRING>' while attempting to add <FEATURE>
records to <TABLE>

Description: Records could not be added to the output <FEATURE> database table <TABLE>. <FEATURE> describes the data contained in <TABLE>, and is either "origin", "origerr", "assoc", "ar_info", "arrival", "stamag", "netmag", or "event_control". Error code <ERR_CODE> and message text <ERR_STRING> returned from the database interface provide additional information about the error.

Action: If <TABLE> does not exist in the output database account, then create it. If <TABLE> does exist, confirm that no uncommitted changes have been made to it. If the problem is not solved, ask the database administrator for help.

libloc Configuration-related Error Messages

Message: Problems encountered reading travel-time info.
Exiting!!

Description: Some type of problem occurred while reading the travel-time files. This general message should be accompanied by a more specific message containing more details of the problem.

Action: Look for another error message preceding this message in the *EvLoc* output log, and refer to the corresponding action.

Message: Error: setup_tttables: Null phase list!

Description: The list of allowed phases in the *list_of_phases* parameter is empty.

Action: In the *EvLoc* par file, ensure that the *list_of_phases* parameter contains at least one phase.

Message: read_tt_tables: VMSF: <FILE> will not open!

Description: *libloc* is unable to open the VMSF for reading. The file is either unreadable or not there.

Action: Ensure that the *vmodel_spec_file* parameter in the *EvLoc* par file points to an existing, readable VMSF.

Message: read_tt_tables: No / in <DATA> directory name!

Description: The path to the travel-times <DATA> in the VMSF lacks a "/", and therefore is not valid. <DATA> is either "LP" or "radial_2d".

Action: Ensure the *vmodel_spec_file* parameter in the *EvLoc* par file points to a valid VMSF.

Message: read_tt_tables: Problems encountered while trying to read <DATA> tables!

Description: An error was encountered while attempting to read the LP or radial 2-D travel-time tables <DATA>. This message should be accompanied by one or more other more specific messages regarding the error. <DATA> is either "LP" or "radial_2d".

Action: In the *EvLoc* par file, ensure that the *vmodel_spec_file* parameter points to a valid VMSF. Confirm that the LP or radial 2-D path in the VMSF does exist in the configuration tree. Look for an accompanying message in the log file for more details, and find the corresponding description in this error listing.

▼ Troubleshooting

Message: Error: read_LP_info: File: <FILE> is missing!

Description: One of the required LP grid or velocity files <FILE> is not in the directory specified in the VMSF. The expected files, whose names are hard-coded in *libloc*, are: "LP_grid.LR", "LP_grid.LQ", "LP_vel.LR", and "LP_vel.LQ".

Action: In the *EvLoc* par file, ensure that the *vmodel_spec_file* parameter points to a valid VMSF.

Confirm that the LP path in the VMSF corresponds to a valid directory containing all the LP velocity data files.

Message: read_LP_info: Error reading <DATA> in file: <FILE>!

Description: An error occurred while reading LP data <DATA> from one of the LP grid or velocity data files <FILE>.

Action: In the *EvLoc* par file, ensure that the *vmodel_spec_file* parameter points to a valid VMSF. Confirm that the LP path in the VMSF corresponds to a valid directory containing all the LP velocity data files.

Message: read_tt_tables: Error reading <DATA> in file <FILE>

Description: <DATA> in this message can refer to station/phase/model-specific lines in the VMSF, or to a variety of values in the 1-D TT tables. The error is an indication of some type of format problem with file <FILE>.

Action: Verify the *vmodel_spec_file* parameter points to a valid VMSF.

If <FILE> is one of the 1-D TT files, verify that the 1-D velocity model(s) listed in the VMSF is correct.

If the velocity model name is correct, attempt to find the problem with <FILE> by comparing its contents with the format specification described in "One-dimensional Travel-time Table" on page 39.

Message: read_tt_tables: Velocity model not specified for:
<VMODEL> in station/phase/model listing

Description: One of the lines in the station/phase/model listing at the end of the VMSF is missing a valid velocity model <VMODEL> name.

Action: Verify the *vmodel_spec_file* parameter points to a valid VMSF. Velocity model names in the station/phase/model section of the VMSF must match one of the 1-D model names in the VMSF.

Message: read_tt_tables: No / in ellipticity correction
directory name!

Description: The path specified in the <*vmodel*>.elcor_dir file, located in the 1-D velocity model directory, is missing a "/", and therefore is not a valid absolute or relative path.

Action: If ellipticity corrections exist for <*vmodel*>, correct the path in the <*vmodel*>.elcor_dir file.

If no ellipticity corrections exist, remove or change the name of the <*vmodel*>.elcor_dir file to prevent *libloc* from reading it.

Message: read_tt_tables: No tables could be opened!

Description: No travel-time tables were read. This error message should be accompanied by other error messages indicating the specific read errors that occurred while attempting to read the travel-time tables.

Action: Verify the *vmodel_spec_file* parameter points to a valid VMSF.

Ensure that the 1-D velocity model path specified in the VMSF points to a valid set of travel-time tables.

▼ Troubleshooting

Message: Fatal error: <PHASE/STATION>: <VALUE> in SSSC file is inconsistent with filename!!

Description: The phase or station information in the SSSC file header does not match the value <VALUE> in the filename. <PHASE/STATION> is either "Phase" or "Station", and <VALUE> is either a station or phase name.

Action: If the particular SSSC file causing the error can be determined, edit the VMSF to comment or remove the corresponding station/phase/model entry. This prevents *libloc* from attempting to read the faulty SSSC file.

Otherwise, use a different velocity model by either switching to a different VMSF, or edit the VMSF 1-D velocity model specification line. The *EvLoc* parameter *vmodel_spec_file* controls the VMSF that is used.

Message: Fatal error: While trying to read current SSSC file: <FILE>

Description: A format or memory allocation problem occurred while reading the SSSC file <FILE>.

Action: Remove the station/phase/model entry from the VMSF corresponding to the values in <FILE>. Otherwise, use a different velocity model by either switching to a different VMSF, or by editing the VMSF 1-D velocity model specification line. The *EvLoc* parameter *vmodel_spec_file* controls the VMSF that is used.

Message: Warning: Bulk station correction in SSSC file is inconsistent with velocity model spec. file value for station: <STA> phase: <PHASE>
SSSC file has been updated for you and put in temporary (and volatile) directory: <DIR>

Description: This warning indicates that the bulk station correction value for the specified station <STA> and phase <PHASE> in the VMSF and SSSC files are different. An updated version of the SSSC file in question is written, using the value from the VMSF, to the specified directory <DIR>.

Action: No action is necessary, but ideally the newly written SSSC file with the updated bulk correction or sedimentary velocity should be copied from <DIR> to the SSSC directory.

Message: Warning: Sed. velocity in SSSC file is inconsistent with velocity model spec. file value for station: <STA> phase: <PHASE>
SSSC file has been updated for you and put in temporary (and volatile) directory: <DIR>

Description: This warning indicates that the sedimentary velocity value for the specified station <STA> and phase <PHASE> in the VMSF and SSSC files are different. An updated version of the SSSC file in question is written, using the value from the VMSF, to the specified directory <DIR>.

Action: No action is necessary, but ideally the newly written SSSC file with the updated bulk correction or sedimentary velocity should be copied from <DIR> to the SSSC directory.

▼ Troubleshooting

Message: Error: problem with number of <ATTRIBUTE> samples in SSSC file for sta <STA>, phase <PHASE>

Description: In the SSSC file for the specified station <STA> and phase <PHASE>, the actual number of grid samples read in for attribute <ATTRIBUTE> does not match the number specified at the top of the file. <ATTRIBUTE> is either "lat", "lon", or "depth".

Action: Edit the SSSC file in question to be self-consistent. If that is not practical, the VMSF can be edited to remove the station/phase line in question. This prevents *EvLoc* from reading the bad SSSC file.

Message: Error: problem with number of <DATA> values in SSSC file for sta <STA>, phase <PHASE>

Description: In the SSSC file corresponding to the specified station <STA> and phase <PHASE>, the number of the data <DATA> values does not match the number of grid node points. <DATA> is either "correction" or "modeling error".

Action: Edit the SSSC file in question to be self-consistent. If that is not practical, the VMSF can be edited to comment out or remove the station/phase line in question. This prevents *EvLoc* from reading the bad SSSC file.

Message: Warning: SSSC file zero boundary value violation.
For more details run *EvLoc* in mode 3

Description: One or more of the SSSC files violates the zero boundary value condition for correction values. This condition is not strictly enforced, which is why this is a warning and not an error. The potential problem that can arise with non-zero corrections on the region boundary is an oscillating, non-convergent location solution.

Action: No action is necessary, but the SSSC files in question should be fixed. As the message indicates, *EvLoc* can be run in mode "3", which lists each SSSC file that violates the zero correction boundary condition. Unlike the other SSSC errors, there is no need to prevent *EvLoc* from using these SSSC files unless a particular location computation fails to converge. However, many potential causes exist for a location computation that does not converge.

Message: ERROR: set_sta_pt: site table not specified or empty!

Function, set_sta_pt(), cannot be called until site table is available!

TT Tables: Cannot set Sta_Pt structure! site table likely missing!

Description: No data were read from the **site** database table specified by the *EvLoc* parameter *site_table*. Either the **site** table does not exist, or the query used by *EvLoc* did not return any rows. The dates of the event(s) specified by the *origin_query* parameter in combination with the network name specified by the *network* parameter, and the **affiliation** table are used to construct the **site** table query.

Action: Ensure that the *site_table* and *affiliation_table* parameters point to valid database tables. Also, ensure that the *network* parameter is set to a valid name (it must be represented in the **affiliation** table).

▼ Troubleshooting

-
- Message:** Problems encountered while trying to read SASCTables. Exiting!!
- Description:** This general message should be accompanied by a more specific message containing more details of the problem.
- Action:** Look for another error message preceding this message in the *EvLoc* output log, and refer to the corresponding entry in this section.
-
- Message:** Message: No SASC tables can be read since no / in SASC directory name!
- Description:** The *EvLoc* parameter *sasc_dir_prefix* does not contain a "/", and therefore is not a valid path. SASC tables contain slowness and azimuth corrections. They are not required, which is why this is a message and not an error. If the data in question contain defining slowness and or azimuthal measurements, then SASC's should be applied.
- Action:** Ensure that the *sasc_dir_prefix* parameter contains the full absolute correct path for the SASC tables.
-
- Message:** Message: No SASC tables can be read since no directory/prefix is specified!
- Description:** The *EvLoc* parameter *sasc_dir_prefix* is not specified. As a consequence, the SASC tables cannot be read. SASC tables contain slowness and azimuth corrections. They are not required, which is why this is a message and not an error. If the data in question contain defining slowness and or azimuthal measurements, then SASC's should be applied.
- Action:** Ensure that the *sasc_dir_prefix* parameter contains the full absolute correct path for the SASC tables.
-

Message: read_sasc: No . in SASC file name!

Description: SASC file names are of the form *sasc.<station>*. However, one or more files in the specified SASC directory are missing a ".", and therefore do not have valid SASC file names. Either legitimate SASC files are incorrectly named or the *sasc_dir_prefix* parameter does not point to a valid SASC directory.

Action: Ensure that the *EvLoc* parameter *sasc_dir_prefix* corresponds to a valid directory containing SASC files. SASC's are optional corrections in *EvLoc* location processing; if no valid SASC's are available, the *sasc_dir_prefix* parameter can be left unspecified.

Message: read_sasc: Number of az/slow corr bins found is > specified in file: <FILE> This will overrun memory, so bailing out!!!!

Description: The format of the SASC file <FILE> is not internally consistent.

Action: Ensure that the *EvLoc* parameter *sasc_dir_prefix* corresponds to a directory containing valid SASC files. SASC's are optional corrections in *EvLoc* location processing; if no valid SASC's are available, the *sasc_dir_prefix* parameter can be left unspecified.

Message: read_sasc: Number of az/slow corr bins found: <B1> disagrees with the <B2> specified in file: <FILE> Will continue, but this should be fixed!!!!

Description: The format of the SASC file <FILE> is not internally consistent.

Action: Ensure that the *EvLoc* parameter *sasc_dir_prefix* corresponds to a directory containing valid SASC files. SASC's are optional corrections in *EvLoc* location processing; if no valid SASC's are available, the *sasc_dir_prefix* parameter can be left unspecified.

▼ Troubleshooting

Message: `get_LP_velocity: Input period of <PERIOD> sec. is out-of-range!`

Description: While looking up values for the LP phase velocity interpolation, the input period `<PERIOD>` was outside the range of the values in the table.

Action: Confirm that the LP path in the VMSF corresponds to a directory containing the valid LP velocity data files.

Message: `ERROR: Travel time requested for station not in 2-D acoustic tables.`

Description: TT data do not exist in the specified radial 2-D TT directory for one of the stations in the **arrival** data set.

Action: Ensure that the path to the radial 2-D TT tables specified in the VMSF points to the correct directory. The location computation completes even when this message appears; the acoustic arrivals at one station are not used in the location.

Message: `Error 26: Requested test-site region not available!`

Description: The test-site region name specified by the `ts_region` parameter does not correspond to a region name in the `<vmodel>.ts_cor` file.

Action: Check the test-site region name specified by the `EvLoc` parameter `ts_region`. Ensure that it corresponds to one of the regions in the test-site file associated with the velocity model specified in the VMSF. Test-site files reside in the same directory as the 1-D TT tables for a given model, and are named `<vmodel>.ts_cor`.

Message: EvID: <EVID> In_OrID: <ORID>
Locator: Warning - No observations to process!

Description: No arrivals were found for the event specified by *evid* <EVID> and *orid* <ORID>.

Action: Several parameters can affect the retrieval of arrivals from the database.
First, make sure the *list_of_phases* parameter contains a reasonable list of phases.
Second, ensure that the *network* parameter specifies a network that includes stations that have recorded the event in question.
Finally, ensure that the *arrival_table* and *assoc_table* parameters specify tables that correspond to the **origin** table.

Message: EvID: <EVID> In_OrID: <ORID>
Locator: Bad input assoc data!

Description: **assoc** data are missing for the event specified by *evid* <EVID> and *orid* <ORID>.

Action: Ensure that the input database table name parameters are correct and consistent with each other. Ensure that the input **assoc** table specified in the *assoc_table* parameter contains the expected data.

Message: Locator: Bad input origin data!

Description: A problem exists with **origin** data for a particular event. This error indicates *EvLoc* has internally mishandled valid **origin** data, and there is no straightforward solution.

Action: Ensure that the input database tables are all self-consistent. This error is not due to a problem with the *origin_query* parameter.

▼ Troubleshooting

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Bad input origerr data!

Description: **origerr** data are missing for the event specified by *evid* <EVID> and *orid* <ORID>.

Action: Ensure that the input database table name parameters are correct and consistent with each other (for example, the specified **origerr** table should contain the same list of events as the **origin** table). Ensure that the input **origerr** table specified in the *origerr_table* parameter contains the expected data.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Lat/Lon value(s) out of range given a fixed lat/lon!

Description: The *fix_latlon* parameter is set true, but for some reason one or both of the initial latitude or longitude values read from the **origin** table in the database for the event specified by *evid* <EVID> and *orid* <ORID> are not valid. In this case the latitude and longitude cannot be fixed.

Action: Set the *EvLoc* parameter *fix_latlon* to false. Once a valid location has been computed, subsequent runs may be performed with latitude and longitude fixed.

libmagnitude Configuration-related Error Messages

Message: Problems encountered while trying to read magnitude info. Exiting!!

Description: This general message should be accompanied by a more specific message containing more details of the problem.

Action: Look for another error message preceding this message in the *EvLoc* output log, and refer to the corresponding action.

Message: read_mdf: MDF: <MDF> will not open!
MDreadErr1: Cannot open MDF!

Description: The Magnitude Description File <MDF> could not be opened because it either does not exist or is unreadable.

Action: Reset the *mag_descrip_file* parameter in the input *EvLoc* par file to point to an existing MDF. If the MDF exists, then change the permissions so that it may be read.

Message: MDF is empty, so no magnitude info can be computed
MDreadErr2: MDF incorrectly formatted!

Description: No magnitude specification data could be read from the MDF because the file is either empty or incorrectly formatted.

Action: Add magnitude specification data to the MDF. Specifically, ensure that the first non-commented line in the MDF is the first line of magnitude specification data.

▼ Troubleshooting

-
- Message:** Incorrect number of arguments per magtype in MDF:
expecting <N1>, found <N2>
MDreadErr2: MDF incorrectly formatted!
- Description:** The number of parameters <N2> read from a line of magnitude specification data in the MDF does not equal the number of parameters expected <N1> due to an internal inconsistency in the MDF.
- Action:** Ensure that all lines of magnitude specification data contain <N1> parameter arguments (ignoring any trailing comments).
-
- Message:** Incorrect LB, UB, or BL value for magtype <MAGTYPE>
in MDF:
Must satisfy $LB \leq BL \leq UB$
MDreadErr2: MDF incorrectly formatted!
- Description:** The lower bound (LB), upper bound (UB), and baseline (BL) standard deviations do not satisfy the relationship $LB \leq BL \leq UB$ for *magtype* <MAGTYPE> in the magnitude specification data section of the MDF.
- Action:** Reset the LB, UB, and BL values so that the above relationship is true for <MAGTYPE>.
-
- Message:** Incorrect number of arguments per sta/TLtype in MDF
file: expecting <N1>, found <N2>
MDreadErr2: MDF incorrectly formatted!
- Description:** The number of parameters <N2> read from a line of bulk magnitude station corrections in the MDF does not equal the number of parameters expected <N1> due to an internal inconsistency in the file.
- Action:** Ensure that all lines of bulk magnitude station correction data contain <N1> parameter arguments (ignoring any trailing comments).
-

Message: Station DFAULT not found in MDF file station correction list for Tltype <TLTYPE>.

MDreadErr2: MDF incorrectly formatted!

Description: This error is due to an internal inconsistency in the MDF. The *wgt_ave* parameter in the MDF is set to "1" but the required station "DFAULT" is not present in the bulk magnitude station correction data section of the MDF for *Tltype* <TLTYPE>.

Action: Add the following line of data to the bulk magnitude station correction data section:

DFAULT <TLTYPE> 0.0 0.0

Message: read_tlsf: site table not specified or empty!

Function, read_tlsf(), cannot be called until site table is available!

SSgetErr1: No input site table info available for Sta_Pt!

Description: No data were read from the **site** database table specified by the *EvLoc* parameter *site_table*. Either the **site** table does not exist or the query used by *EvLoc* did not return any rows. The dates of the event(s) specified by the *origin_query* parameter in combination with the network name specified by the *network* parameter and the **affiliation** table are used to construct the **site** table query.

Action: Ensure that the *site_table* and *affiliation_table* parameters point to valid database tables. Also, ensure that the *network* parameter is set to a valid name (it must be represented in the **affiliation** table).

▼ Troubleshooting

Message: read_tlsf: TLSF: <TLSF> will not open!
TLreadErr1: Cannot open TLSF!

Description: The Transmission Loss Specification File <TLSF> could not be opened. Either the file does not exist or it is not readable.

Action: Reset the *tl_spec_file* parameter in the input *EvLoc* par file to point to an existing TLSF. If the TLSF exists, then change the permissions so that it may be read.

Message: read_tlsf: No / in TL model filename!
TLreadErr2: TLSF incorrectly formatted!

Description: To be valid, the TLSF system filename must have a "/" in its path.

Action: Ensure that the TLSF filename specified in the input *EvLoc* parameter *tl_spec_file* includes a "/" and points to a valid TLSF.

Message: read_tlsf: TL model not specified for: <TLM> for
TLtype:<TLTYPE>
TLreadErr2: TLSF incorrectly formatted!

Description: The TLM pathway section of the TLSF specifies a <TLM> that does not have a corresponding line in the default TLM description data section for *TLtype* <TLTYPE>.

Action: Add default TLM description data for the TLM or remove the TLM from the TLM pathway section of the TLSF.

Message: read_tlsf: Error reading station/TLtype/model info
in file:

TLreadErr2: TLSF incorrectly formatted!

Description: Fewer than the required five values were found in a line of station-specific TLM description data in the TLSF.

Action: Ensure that all lines of station-specific TLM description data contain five arguments (ignoring any trailing comments).

Message: read_tlsf: Warning! STM: <STA>/<TLTYPE>/<ROOT> line
found to be redundant with info specified in
ttype_model_descrip!

STM line: <LINE>

will be ignored!

Description: The *TLtype* <TLTYPE>, root name <ROOT>, and phase defined in <LINE> are the same as those in a previously read default TLM. The line is ignored and the station-specific TLM filename constructed from <ROOT> and <TLTYPE> is not read.

Action: Delete <LINE> from the TLSF if the station-specific TLM is not needed. Otherwise, change the *TLtype*, root name, phase, or channel specified in <LINE> to point to a different station-specific TLM.

▼ Troubleshooting

Message: read_tlsf: Warning! STM: <STA>/<TLTYPE>/<ROOT> line found to be a duplicate with another STM record!
STM line: <LINE>
will be ignored!

Description: The station-specific TLM description data in line <LINE> of the TLSF is equivalent to that in another station-specific TLM. The line is ignored and the station-specific TLM filename constructed from the root name <ROOT> and *Tltype* <TLTYPE> is not read a second time.

Action: Delete <LINE> from the TLSF.

Message: read_tlsf: Error! STM: <STA>/<TLTYPE>/<ROOT> line not associated with any *tltype_model_descrip* *Tltype* definition!
STM line: <LINE>
will be ignored!

Description: The station-specific TLM description data in line <LINE> of the TLSF specifies a *Tltype* <TLTYPE> that was not read from the default TLM description data section. The line is ignored and the station-specific TLM filename constructed from the root name <ROOT> and <TLTYPE> is not read.

Action: Delete <LINE> from the TLSF if the station-specific TLM does not exist or is not needed. Otherwise, change the *Tltype* specified in <LINE> to one of the *Tltypes* listed in the default TLM description data.

Message: Warning: File <TLM> will not open!
TLreadWarn1: A requested TL file was not found!

Description: The TLM <TLM> could not be opened; it either does not exist or is not readable.

Action: Ensure that the TLM path and filename point to a valid TLM. Refer to "Transmission Loss Specification File" on page 69 to see how the TLM path is constructed.

Message: read_tlsf: No TL tables could be opened!
TLreadErr3: No TL tables could be found!

Description: None of the TLMs specified in the TLSF could be opened.

Action: Ensure that the TLM path and filename point to a valid TLM. Refer to "Transmission Loss Specification File" on page 69 to see how the TLM path is constructed.

Message: read_tl_table: Error reading number of depth samples in file: <TLM>
TLreadErr4: TL table incorrectly formatted!

Description: The number of sample depths specified in the transmission loss data section of the TLM <TLM> could not be read or is missing.

Action: Format the number of sample depths as indicated in Table 29 on page 78.

▼ Troubleshooting

Message: read_tl_table: Error reading depth sample value in file: <TLM>

TLreadErr4: TL table incorrectly formatted!

Description: At least one of the sample depths specified in the transmission loss data section of the TLM <TLM> could not be read or is missing.

Action: Format the sample depths as indicated in Table 29 on page 78.

Message: read_tl_table: Error reading number of distance samples in file: <TLM>

TLreadErr4: TL table incorrectly formatted!

Description: The number of sample distances specified in the transmission loss data section of the TLM <TLM> could not be read or is missing.

Action: Format the number of sample distances as indicated in Table 29 on page 78.

Message: read_tl_table: Error reading distance sample value in file: <TLM>

TLreadErr4: TL table incorrectly formatted!

Description: At least one of the sample distances specified in the transmission loss data section of the TLM <TLM> could not be read or is missing.

Action: Format the sample distances as indicated in Table 29 on page 78.

Message: read_tl_table: Error reading transmission-loss value in file: <TLM>

TLreadErr4: TL table incorrectly formatted!

Description: At least one of the transmission loss values specified in the transmission loss data section of the TLM <TLM> could not be read or is missing.

Action: Format the transmission loss value as indicated in Table 29 on page 78.

Message: read_tl_table: Error reading premature EOF in file: <TLM>

TLreadErr5: TL modelling error table incorrectly formatted!

Description: The required comment line separating the number of distance/depth samples and the transmission loss modeling error(s) in the transmission loss modeling error data section of the TLM <TLM> could not be read or is missing. This error only arises when parsing a TLM that specifies either a single-valued or distance-dependent transmission loss modeling error(s).

Action: Format the transmission loss modeling error section as indicated in Table 30 on page 81 or Table 31 on page 82. In particular, ensure that one and only one comment line separates the number of distance/depth samples and the transmission loss modeling error(s).

▼ Troubleshooting

Message: read_tl_table: Error reading single bulk modelling error in file: <TLM>
TLreadErr5: TL modelling error table incorrectly formatted!

Description: The single-value transmission loss modeling error specified in the transmission loss modeling error data section of the TLM <TLM> could not be read or is missing. This error only arises when parsing a TLM that specifies a single-value transmission loss modeling error.

Action: Format the single-value transmission loss modeling error as indicated in Table 30 on page 81.

Message: read_tl_table: Error reading modelling error distance sample value in file: <TLM>
TLreadErr5: TL modelling error table incorrectly formatted!

Description: At least one of the sample distances specified in the transmission loss modeling error data section of the TLM <TLM> could not be read or is missing. This error only arises when parsing a TLM that specifies either distance-dependent or distance/depth-dependent transmission loss modeling errors.

Action: Format the sample distances as indicated in Table 31 on page 82 or Table 32 on page 83.

Message: read_tl_table: Error reading distance-dependent modelling error in file: <TLM>
TLreadErr5: TL modelling error table incorrectly formatted!

Description: At least one of the distance-dependent transmission loss modeling errors specified in the TLM <TLM> could not be read or is missing. This error only arises when parsing a TLM that specifies distance-dependent transmission loss modeling errors.

Action: Format the distance-dependent transmission loss modeling errors as indicated in Table 31 on page 82.

Message: read_tl_table: Error reading modelling error depth sample value in file: <TLM>
TLreadErr5: TL modelling error table incorrectly formatted!

Description: At least one of the sample depths specified in the transmission loss modeling error data section of the TLM <TLM> could not be read or is missing. This error only arises when parsing a TLM that specifies distance/depth-dependent transmission loss modeling errors.

Action: Format the sample depths as indicated in Table 32 on page 83.

▼ Troubleshooting

Message: read_tl_table: Error reading distance/depth modeling error in file: <TLM>
TLreadErr5: TL modelling error table incorrectly formatted!

Description: At least one of the distance/depth-dependent transmission loss modeling errors specified in the TLM <TLM> could not be read or is missing. This error only arises when parsing a TLM that specifies distance/depth-dependent transmission loss modeling errors.

Action: Format the distance/depth-dependent transmission loss modeling errors as indicated in Table 32 on page 83.

Message: Error encountered while attempting to read mag. test-site corr. file:
<TSF>
TLreadErr6: TL test-site corr. file incorrectly formatted!

Description: Fewer than the required four values were found in a line of test-site region identification data in the magnitude test-site correction file <TSF>.

Action: Ensure that all lines of test-site region identification data contain four parameter arguments (ignoring any trailing comments).

Message: Requested mag. test-site region: <REG>, not available!

Description: The test-site region name specified by the *mag_ts_region* parameter does not correspond to a region name in the magnitude test-site correction file.

Action: Check the test-site region name specified by the *EvLoc* parameter *mag_ts_region*; it must correspond to one of the regions in the test-site file.

Message: Magtype: <MAGTYPE> is not specified within MDF
Hence, this magnitude cannot be computed!

Description: The *list_of_magtypes* parameter used to tell *EvLoc* the *magtype* to compute contains *magtype* values that are not defined in the MDF.

Action: In the *EvLoc* parameter *list_of_magtypes*, change <MAGTYPE> to one that does have magnitude description parameters specified in the MDF.

Alternatively, add magnitude description parameters to the MDF; add TLM pathway, default TLM description, and optional station-specific TLM description parameters to the TLSF; and create TLMs for <MAGTYPE>. See "Constructing Magnitude Earth-model Files" on page 64 for details of the format of the MDF and TLSF.

▼ Troubleshooting

Message: Warning: Invalid magtype, <MAGTYPE>, specified in attempt to reset <PARAMETER> settings. Values specified in MDF will be retained!

Description: The parameter <PARAMETER> in the magnitude description data section of the MDF could not be dynamically changed for *magtype* <MAGTYPE> because <MAGTYPE> is not defined in the MDF. <PARAMETER> is either "amptype", "algorithm", "minimum distance", "maximum distance", "lower/upper bound limit", "baseline stdev", or "weighted average flag".

Action: In the *EvLoc* parameters *list_of_magtypes* and *list_of_mb_magtypes*, change <MAGTYPE> to one that has magnitude description parameters specified in the MDF.

Message: Warning: Invalid magtype, <MAGTYPE>, specified in attempt to revert <PARAMETER> settings. Override values will be retained!

Description: The parameter <PARAMETER> could not be dynamically restored to its original value (specified in the magnitude description section of the MDF) for *magtype* <MAGTYPE>. <PARAMETER> is either "amptype", "algorithm", "minimum distance", "maximum distance", "lower/upper bound limit", "baseline stdev", or "weighted average flag".

Action: In the *EvLoc* parameters *list_of_magtypes* and *list_of_mb_magtypes*, change <MAGTYPE> to one that has magnitude description parameters specified in the MDF.

Message: <MAGTYPE> weighted average desired, but no <STA> station weights available!

Description: This error is caused by an internal inconsistency either within the MDF or between the MDF and a TLM. The *wgt_ave* parameter for <MAGTYPE> is set to "1" in the MDF but the required modeling errors do not exist in the MDF or a TLM for station <STA> and *magtype* <MAGTYPE>. Modeling errors must exist to estimate a weighted-average network magnitude.

Action: Add transmission loss modeling error data to the TLM or add a line of bulk magnitude station correction parameters to the MDF for the station and *TLtype*. If weighted-average network magnitudes are not to be estimated, then set *wgt_ave*=0.

libloc Processing-related Error Messages

Message: Warning: Sum of individual LP segments differ by > 0.1% from original distance!

Description: The sum of the individual segments of a LP raypath as calculated in *libLP* differs too much from the input distance as computed by the *distaz()* function.

Action: No action is necessary, but it may be useful to examine the uncertainties and/or data importances in the log file of any LP data used in the location. Any suspect LP arrivals may be set non-defining for a subsequent *EvLoc* run to re-locate the event(s). Action by the software maintainer may be required to determine why the ray tracer is failing.

▼ Troubleshooting

Message: ERROR! Acoustic travel time requested without epoch time set. Will use <SEASON> tables as default!

Description: Because acoustic 2-D travel times vary with the time of year, the epoch time must be known to select the appropriate set of TT data. This warning error indicates that the epoch time is not set, so tables for the season <SEASON> are used.

Action: No action is necessary for this extremely unlikely, non-fatal error.

Message: trv_time_w_ellip_elev: phase_index is too large!!!

Description: An internal inconsistency with *libloc*'s list of phases was found. The location computation completes if enough other arrivals remain. The arrival for which the error occurred is set non-defining and not used in the location.

Action: No straightforward solution exists for this problem. Contact the maintainer of the software.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Mismatch between input arrival/assoc data!

Description: The internal *EvLoc* arrays containing **arrival** and **assoc** data have gotten out of order during processing of the event specified by *evid* <EVID> and *orid* <ORID>. This error indicates an internal mishandling of data by *EvLoc*.

Action: No straightforward solution exists for this problem. Ensure that the data in the input **arrival** and **assoc** tables are self-consistent. If this has no effect, then contact the maintainer of the software.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Insufficient defining data before location is even attempted!

Description: Either the number of defining data is less than the number of free solution parameters (one or more of latitude, longitude, depth, and time, depending on the parameter settings), or the origin time is not fixed and there are no time-defining data for the event specified by *evid* <EVID> and *orid* <ORID>.

Action: Ensure that the *list_of_phases* and *network* parameters are appropriate, so that all available data are being used.

Alternatively, fix one or more of the location solution parameters via the *fix_ot*, *fix_latlon*, and *fix_depth* *EvLoc* parameters.

Message: best_guess: Too few data to get an initial location
Try Tonga-Fiji !!

Description: A warning that in the absence of an existing initial location read from the database, the *best_guess()* function was called but failed to establish an initial location. Instead, the seed location for the location calculation is set to the coordinates of Tonga-Fiji (20E, 175S).

Action: No action is necessary, although a different starting location could be set by manually editing the latitude and longitude values in the input **origin** database table.

▼ Troubleshooting

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Insufficient defining data remains while attempting to locate event!

Description: The number of data points is less than the number of solution parameters for the event specified by *evid* <EVID> and *orid* <ORID>.

Action: Ensure that the *list_of_phases* and *network* parameters are appropriate so that all available data are being used. Alternatively, fix one or more of the location solution parameters via the *fix_ot*, *fix_latlon*, and *fix_depth* EvLoc parameters.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Insufficient defining data remains due to large residual restriction!

Description: The EvLoc parameter *ignore_big_res* is set true, and some data were discarded because their residuals were too large (larger than *big_res_mult* times the data standard error). This data restriction has caused the number of defining data to fall below the number of solution parameters for the event specified by *evid* <EVID> and *orid* <ORID>. The most likely cause of this error is a bad association set.

Action: Try re-associating and locating again. If the associations are good, there are several other possible solutions. Set *ignore_big_res* false. Increase the value of *big_res_mult*. Fix one or more of the location solution parameters via the *fix_ot*, *fix_latlon*, and *fix_depth* parameters. Ensure that the *list_of_phases* and *network* parameters are appropriate so that all available data are being used.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Insufficient defining data remains due to TT correction restriction!

Description: The parameter *only_sta_w_corr* is set true, meaning that only arrivals receiving source-dependent corrections (for example, SSSC, test-site) are used in the location. This data restriction caused the number of defining data to fall below the number of solution parameters for the event specified by *evid* <EVID> and *orid* <ORID>.

Action: Remove the data restriction by setting *only_sta_w_corr* false.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Insufficient defining data remains due to hole in T-T table(s)!

Description: One or more data points for the event specified by *evid* <EVID> and *orid* <ORID> was set non-defining and not used because the TT table for that phase contains no data at the distance in question. This data restriction has caused the number of defining data to fall below the number of solution parameters.

Action: First, try a different initial location by editing the **origin** record. Second, ensure that the *list_of_phases* and *network* parameters are appropriate so that all available data are being used. As a last resort, fix one or more of the location solution parameters via the *fix_ot*, *fix_latlon*, and *fix_depth* EvLoc parameters.

▼ Troubleshooting

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Insufficient defining data remains due to
T-T table extrapolation!

Description: The distance and/or depth point being interpolated in the TT table data for one of the arrivals of the event specified by *evid* <EVID> and *orid* <ORID> was out of range, causing the arrival to be set non-defining. This data restriction has caused the number of defining data to fall below the number of solution parameters. The likely cause of this error is a badly identified phase.

Action: Rename any phase that would not normally appear at that distance.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: SVD routine cannot decompose given matrix!

Description: The single value decomposition computation failed for the event specified by *evid* <EVID> and *orid* <ORID>. The association set may be bad.

Action: There may not be a simple solution to this error. If possible, include more defining data by adding to the *list_of_phases* parameter. Re-associate and try again.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Divergent solution encountered!

Description: The iterative location computation diverged and no solution was found for the event specified by *evid* <EVID> and *orid* <ORID>. The association set may be bad.

Action: There may not be a simple solution to this error. If possible, include more defining data by adding to the *list_of_phases* parameter. Re-associate and try again.

Message: EvID: <EVID> In_OrID: <ORID>

Locator: Maximum number of iterations exceeded!

Description: The iterative location computation exceeded the allowed number of iterations without converging on a solution for the event corresponding to *evid* <EVID> and *orid* <ORID>. The likely causes are too few data or a bad association set.

Action: Try re-associating and locating again. If possible, include more defining data by adding to the *list_of_phases* parameter. Alternatively, increase the parameter *max_iter*, which controls how many iterations are allowed before the computation is abandoned. Some solutions converge slowly and the solution determined at the last iteration may be acceptable.

libmagnitude Processing-related Error Messages

Message: `interp_for_tl_value: tl_index is too large!!!`

Description: An internal inconsistency with *libmagnitude*'s list of TLMs was found. The magnitude computation completes if enough other arrivals remain. The station magnitude for which the error occurred is non-defining and not used in any network magnitude estimates.

Action: No straightforward solution exists for this problem. Contact the maintainer of the software.

▼ Troubleshooting

Message: Warning: Network stdev = <STDEV> < lower bound in
mdf file = <LB> ->
Setting network sigma = <LB>

Description: The standard deviation <STDEV> of the network-average magnitude is set to maximum of the lower bound <LB> and <STDEV>.

Action: No action is necessary, although the lower bound in the magnitude description data may be decreased if desired.

Message: Warning: Network stdev = <STDEV> > upper bound in
mdf file = <UB> ->
Setting network sigma = <UB>

Description: The standard deviation <STDEV> of the network-average magnitude is set to the minimum of the upper bound <UB> and <STDEV>.

Action: No action is necessary, although the upper bound in the magnitude description data may be increased if desired.

Message: mag_boot_strap: Cannot generate epoch time for ran-
dom number generation!

Description: A call to a *libstdtime* function returned an error code during compu-
tation of bootstrap-resampled MLE magnitudes. The bootstrap-resa-
mpled MLE magnitudes written to the database and log file are
useless.

Action: Contact the maintainer of the software.

Message: EM ESTIMATOR HAS NOT CONVERGED AFTER <N> ITERATIONS!

Description: The expectation-maximization (EM) algorithm did not produce a stable result for the MLE magnitude given the input set of amplitude data.

Action: In general, the results may be accepted.

If a solution is desired, change the input set of amplitude records (for example, add records) to increase or decrease the number of defining station magnitudes.

Otherwise, contact the maintainer of the software.

SOLVING COMMON PROBLEMS

This section provides instructions for solving common problems that occur in the *EvLoc* software.

No Output

When a particular *EvLoc* run produces no logged output and no error messages, it means that the origin query passed to *EvLoc* returned no events to process. Provided that *EvLoc* returns with an exit code of 0, then this situation is not an error. In the course of normal operational processing some time periods may contain no events, so it should not be an error when there are no data to process.

To change the origin query, edit the *EvLoc* parameter *origin_query*. Origin queries should take the form of a "where" SQL clause that uses one (or more) criterion such as *orid* or a time period to specify the rows of the **origin** table to read into *EvLoc* for processing. An origin query can be manually tested prior to running *EvLoc* to ensure that it returns one or more events.

▼ Troubleshooting

Error Recovery

In the event of an *EvLoc* failure during processing, examine the log file for information related to the cause of the failure. “Interpreting Error Messages” on page 93 may be helpful for diagnosing the problem. Solve the problem and restart *EvLoc* as described in “Software Startup” on page 20. The output tables from the failed *EvLoc* interval need not be cleaned up because output data were not written. *EvLoc* only writes output data at the very last stage of processing, after all error messages have been exhausted.

REPORTING PROBLEMS

The following procedures are recommended for reporting problems with the application software:

1. Diagnose, log, and document the problem as best as you are able.
2. Record information regarding symptoms and conditions at the time of the software failure.
3. Retain copies of relevant sections of application log files.
4. Contact the provider or maintainer of the software for problem resolution if local changes of the environment or configuration are not sufficient.
5. Contact the system administrator if the problem is system-related.

Chapter 4: Installation Procedures

This chapter provides instructions for installing the software and includes the following topics:

- Preparation
- Executable Files
- Input Files
- Database
- Tuxedo Files
- Initiating Operations
- Validating Installation

Chapter 4: Installation Procedures

PREPARATION

This section describes how to obtain the Event Location and Magnitude software and the hardware requirements needed to run the software.

Obtaining Released Software

The software is obtained via FTP from a remote site or via a physical medium, such as tape or CD-ROM. The software and associated system files are stored as one or more tar files. The software and data files are first transferred via FTP or copied from the physical medium to an appropriate location on a local hard disk. The tar files are then untarred into a standard UNIX directory structure.

Hardware Mapping

The user must select the hardware on which to run the software components. Software components are generally mapped to hardware to be consistent with the software configuration model. See “Environment and States of Operation” on page 16 for additional details regarding the hardware environment necessary to run the Event Location and Magnitude software.

EXECUTABLE FILES

Install the shared libraries *libloc* and *libmagnitude* in a location such that applications may link to them during compilation. Install the executable file *EvLoc* in a location accessible to the operational system. Ensure that the *\$PATH* environment variable contains the directory location of the *EvLoc* executable.

INPUT FILES

Several input files must be installed in the operational configuration tree prior to execution of *EvLoc* or any application that links to the *libloc* or *libmagnitude* software libraries. These files include par files necessary for running *EvLoc* and configuration and earth-model files read by *libloc* and *libmagnitude* functions.

EvLoc Parameter Files

A par file containing required and optional parameter control values is needed to run *EvLoc*. The par file can reside anywhere as it is specified on the command line at run time. Unlike the system configuration and earth-model files, par files are not necessarily part of the installed system, in part because they may require regular editing or updating. Often, however, *EvLoc* par files reference other system-level parameter files, which may be part of the installed system. One or more sample par files may exist in the */doc* area of the *EvLoc* source code directory. These files can be copied and altered for use with *EvLoc*. See “Control Parameter Setup” on page 22 for a detailed explanation of the creation and use of *EvLoc* parameter files.

Location Earth-model Files

The earth-model files required by *libloc* for location and TT processing are listed in Table 5 on page 14 and described in “Constructing Location Earth-model Files” on page 34. These files can be customized when integrating new travel-time information into the operational system.

To be utilized by *libloc*, the location earth-model files must all be installed in the configuration tree. With the exception of the SASC tables, all location earth-model files are either specified directly in the VMSF or indirectly via the path to the 1-D velocity-model directory (for example, the EC, SSSC, and LTSC files).

The input par files or scheme arguments of applications that compute locations must be configured to point to the VMSF and SASC table paths. See “Software Overview” on page 2 for a list of the existing applications, and their software user

▼ Installation Procedures

manuals or man pages to see how to specify the paths to the VMSF and SASC tables. The *EvLoc* parameters controlling these paths are *vmodel_spec_file* and *sasc_dir_prefix*, respectively.

Magnitude Earth-model Files

The earth-model files required by *libmagnitude* for magnitude processing are listed in Table 6 on page 15 and described in “Constructing Magnitude Earth-model Files” on page 64. These files can be customized when integrating new magnitude information into the operational system.

To be utilized by *libmagnitude*, the magnitude earth-model files must be installed in the configuration tree. With the exception of the MDF, all magnitude earth-model files are either specified directly in the TLSF or indirectly via the path to the TLM directory (for example, the MTSC file).

The input par files or scheme arguments of applications that estimate magnitudes must be configured to point to the MDF and TLSF filesystem paths. See “Software Overview” on page 2 for a list of the existing applications, and their software user manuals or man pages to see how to specify the MDF and TLSF filesystem paths. The *EvLoc* parameters controlling these paths are *mag_descrip_file* and *tl_spec_file*, respectively.

DATABASE

This section describes database elements required for operation of *EvLoc*, including accounts, tables, and initialization of the **lastid** table. The *libloc* and *libmagnitude* shared libraries do not interface with a database. Refer to the software user manuals for other applications that link to *libloc* and *libmagnitude* for information about their required database accounts, tables, and **lastid** table initialization.

Accounts

In general, *EvLoc* exchanges data with any account that has sufficient read and write privileges. In IDC operations, *EvLoc* exchanges data with the LEB account when run as part of Post-analysis Processing and when called by *rebrevise*.

Tables

See “Database Tables” on page 12 for inventories of database tables required by *EvLoc* in location mode and magnitude mode, respectively. *EvLoc* does not require special database tables.

Initialization of *lastid*

The *lastid* table contains the names of identifiers (*keynames*) and identifier values (*keyvalues*) for tables in the database schema [IDC5.1.1Rev3]. *EvLoc* retrieves a *keyvalue* for the *keyname* “*orid*” when operating in location mode, and it retrieves a *keyvalue* for the *keyname* “*magid*” when operating in magnitude mode. When pre-existing origins or magnitudes are present in the database account, the values for *orid* and *magid* in the *lastid* table should be larger than the maximum values of these identifiers in the *origin* and *stamag* tables to avoid non-uniqueness errors.

TUXEDO FILES

EvLoc does not require a DACS configuration, but the IDC operational model necessitates its usage. When an *EvLoc* child process is added to a processing pipeline, the TUXEDO configuration must be updated. This update includes configuring the *ubbconfig* template, creating a disk queue, and creating a *tuxshell* parameter file. This section provides an overview for how to reconfigure the TUXEDO software to process an additional *EvLoc* child process, although the same general steps are applicable for adding child process(es) to other automatic processing pipelines. Additional details may be found in [BEA96] and [IDC6.5.2Rev0.1].

▼ Installation Procedures

Configuring ubbconfig Template

The `ubbconfig` file is a text file that contains the complete DACS configuration for the operational system. In practice, the `ubbconfig` file is built by compiling a template file. This template file, named `ubb_process.tpl` at the IDC, contains symbolic references to files and directories that are replaced with site-specific parameters by a script prior to compilation. Refer to [IDC6.5.2Rev0.1] for descriptions of the template file content.

When an *EvLoc* child process is added to a pipeline, the `ubbconfig` template must be appropriately configured. In particular, entries must be added to the `*SERVERS` and `*SERVICES` sections of the template. The entries in the `*SERVERS` section should look something like the following example:

```
tuxshell          SRVGRP=RCL_PRI          SRVID=920
                  CLOPT="-s EvLoc-mb_ave:tuxshell -o /dev/null -e /dev/null --
                  par=$(DISTRIBUTED-DIR)/tuxshell/recall/tuxshell-EvLoc-mb_ave.par"
tuxshell          SRVGRP=RCL_BAK          SRVID=10920
                  CLOPT="-s EvLoc-mb_ave:tuxshell -o /dev/null -e /dev/null --
                  par=$(DISTRIBUTED-DIR)/tuxshell/recall/tuxshell-EvLoc-mb_ave.par"
```

These entries specify that *tuxshell* is a generalized processing server belonging to the server groups `RCL_PRI` and `RCL_BAK`. The `CLOPT` string specifies the command line options used to call the *tuxshell* server. When the `EvLoc-mb_ave` service is requested, the *tuxshell* server executes using the `par` file specified on the `CLOPT` string as input. More information about this `par` file is given in "Creating Tuxshell Parameter Files" on page 154.

The forwarding agent *TMQFORWARD* dequeues messages from a specific disk queue and sends them to the *tuxshell* server that advertises the desired service (`EvLoc-mb_ave` in this example). There should also be *TMQFORWARD* entries in the `*SERVERS` section, such as:

```
TMQFORWARD        SRVGRP=QM_PRI        SRVID=5920
                  CLOPT="-- -i 10 -q EvLoc-mb_ave -t 1300"
TMQFORWARD        SRVGRP=QM_BAK        SRVID=15920
                  CLOPT="-- -i 10 -q EvLoc-mb_ave -t 1300"
```

These entries specify that *TMQFORWARD* belongs to the server groups *QM_PRI* and *QM_BAK*. The *CLOPT* string specifies the command line options used to call *TMQFORWARD*. There should be one *TMQFORWARD* for each *tuxshell*.

The entries in the **SERVICES* section should look something like the following example:

```
"EvLoc-mb_ave"  LOAD=10          SRVGRP=RCL_PRI
"EvLoc-mb_ave"  LOAD=20000       SRVGRP=RCL_BAK
```

These entries assign values to certain parameters for specific services (*EvLoc-mb_ave* in this example) in a given server group.

Creating Queues

The TUXEDO queuing system stores processing requests as messages in disk queues. Each queue holds requests for a certain service. *TMQFORWARD* dequeues messages from the queue and forwards them to *tuxshell* for processing.

When an *EvLoc* child process is added to a pipeline, a queue must be created to store requests for the corresponding *EvLoc* service. To create a new queue, an entry must be added to the script *crDacsQueues*. This entry should look something like the following example:

```
qcreate EvLoc-mb_ave priority,time top,msgid 2 30 80% 0%
"$CMS_SCRIPTS/bin/mailFullQ EvLoc-mb_ave"
```

The queue name in this example is "*EvLoc-mb_ave*". By convention, the queue names and the service names are identical.

Queues can only be created while configuring the DACS system. To add a new queue, halt system processing, erase existing queues, create new queues with the amended *crDacsQueues* script, and restart processing. For more information, refer to [IDC6.5.2Rev0.1].

▼ Installation Procedures

Creating Tuxshell Parameter Files

The *tuxshell* par file contains parameters that specify how to build a command line to invoke the child process. These parameters include the path to, and the name of, the child process par file that is used in the call to the child process.

When an *EvLoc* child process is added to a pipeline, a *tuxshell* par file for that service must be created and installed in an appropriate location in the configuration tree. Figure 16 shows a sample *tuxshell* par file for an *EvLoc* child process.

```

role=EvLoc-mb_ave
prefix=EvLoc

par=$(IMSPAR)
par=$(DISTRIBUTED)

sendkeys=0

EvLoc-exec=$(RELBIN)/EvLoc

EvLoc-key[0]=time
EvLoc-key[1]=endtime

# Put these values on the command line.
EvLoc[1]="start-time=$(time)"
EvLoc[2]="end-time=$(endtime)"
EvLoc[3]="par=$(AUTOMATIC-DIR)/EvLoc/EvLoc-recall.par"

EvLoc-timeout=1200
EvLoc-true-exit=0,2

# Queuing Flow
#
interval-source=queue
destqueue=EvLoc-mlppn

log=$(LOGDIR)/%jdate/tuxshell/$(role)-%host-%pid
outfile=$(LOGDIR)/%jdate/Recall/$(role)
errfile=$(outfile)

```

FIGURE 16. SAMPLE TUXSHELL PARAMETER FILE

Some of the important parameters specified in the *tuxshell* par file are the service name, the executable filesystem path, and the child process command line arguments. The *role* parameter identifies the name of the service (*EvLoc-mb_ave* in this example). The *EvLoc-exec* parameter specifies the filesystem path to the exe-

cutable. Finally, the *EvLoc[i]* array contains arguments placed on the command line by *tuxshell* when it spawns the child process. The string “EVLOC” used in the parameter names is unique to this service. Other *tuxshell* par files replace “EVLOC” with names describing the offered service.

INITIATING OPERATIONS

Before initiating *EvLoc* or any application that uses *libloc* or *libmagnitude* functionality, verify that the following installation and configuration steps have been completed:

- The hardware infrastructure is physically configured.
- The executable binaries are installed in the operational filesystem.
- The system data files are installed and configured in the operational configuration filesystem.
- The database server is installed and configured.
- Input data are present in the input database tables.
- If used in an operational system, the DACS is configured to initiate processing.

When the installation and configuration process is complete, invoke the application. See “Software Startup” on page 20 for the *EvLoc* invocation procedures. Refer to “Software Startup” in other software user manuals for the invocation procedures of other applications.

VALIDATING INSTALLATION

To validate the installation and operation of *EvLoc*, use the following techniques:

- Monitor *EvLoc* processing in the pipeline using the *WorkFlow* tool and ensure that intervals are being properly queued and processed with no failures.
- Perform database queries to ensure that event locations and/or magnitudes are being computed.

▼ Installation Procedures

- Monitor the log file to ensure that the software operates as intended.

Refer to “Monitoring” on page 92 for additional details on validating installation and monitoring *EvLoc* processing.

To validate the installation and operation of other applications that link to the *libloc* and *libmagnitude* shared libraries, refer to the “Validating Installation” section of the associated software user manual.

References

The following sources supplement or are referenced in the document:

- [BEA96] BEA Systems, Inc., *BEA TUXEDO Reference Manual*, 1996.
- [Gan79] Gane, C., and Sarson, T., *Structured Systems Analysis: Tools and Techniques*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979.
- [IDC5.1.1Rev3] Science Applications International Corporation, Veridian Systems, *Database Schema, (Part 1, Part 2, and Part 3), Revision 3*, SAIC-01/3052, TN-2866, 2001.
- [IDC5.2.1Rev1] Science Applications International Corporation, *IDC Processing of Seismic, Hydroacoustic, and Infrasonic Data, Revision 1*, 2001.
- [IDC6.2.1] Science Applications International Corporation, *Release 2 Operations and Maintenance - Seismic, Hydroacoustic, and Infrasonic System*, SAIC-00/3000, 2000.
- [IDC6.2.5] Science Applications International Corporation, *Analyst Instructions for Seismic, Hydroacoustic, and Infrasonic Data*, SAIC-98/3002, 1998.
- [IDC6.5.2Rev0.1] Science Applications International Corporation, *Distributed Application Control System (DACs) Software User Manual, Revision 0.1*, SAIC-00/3038, 2000.
- [IDC6.5.11] Science Applications International Corporation, *Station Processing (StaPro) Software User Manual*, SAIC-01/3036, 2001.

▼ References

- [IDC6.5.12] Science Applications International Corporation, *Global Association (GA) Subsystem Software User Manual*, SAIC-01/3003, 2001.
- [IDC6.5.16] Science Applications International Corporation, *Surface Wave Identification and Measurement Software User Manual*, SAIC-01/3014, 2001.
- [IDC7.1.3] Science Applications International Corporation, *Surface Wave Identification and Measurement*, SAIC-01/3008, 2001.
- [IDC7.1.5] Science Applications International Corporation, *Event Location Software*, SAIC-01/3010, 2001.
- [IDC7.1.6] Science Applications International Corporation, *Event Magnitude Software*, SAIC-01/3011, 2001.
- [Ker88] Kernighan, B.W., and Ritchie, D.M., *The C Programming Language, Second Edition*, Prentice Hall, Englewood Cliffs NJ, 1988.
- [Kun02] Kung, Y. and Bondar, I., *Hypocenter Location Server Release 1.0*, CCB-PRO-02/07Rev.1, 2002.
- [Nag96] Nagy, W., *New Region-Dependent Travel-time Handling Facilities at the IDC; Functionality, Testing and Implementation Details*, Science Applications International Corporation, SAIC-96/1179, 1996.
- [Vei72] Veith, K. F., and Clawson, G. E., "Magnitude of Short-period P Wave Data," *Bulletin of the Seismological Society of America*, Volume 62, pp. 435–453, 1972.

Glossary

Symbols

1-D

One-dimensional.

1-D travel-time table

See one-dimensional travel-time table.

2-D

Two-dimensional.

A

amplitude

Zero-to-peak height of a waveform in nanometers.

amtype

Descriptor that uniquely identifies an amplitude measurement type (for example, A5 / 2 or SBSNR).

analyst

Personnel responsible for reviewing and revising the results of automatic processing.

Analyst Review Station

This application provides tools for a human analyst to refine and improve the event bulletin by interactive analysis.

arid

Arrival identifier.

arrival

Detected signal that has been associated to an event. First, the Global Association (GA) software associates the detection to an event. Later, during interactive processing, many arrivals are confirmed, improved, or added by visual inspection.

arrival-based amplitude

Amplitude measured by *DFX* for a detected signal.

ARS

See Analyst Review Station.

ASCII

American Standard Code for Information Interchange. Standard, unformatted 256-character set of letters and numbers.

attribute

(1) Database column. (2) Characteristic of an item; specifically, a quantitative measure of a S/H/I detection such as azimuth, slowness, period, or amplitude.

▼ Glossary

azimuth

Direction, in degrees clockwise with respect to North, from a station to an event.

B**bootstrap resampling**

Statistical technique of random resampling of data elements with replacement (that is, without regard to which elements have already been selected) used to estimate errors in parameters estimated from a distribution.

bulk station correction

Empirical station- and TLtype-specific term added to the logarithm of the amplitude during computation of a station magnitude to correct for local station effects.

bulk station correction error

Estimate of the standard error associated with the bulk station correction.

C**chan.**

Channel.

channel

Component of motion or distinct stream of data.

child process

UNIX process created by the *fork* routine. The child process is a snapshot of the parent at the time it called *fork*.

CMR

Center for Monitoring Research.

command

Expression that can be input to a computer system to initiate an action or affect the execution of a computer program.

comments

Free text field containing comments made by a station operator or IDC analyst.

component

(1) One dimension of a three-dimensional signal; (2) The vertically or horizontally oriented (north or east) sensor of a station used to measure the dimension; (3) One of the parts of a system; also referred to as a module or unit.

Computer Software Component

Functionally or logically distinct part of a computer software configuration item; possibly an aggregate of two or more software units.

Computer Software Configuration Item

Aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process.

configuration

(1) (hardware) Arrangement of a computer system or components as defined by the number, nature, and interconnection of its parts. (2) (software) Set of

adjustable parameters, usually stored in files, which control the behavior of applications at run time.

connection

Open communication path between protocol peers.

crash

Sudden and complete failure of a computer system or component.

CSC

See Computer Software Component.

CSCI

See Computer Software Configuration Item.

D**DACS**

See Distributed Application Control System.

data flow

Sequence in which data are transferred, used, and transformed during the execution of a computer program.

defining

Arrival attribute, such as arrival time, azimuth, or slowness, which is used in calculating the event's location or magnitude.

defining arrival

Arrival whose attributes (time, azimuth, and/or slowness) are used to compute an event location.

defining magnitude

Station magnitude that is used to compute a network magnitude.

deg.

Degrees (as a distance).

dequeue

Remove a message from a Tuxedo queue.

detection

Probable signal that has been automatically detected by the Detection and Feature Extraction (*DFX*) software.

DFX

Detection and Feature Extraction. *DFX* is a programming environment that executes applications written in Scheme (known as *DFX* applications).

dispersion

Expansion of the length of a seismic wavetrain due to each wavelength travelling with its own velocity.

Distributed Application Control System

This software supports inter-application message passing and process management.

E**earth model definitions**

Structured collection of velocity (travel-time) model information. The information is organized and specified by a Velocity Model Specification File (*VMSF*).

▼ Glossary

EC table

See Ellipticity Correction file.

ellipticity correction

Travel-time correction accounting for deviations in the earth's shape from a perfect sphere.

Ellipticity Correction file

File that specifies distance- and depth-dependent ellipticity correction coefficients to a 1-D velocity model for a particular phase.

EM

See expectation maximization algorithm.

event

Unique source of seismic, hydroacoustic, or infrasonic wave energy that is limited in both time and space.

event

Event identifier.

EvLoc

Application used to compute event location and/or magnitude.

execute

Carry out an instruction, process, or computer program.

expectation maximization algorithm

Algorithm used to approximate a probability density function during computation of maximum likelihood magnitude estimates.

F**failure**

Inability of a system or component to perform its required functions within specified performance requirements.

filesystem

Named structure containing files in sub-directories. For example, UNIX can support many filesystems; each has a unique name and can be attached (or mounted) anywhere in the existing file structure.

fork

UNIX system routine that is used by a parent process to create a child process.

forwarding agent

Application server *TMQFORWARD* that acts as an intermediary between a message queue on disk and a group of processing servers advertising a service. The forwarding agent uses transactions to manage and control its forwarding function.

FTP

File Transfer Protocol; protocol for transferring files between computers.

function

Named section of a program that performs a particular task.

G**GA**

See Global Association.

GB

See gigabyte.

generalized processing server

DACS application server (*tuxshell*) that is the interface between the DACS and the automatic processing system. It executes application programs as child processes.

gigabyte

Measure of computer memory or disk space that is equal to 1,024 megabytes.

Global Association

Subsystem that associates S/H/I phases to events.

H**hydroacoustic**

Pertaining to sound in the ocean.

I**IASPEI**

International Association of Seismology and Physics of the Earth's Interior.

ID

Identification; identifier.

IDC

International Data Centre.

infrasonic (infrasound)

Pertaining to low-frequency (sub-audible) sound in the atmosphere.

insufficient data

Category of S/H/I events that lack adequate measurements to apply any of the event-screening criteria.

invoke

Execute or run an application, script, or program.

IPC

Interprocess communication. The messaging system by which applications communicate with each other through *libipc* common library functions. See *tuxshell*.

K**km**

Kilometer.

L**lat.**

Latitude.

Late Event Bulletin

List of analyst reviewed S/H/I events and event parameters (origin and associated arrival information).

LEB

See Late Event Bulletin.

libgdi

Library containing functions for RDBMS access.

▼ Glossary

LMID

See logical machine identifier.

local

(1) (distance) Source to seismometer separations of a few degrees or less. (2) (event) Recorded at distances where the first P and S waves from shallow events have traveled along direct paths within the crust.

Location Test-site Correction file

File that specifies travel-time corrections to a 1-D velocity model by region and station/phase, where region is a defined geographic area, such as a nuclear test-site region.

logical machine identifier

Logical reference to a machine used by a Tuxedo application. LMIDs can be descriptive, but they should not be the same as the UNIX hostname of the machine.

lon.

Longitude.

Long-period Grid Index file

File that specifies phase velocity table indices for a particular long-period phase.

Long-period Phase Velocity file

File that specifies indexed tables of phase velocities for a particular long-period phase.

Love wave

A seismic phase that propagates with transverse particle motion along the surface of the earth.

lower-bound magnitude

Arithmetic mean of a set of station magnitudes that were computed from clipped amplitudes.

LP

Long period.

LPGI file

See Long-period Grid Index file.

LPPV file

See Long-period Phase Velocity file.

LQ

See Love wave.

LR

Rayleigh wave. A seismic phase that travels along the surface of the earth.

LTSC file

See Location Test-site Correction file.

M**magnitude**

Empirical measure of the size of an event (usually made on a log scale).

magnitude correction

A correction added to the logarithm of the amplitude during computation of a station magnitude.

Magnitude Description File

File that maps amptypes and TLtypes to magtypes and specifies magnitude control settings and bulk station correction data.

magnitude-defining

See defining magnitude.

Magnitude Test-site Correction file

File that specifies magnitude corrections by region and station/*magtype*, where region is a defined geographic area, such as a nuclear test-site region.

magtype

Descriptor that uniquely identifies a computed magnitude type (for example, mb_ave or mb_mle).

maximum likelihood estimate

Estimate of one or more unknown model parameters that maximize the probability of obtaining a particular sample data set.

MB

See megabyte.

 m_b

Magnitude estimated from seismic body waves.

MDF

See Magnitude Description File.

megabyte

1,024 kilobytes.

 M_L

Magnitude estimated from seismic waves measured near the source.

MLE

See maximum likelihood estimate.

modeling error

Estimate of the component of error in a travel-time or magnitude calculation that reflects the uncertainty in the corresponding earth model.

 M_s

Magnitude of seismic surface waves.

MSO

Monitoring Systems Operation. An organizational unit within SAIC.

MTSC file

See Magnitude Test-site Correction file.

N**network**

Spatially distributed collection of seismic, hydroacoustic, or infrasonic stations for which the station spacing is much larger than a wavelength.

network processing

Processing that uses the results of Station Processing from a network of stations to define and locate events.

network-average magnitude

Arithmetic mean of a set of station magnitudes computed from arrival-based amplitudes.

▼ Glossary

nondefining

Arrival attribute, such as arrival time, azimuth, or slowness, which is associated, but not used in calculating the event's location or magnitude.

nondefining arrival

Arrival whose measured attributes (time, azimuth, and/or slowness) are not used in a particular location computation.

nondefining magnitude

Station magnitude that is not used to compute a network magnitude.

NULL

Empty, zero.

O**one-dimensional travel-time table**

S/H/I: File that specifies travel times and modeling errors discretized in one or two dimensions based on a 1-D velocity model for a particular S/H/I phase.

operations database

Relational database used by the operational system.

ORACLE

Vendor of the database management system used at the PIDC and IDC.

orid

Origin Identifier.

origin

Hypothesized time and location of a seismic, hydroacoustic, or infrasonic event. An event may have many origins. Characteristics such as magnitudes and error estimates may be associated with an origin.

origin-based amplitude

Amplitude measured in a time window computed from the predicted travel time from the origin.

P**parameter**

User-specified token that controls some aspect of an application (for example, database name, threshold value). Most parameters are specified using [*token* = *value*] strings, for example, `dbname=mydata/base@oracle`.

parameter (par) file

ASCII file containing values for parameters of a program. Par files are used to replace command line arguments. The files are formatted as a list of [*token* = *value*] strings.

parrival

Database table that contains the predicted arrivals and associations for origin-based amplitude measurements.

parse

Decompose information contained in a set of data.

pathname

Filesystem specification for a file's location.

period

Average duration of one cycle of a phase, in seconds per cycle.

phase

Arrival that is identified based on its path through the earth.

phase name

Name assigned to a seismic, hydroacoustic or infrasonic arrival associated with a travel path.

pipeline

1) Flow of data at the IDC from the receipt of communications to the final automated processed data before analyst review. 2) Sequence of IDC processes controlled by the DACS that either produce a specific product (such as a Standard Event List) or perform a general task (such as station processing).

post-analysis processing

Automated processing that occurs after analysts have reviewed the automatic event bulletins.

post-location processing

Software that computes various magnitude estimates and selects data to be retrieved from auxiliary stations.

process

Function or set of functions in an application that perform a task.

program

Organized list of instructions that, when executed, causes the computer to behave in a predetermined manner. A program contains a list of variables and a list of statements that tell the computer what to do with the variables.

Q**query**

Request for specific data from a database.

R**Radial 2-D Travel-time table**

File that specifies azimuth- and distance-dependent travel times and modeling errors for a 2-D velocity model associated with a particular station, phase, and time period of the year.

RAM

Random Access Memory.

Rayleigh wave

A seismic phase that propagates with radial elliptical particle motion along the surface of the earth.

real time

Actual time during which something takes place.

▼ Glossary

recovery

Restoration of a system, program, database, or other system resource to a state in which it can perform required functions.

regional

(1) (distance) Source-to-seismometer separations between a few degrees and 20 degrees. (2) (event) Recorded at distances where the first P and S waves from shallow events have traveled along paths through the uppermost mantle.

residual

Difference between the observed value for an attribute (for example, time, azimuth, slowness, or magnitude) and its corresponding theoretical value.

rms

Root mean square.

root name

Base name in a filename, as distinguished from the path or suffix (for example, `qfvc` is the root name in the filename `qfvc.mb`).

run

(1) Single, usually continuous, execution of a computer program. (2) To execute a computer program.

S**s**

Second(s) (time).

SAIC

Science Applications International Corporation.

SASC file

See Slowness/Azimuth Station Correction file.

schema

Database structure description.

Scheme

Dialect of the Lisp programming language that is used to configure some IDC software.

seismic

Pertaining to elastic waves traveling through the earth.

server

Software module that accepts requests from clients and other servers and returns replies.

server (group)

Set of servers that have been assigned a common GROUPNO parameter in the `ubbconfig` file. All servers in one server group must run on the same logical machine (LMID). Servers in a group often advertise equivalent or logically related services.

service

Action performed by an application server. The server is said to be advertising that service. A server may advertise several services (multiple personalities), and several servers may advertise the same service (replicated servers).

S/H/I

Seismic, Hydroacoustic, and Infrasonic.

shutdown

Action of terminating a server process as a memory-resident task. Shutting down the whole application is equivalent to terminating all specified server processes (admin servers first, application servers second) in the reverse order that they were booted.

site

Location of a sensor within a station.

slowness

Inverse of velocity, in seconds/degree; a large slowness has a low velocity.

Slowness/Azimuth Station Correction file

File that specifies slowness/azimuth corrections, modeling errors, and optional rotation coefficients for a particular station.

slowness vector

Vector in 2-D wavenumber space. The magnitude of the vector corresponds to the inverse of the phase velocity of a traveling plane wave. The direction of the vector is usually defined as being from the station to the source.

Solaris

Name of the operating system used on Sun Microsystems hardware.

Source-specific Station Correction file

File that specifies source-specific, regional travel-time corrections to a 1-D velocity model for a particular station/phase pair.

SQL

Structured Query Language; a language for manipulating data in a relational database.

SSSC file

See Source-specific Station Correction file.

sta

Station.

StaPro

Station Processing application for S/H/I data.

station

Collection of one or more monitoring instruments. Stations can have either one sensor location (for example, BGCA) or a spatially distributed array of sensors (for example, ASAR).

station code (or ID)

(1) Code used to identify distinct stations. (2) Site code.

station processing

Processing based on data from a single station.

station weight

Inverse square of the station uncertainty.

▼ Glossary

subsystem

Secondary or subordinate system within the larger system.

surface wave

Seismic wave propagating along the surface of the earth.

T**TL**

Transmission loss.

TLM

See Transmission Loss Model.

TLSF

See Transmission Loss Specification File.

TLtype

Descriptor that uniquely identifies the transmission-loss model associated with a particular magnitude type (for example, mb or ms).

Transmission Loss Model

Distance- and depth-dependent magnitude corrections and modelling errors for an attenuation curve associated with a particular TLtype and optional phase type and channel/frequency identifier.

Transmission Loss Specification File

File that specifies all mappings between global and regional transmission loss types and models.

travel time

Time necessary for a phase to propagate from source to receiver through the earth, ocean, or atmosphere.

TT

See travel time.

Tuxedo

Transactions for UNIX Extended for Distributed Operations.

tuxshell

Process in the Distributed Processing CSCI used to execute and manage applications. See IPC.

U**ubbconfig file**

Human readable file containing all of the Tuxedo configuration information for a single DACS application.

uncertainty

Estimate of the deviation from the true mean for the parameter or variable of interest.

UNIX

Trade name of the operating system used by the Sun workstations.

upper-bound magnitude

Arithmetic mean of a set of station magnitudes computed from origin-based amplitudes.

V

version

Initial release or re-release of a computer software component.

Velocity Model Specification File

File for setting travel-time models in S/H/I event location.

VMSF

See Velocity Model Specification File.

W

WaveExpert

Application in the Automatic Processing CSCI that determines data intervals to request from auxiliary stations.

weighted-average magnitude

Network magnitude that is estimated by weighting each defining station magnitude by its station weight.

WorkFlow

Software that displays the progress of automated processing systems.

workstation

High-end, powerful desktop computer preferred for graphics and usually networked.

Index

Numerics

- 1-D travel-time (TT) tables 39
- 2-D radial travel-time tables 48

A

- adding new stations 88

B

- bulk magnitude station correction parameters 68

C

- control parameters 22
- conventions
 - data flow symbols iv
 - format codes 33
 - typographical v
- COTS libraries 12

D

- database 87, 150
 - updating static tables 88

- database tables 12
- data flow symbols iv

E

- ellipticity correction table 49
- environment
 - hardware 16
 - software 16
- error messages 93
 - configuration-related
 - EvLoc* 101
 - libloc* 110
 - libmagnitude* 123
 - index 94
 - processing-related
 - libloc* 137
 - libmagnitude* 143
- EvLoc*
 - alternate states of operation 17
 - control parameters 22
 - creating a parameter file 22
 - database tables 12
 - error messages 93
 - configuration-related 101
 - index 94
 - error recovery 146
 - features and capabilities 8
 - functionality 8
 - general parameters 24
 - initiating operations 155
 - location control parameters 26
 - log files 87, 93
 - magnitude control parameters 28
 - magnitude specification parameters 66
 - monitoring 92
 - overview 2
 - parameter files 149

▼ Index

- performance 10
- relationship to operational scripts 7
- shutdown 21
- solving common problems 145
- startup 20
- TLSF 69
- TUXEDO files 151
- validating installation 155

F

- features and capabilities 8
- format codes 33
- formats
 - bulk magnitude station correction
 - parameters 68
 - distance/depth-dependent travel-time
 - modeling error 46
 - distance-dependent travel-time
 - modeling error 44
 - LPGI file 56
 - LPPV file 58
 - LTSC file 54
 - MTSC files 86
 - SASC file 61
 - single-value travel-time modeling
 - error 43
 - SSSC file 51
 - TLM distance/depth-dependent
 - errors 83
 - TLM distance-dependent errors 82
 - TLM file 78
 - TLM pathway parameters 71
 - TLM single-value errors 81
 - travel-time data 39
 - VMSF 35

G

- general parameters 24

H

- hardware 16, 148

I

- input files 14, 149

L

- lastid** 151

- libloc*
 - 1-D TT table 39
 - applications that use 6
 - ellipticity correction table 49
 - error messages
 - configuration-related 110
 - processing-related 137
 - features and capabilities 8
 - functionality 8
 - input files 14
 - location earth-model files 149
 - LPGI file 55
 - LPPV file 58
 - LTSC file 53
 - overview 2
 - performance 10
 - radial 2-D travel-time tables 48
 - SASC file 60
 - SSSC table 49
 - VMSF 35

- libmagnitude*
 - applications that use 6
 - error messages
 - configuration-related 123
 - processing-related 143
 - features and capabilities 8
 - functionality 8
 - input files 15
 - magnitude earth-model files 150
 - overview 2

- performance 10
- libraries 12
- location control parameters 26
- location input files
 - modifying 88
- Location Test-site Correction (LTSC) file 53
- log files 87, 93
- Long-period Grid Index (LPGI) file 55
- Long-period Phase Velocity (LPPV) file 58
- LPGI file 55
 - format 56
- LPPV file 58
 - format 58
- LTSC file 53
 - format 54
 - modifying 88

M

- magnitude control parameters 28
- Magnitude Description File (MDF) 64
- magnitude specification parameters 66
- Magnitude Test-site Correction (MTSC)
 - files 85
- maintenance 87
- man pages iii
- MDF 64
 - modifying 89
- models
 - integrating new 88
- monitoring *EvLoc* 92
- MTSC files 85
 - format 86

P

- performance 10
- processing flow model 4
- processing status 92

S

- SASC file 60
 - format 61
 - modifying 88
- shared libraries 12
- Slowness/Azimuth Station Correction (SASC)
 - file 60
- software 16
 - maintenance 87
 - obtaining 148
- Source-specific Station Correction (SSSC)
 - table 49
- SSSC file 49
 - format 51
 - modifying 88
- stations
 - adding new 88
- status of processing 92

T

- TLM 76
 - default pathnames 73
 - description parameters 71
 - file format 78
 - formats for distance/depth-dependent errors 83
 - formats for distance-dependent errors 82
 - formats for single-value errors 81
 - pathway parameters 71
 - station-specific description parameters 74
 - station-specific pathname examples 76
- TLSF 69
 - modifying 89
- Transmission Loss Model (TLM) 76
- Transmission Loss Specification File (TLSF) 69
- travel-time data format 39
- travel-time modeling error

▼ Index

- distance/depth-dependent format 46
- distance-dependent format 44
- single-value format 43
- travel-time tables
 - 1-D 39
 - radial 2-D 48
- troubleshooting 91
- TUXEDO
 - creating queues 153
 - files 151
- tuxshell*
 - creating par files 154
- typographical conventions v

U

- ubbconfig
 - configuring 152

V

- Velocity Model Specification File (VMSF) 35
- verbosity 25
- VMSF 35
 - format 35
 - modifying 88

W

- WorkFlow* 92