THE 2004 KNOWLEDGE BASE PARAMETRIC GRID DATA SOFTWARE SUITE

Sandy Ballard, Marcus C. Chang, James R. Hipp, Lee A. Jensen, Randall W. Simons, and Lisa K. Wilkening

Sandia National Laboratories

Sponsored by National Nuclear Security Administration
Office of Nonproliferation Research and Engineering
Office of Defense Nuclear Nonproliferation

Contract No. DE-AC04-94AL85000

ABSTRACT

One of the most important types of data in the National Nuclear Security Administration (NNSA) Ground-Based Nuclear Explosion Monitoring Research and Engineering (GNEM R&E) Knowledge Base (KB) is parametric grid (PG) data. PG data can be used to improve signal detection, signal association, and event discrimination, but so far their greatest use has been for improving event location by providing ground-truth-based corrections to travel-time base models. In this presentation we discuss the latest versions of the complete suite of Knowledge Base PG tools developed by NNSA to create, access, manage, and view PG data.

The primary PG population tool is the Knowledge Base calibration integration tool (KBCIT). KBCIT is an interactive computer application to produce interpolated calibration-based information that can be used to improve monitoring performance by improving precision of model predictions and by providing proper characterizations of uncertainty. It is used to analyze raw data and produce kriged correction surfaces that can be included in the Knowledge Base. KBCIT not only produces the surfaces but also records all steps in the analysis for later review and possible revision. New features in KBCIT include a new variogram autofit algorithm; the storage of database identifiers with a surface; the ability to merge surfaces; and improved surface-smoothing algorithms.

The Parametric Grid Library (PGL) provides the interface to access the data and models stored in a PGL file database. The PGL represents the core software library used by all the GNEM R&E tools that read or write PGL data (e.g., KBCIT and LocOO). The library provides data representations and software models to support accurate and efficient seismic phase association and event location. Recent improvements include conversion of the flat-file database (FDB) to an Oracle database representation; automatic access of station/phase tagged models from the FDB during location; modification of the core geometric data representations; a new multimodel representation for combining separate seismic data models that partially overlap; and a port of PGL to the Microsoft Windows platform.

The Data Manager (DM) tool provides access to PG data for purposes of managing the organization of the generated PGL file database, or for perusing the data for visualization and informational purposes. It is written as a graphical user interface (GUI) that can directly access objects stored in any PGL file database and display it in an easily interpreted textual or visual format. New features include enhanced station object processing; low-level conversion to a new core graphics visualization library, the visualization toolkit (VTK); additional visualization support for most of the PGL geometric objects; and support for the Environmental Systems Research Institute (ESRI) shape files (which are used to enhance the geographical context during visualization).

The Location Object-Oriented (LocOO) tool computes seismic event locations and associated uncertainty based on travel time, azimuth, and slowness observations. It uses a linearized least-squares inversion algorithm (the Geiger method), enhanced with Levenberg-Marquardt damping to improve performance in highly nonlinear regions of model space. LocOO relies on PGL for all predicted quantities and is designed to fully exploit all the capabilities of PGL that are relevant to seismic event location. New features in LocOO include a redesigned internal architecture implemented to enhance flexibility and to support simultaneous multiple event location. Database communication has been rewritten using new object-relational features available in Oracle 9i.

OBJECTIVE

Monitoring sensor data for possible nuclear explosions involves four fundamental tasks: signal detection, signal association, event location, and event identification. For all but the first of these, the task is accomplished by minimizing the misfit between observations and theoretical predictions, which in turn depend on the model of the Earth used. Thus, improving the quality of the earth model used for monitoring, is of critical importance, and is the primary focus of the GNEM R&E efforts.

Having a better earth model is not enough, however; if the model is to be maintained and used for operational monitoring, we need a cohesive set of software tools to leverage the improved model benefits. Population tools are needed to make it easy to process the data into the proper format, and to update the model when new data become available. The earth model must be designed and defined into an integrated and extensible software system. An interface is needed around the model to serve predictions to various applications. An efficient, extensible storage format is needed to archive the model on disk. Finally, some sort of data management and viewing utility is needed to easily allow users to browse through the model to better understand it, and to make simple edits, such as deletions and insertions, as needed. Once the models are thoroughly integrated into a software system, they must be used by the seismic event analysis tools such as Sandia National Laboratories (SNL) LocOO code that computes event locations and associated uncertainty based on travel time, azimuth, and horizontal slowness observations (Ballard, 2002).

Researchers at SNL have been working to meet these needs by developing an integrated suite of tools collectively expressed as the Parametric Grid Software Suite (PGSS).

RESEARCH ACCOMPLISHED

The PGSS presently consists of the primary data population tool KBCIT, the low-level storage, representation, and model library PGL; the database management tool, DM; and a seismic location tool, LocOO. We briefly describe each tool below accompanied by a discussion of the new enhancements incorporated over the previous year. See Hipp et al. (2002) for more information concerning the basic operation of each tool.

Knowledge Base Calibration Integration Tool

The Knowledge Base Calibration Integration Tool (KBCIT) is an interactive computer application whose goal is to produce interpolated calibration-based information that can be used to improve monitoring performance by improving precision of model predictions and by providing proper characterizations of uncertainty. It is used to analyze raw data and produce kriged correction surfaces that can be included in the KB. The KB calibration integration tool not only produces the surfaces, but also records every step in the analysis for later review and possible revision.

KBCIT allows the user to define the boundaries of regions, where the data in each region can be treated differently, including using different base models, trend removals, outlier limits, and variograms in each region (Figure 1). KBCIT allows the user to set parameters for blending functions to apply across region boundaries to avoid discontinuities. A declustering operation is available to reduce dataset size by removing redundancy, and stabilize kriging by averaging values close to one another.

KBCIT uses the PGL, described below, to implement both stationary and nonstationary Bayesian kriging for interpolation of empirical data with uncertainties (Figure 2). The results of kriging can be cross validated by comparing empirical values with interpolated predictions based on a kriged surface formed using all but the values to be compared.

To provide a faster alternative to interpolating a kriged surface, the user can specify a tessellation on which KBCIT will precompute kriged values. Any combination of kriged and tessellated surfaces can be summed or differenced and viewed with an interactive three-dimensional (3D) viewer. Kriged and tessellated surfaces can be saved in a Parametric Grid database for use by other programs, such as those doing event location.

New features added to KBCIT this year (2004) include the following:

1. New variogram autofit algorithm – added a new algorithm to fit a curve to variogram data, allowing multiple degrees of freedom (nugget, sill, range, and fraction of curve used) using a simplex optimization method.

2. Storage of database identifiers with a surface – Origin ID, Event ID, and Arrival ID from the NNSA schema database can now be input with the observation data, and will be stored with the kriged surface in the file database.

3. Ability to merge surfaces – added support for merging multiple kriged surfaces, specifying boundaries for where each surface applies, and specifying how to smooth between surfaces. This capability is expected to be useful for merging results from two or more researchers producing correction surfaces for the same station/phase.

4. Improved surface-smoothing algorithms – better algorithms for smoothing between regions to avoid discontinuities in the resulting surface.

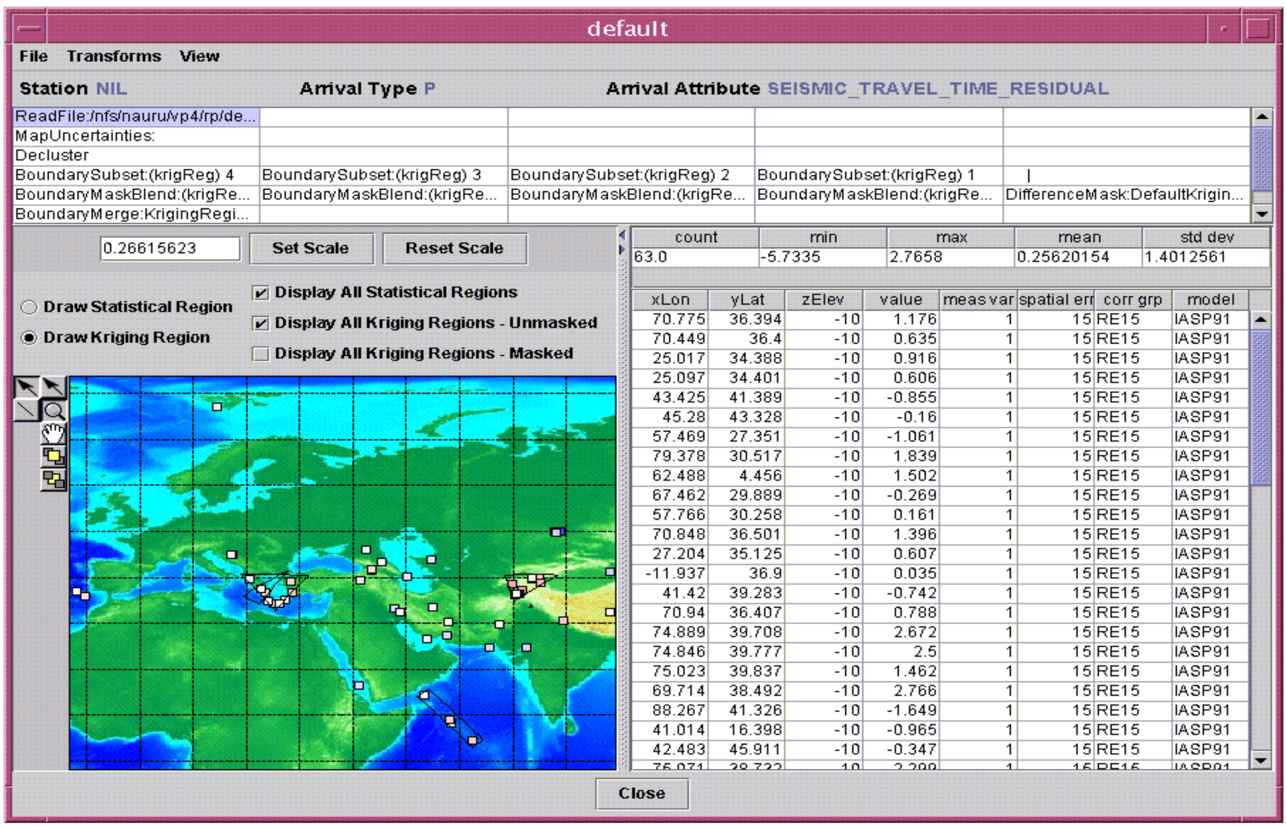

Figure 1. Kriging set window in KBCIT.

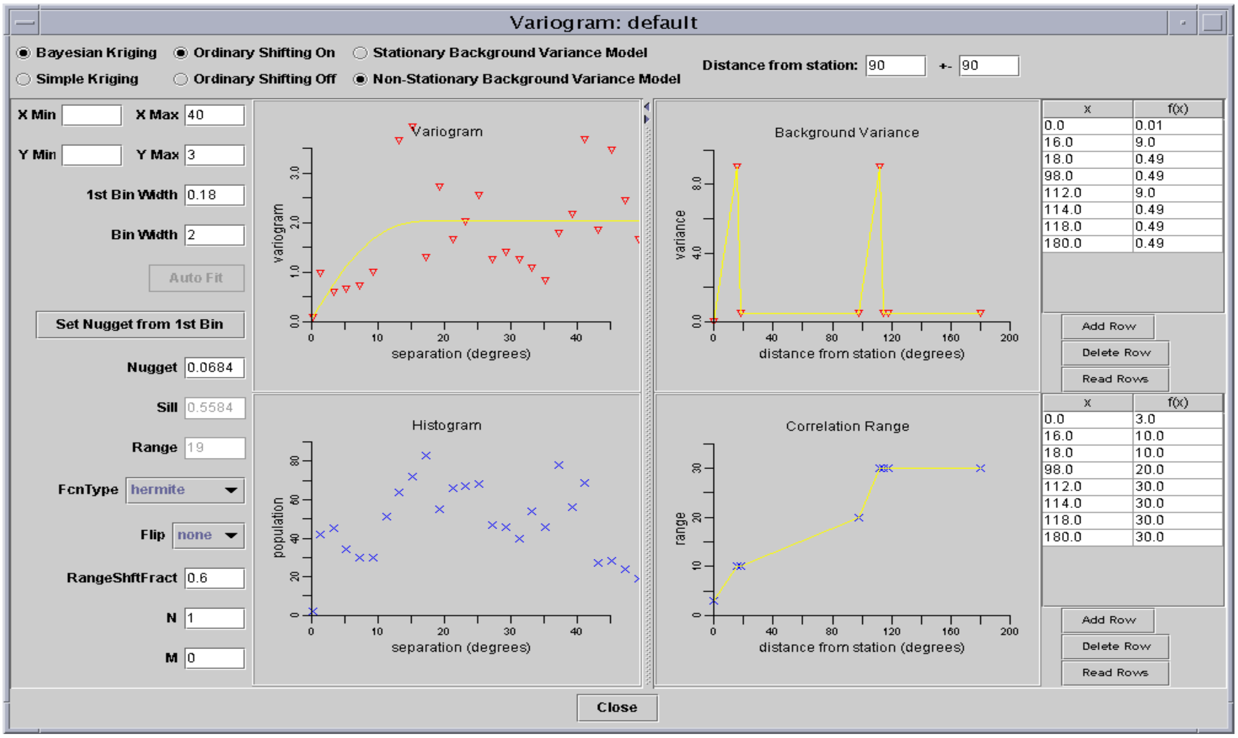

Figure 2. Variogram window in KBCIT.

Parametric Grid Library

The parametric grid library (PGL) provides the internal representation and client interfaces to access the data and models used by all the GNEM R&E tools. This library includes tools that provide a means of populating the PGL (e.g., KBCIT described above) and those that require access to its data or models (e.g., LocOO).

Several improvements were made to the 2004 version of PGL including:

1. The addition of a new PGL Oracle database (ODB) storage mechanism. Previously, a flat-file database stored all of the PGL data files directly into the file system. Now, an ODB can be used to store the PGL data files in an Oracle 9i database. Because the code utilizes the new Oracle C++ call interface (OCCI) Oracle 9i is required. Each PGL object is stored as a binary large object (BLOB) in the database, with the MD5 key string set as its primary key. Translation functionality has been implemented to allow the users to translate from an FDB to an ODB, and vice-versa. With the new ODB, station data from the site table is used to create station objects in the ODB. When merging stations from an FDB into an ODB, the translation utility will attempt to resolve conflicts (stations with identical names) by using the existing ODB station instead; this is done if the two stations have matching data, or if there are no dependencies on the FDB station. However, if other FDB objects are dependent on the conflicting FDB station, errors describing the conflicts will be output to a log file indicating that those FDB objects were not added to the ODB. The user, as a result, has the option of forcing a switch of dependencies on an FDB station to an ODB station, thus enabling those FDB objects to be added to the ODB.

2. Extensive modifications were made to the internal data structure of an FDB, and automatic translation functionality is provided to update an existing FDB to the latest version. FDB versions were introduced to solve several user issues including (a) a simplified and cleaner FDB file structure, (b) forcing the MD5 hash key to depend only on an object's binary data and its dependencies (objects required for instantiation) instead of its metadata attributes, which do not guarantee uniqueness, and (c) ensuring MD5 key cross-platform compatibility, with little-endian architectures (Intel x86, etc., see 7 below).

3. We began a major design renovation that collects the set of mid-level PGL data representation objects into a single interface object called a *DataModel*. The *DataModel* object uses a consistent collection of objects with which any generic geometric data-set representation can be described. The objects include (1) an abstract *Grid* object that defines the geometric position, dimensionality, and topology of the data, (2) a set of point and cell attributes, stored in new data-array objects, that are defined at each point or cell of the grid (3) an *Interpolator* object that is used to interpolate the attributes over the grid using an arbitrary user defined order, and (4) a *Polygon* object that is used to constrain the influence of the *DataModel* to a region that may be more restrictive than the *DataModel* grid. The concrete implementations of the *Grid* object account for dimensionality (1-, 2-, or 3-dimensional curves, surfaces, or volumes), and various geometry/topology portrayals, including structured, unstructured, layered, and polygonal data representations.

4. New "Merged-Model" support was added to PGL to handle cases where several geophysical models are developed for the same station/phase by two or more research organizations resulting in regions where the models overlap. This approach simply uses the new *Polygon* object, defined above, to define regions where interpolation results from each regional model are used exclusively. Transition boundaries assigned to the *Polygons* guarantee a smooth transition from one region to the next.

5. A new *StationRow* object was added that works in conjunction with the *Station* object to track the input rows provided from the NDC database site table. The information includes station on- and off-dates, position, an array flag (true implies an array while false implies a stand-alone station), a reference station vector of strings, a description, a north/east component vector, and load date attributes. All attributes are accessible using corresponding "get" functions. This object was added to properly track station information in a manner consistent with the NDC database site-table. Recently, errors inherent in the site table have been a cause for concern as algorithms increasingly use the site table data, and require logical consistency from their representations in order to behave as intended.

6. We provided significant enhancements to the locator/PGL interface to improve its capability to assemble ad hoc geophysical models on the fly and to minimize the pre-input interaction required by users connected to the interface through location algorithms. These enhancements and features include the following (1) a new optional model configuration file that replaces the required master file — this file uses a simple arbitrarily formatted keyword approach to easily and effectively define the input data, (2) an automatic lookup of path corrected models within the FDB without the need for their specification in the model configuration file (i.e., no master file), (3) an automatic generation of path corrected models from array siblings that already possess a path corrected definition — if a path corrected model is not found for a client requested station/phase then the PGL locator interface will attempt to find a related array station (another element or the array itself) for which a defined path correction exists in the database. If one is found a new model is constructed on the fly that uses the requested station (for which no model was found) and the discovered related model path correction components to construct a new model, which can be used for the requested station/phase. If more than one related model is discovered, the one closest to the requested station is used to construct the new model, (4) if the first two attempts (described in steps 2 and 3 above) fail to find a path corrected model then a standard default base model for the requested station/phase will be constructed, and (5) a new log file (pglli.log) is automatically generated in the execution directory that contains information about the input file parse, and about all models that are (a) loaded from the database, (b) constructed from a related sister array that has an FDB definition, or (c) constructed as a standard base model definition.. All issued alerts, warnings, and errors are output to the log file.

7. Finally, The PGL code was successfully migrated from a Unix/Sun Solaris port to a Microsoft Windows VC++ 7.0 .Net platform. This migration was accomplished to partially satisfy some of our customers needs to run the GNEM software on platforms other than a Sun Solaris. In the future, ports to MAC OS X and LINUX will be migrated.

Data Manager Tool

The Data Manager tool (DM) provides access to the PG data for purposes of managing the organization of the generated PG database, and for perusing the data for visualization and information purposes. The tool is written in Java providing a GUI front end and a Java native interface into the PGL. The low-level visualization support component is primarily handled by the visualization toolkit library, which is described below.

The database management facilities of the data manager include a means for inspecting the entire object reference and dependency hierarchy for any object in multiple databases simultaneously. These facilities provide mechanisms to copy or move objects and their dependents from one database to another or to simply remove them from their containing database entirely. These services allow a user to construct new databases from existing databases that use a subset of the objects contained in any of the existing databases. This functionality provides a means for users to build databases that are tailored for spatially interesting regions or that contain objects that are specific for certain types of problems (e.g., seismic location). Figure 3 below illustrates a typical hierarchical dependency view of a database including object meta data key listings and their dependents.

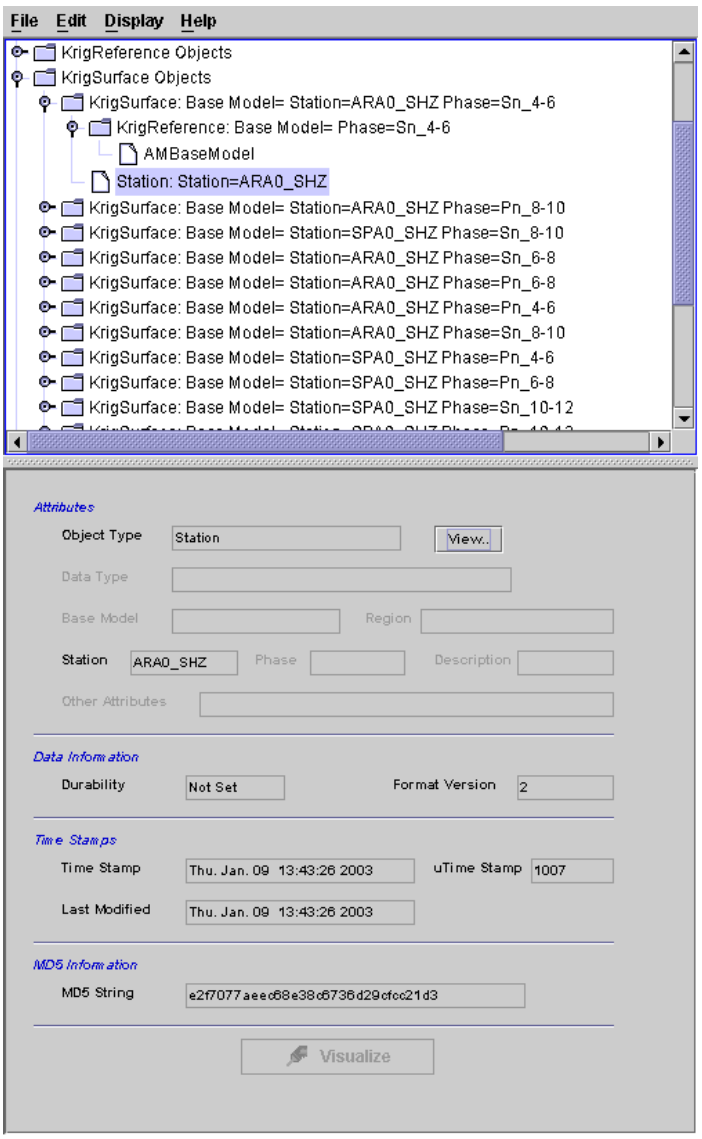

Figure 3. A typical screen shot of the DM GUI displaying a hierarchical view of the database objects and their dependents.

Aside from standard overhead management facilities, the DM was improved this year with the addition of the following features.

1. DM was updated to support PG *Station*, station row, and station data classes. These modifications allow for the creation of *Station* objects that more closely resemble stations in the NNSA schema site table. A user can import dumps from the site table (using a SQL script provided with DM to customize the format of the file) that now includes all fields in the site table and not just the positional information that provided for in the previous years. The import function contains complex error checking that attempts to resolve outstanding discrepancies in the site table data, such as incorrect or differing spatial and temporal positional information or invalid use of reference stations. Few assumptions are made about the data as they are read, however, multiple detailed messages are output regarding the state of the data as they are processed and fed into an FDB.
2. The station editor table was corrected and expanded to utilize the underlying PGL station classes and the error checking described above, and to improve visualization of the data and editing capabilities of the table itself. Stations may be input individually, rows added or deleted, and information altered, given proper permissions and object reference limitations.
3. A small class was added to employ some of the functionality of the PGL FDB_Query command line tool in the DM. The functions of this class allow the user to select a database, or an object within that database, and return information about it, which is dumped to a separate text pane. This capability is part of an ongoing effort to consolidate PG object management tasks into DM.

Besides these new input/output and object information enhancements, several global improvements involving the DM visualization system are also underway. Some of these major improvements include the following.

1. The low-level visualization library was converted to the VTK. This conversion involved significant modifications to the underlying DM interface to eliminate old functionality and replace it with the new calls to the VTK application programmer interface.
2. The old geophysical volume viewer was converted to utilize the new VTK interface, preserving most of the original functionality. In the future, new capabilities will be added that access features found in VTK that were not available in the old visualization system.
3. We provided the beginnings of a general-purpose visualization system that can depict any objects found in a PGL FDB. The new interface currently visualizes all PGL model and base-model objects. In the future polygon, tessellation, and parametric grid objects will be added.
4. An ESRI shape file to VTK data file translator was created so that the many ESRI shape files used by the Air Force Technical Applications Center (AFTAC) can easily be converted to the new VTK format for subsequent visualization. The shape file objects are used as contextual backdrops when visualizing PGL data objects.

Figures 4 and 5 illustrate an example VTK screen shot of the geophysical model and the base-model component of a travel-time model, respectively.

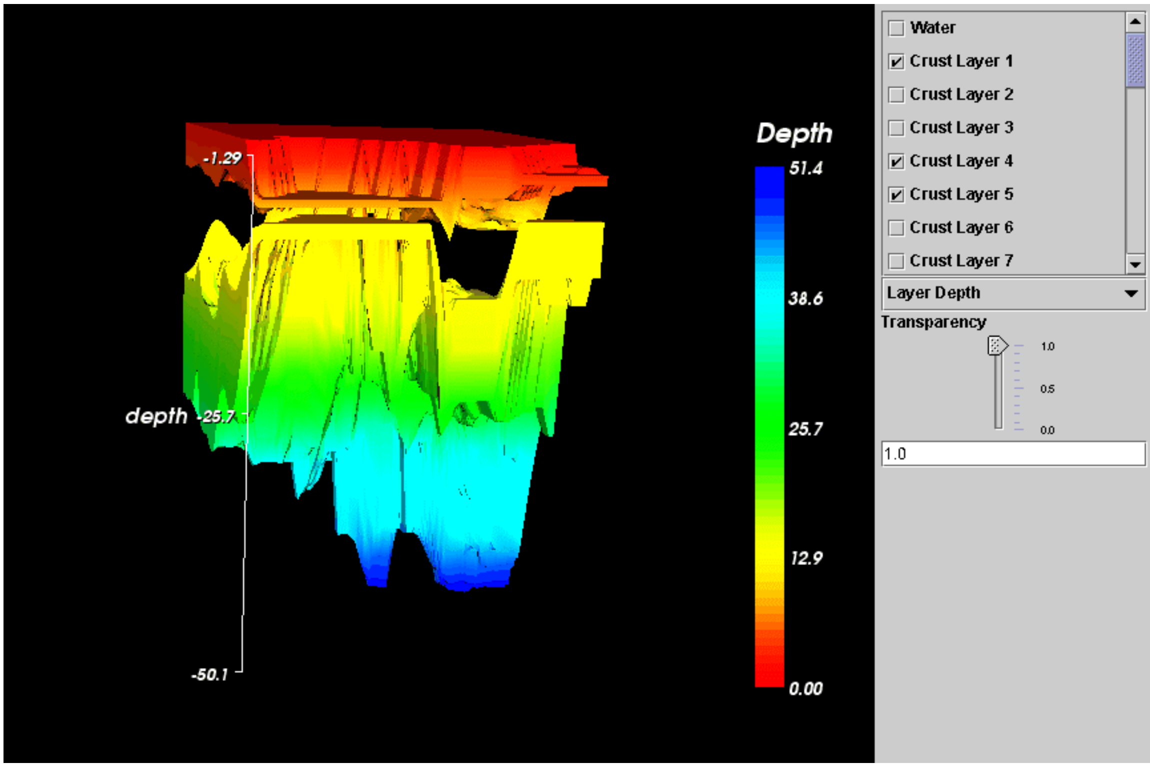

Figure 4. A typical screen shot of the DM geophysical model view.

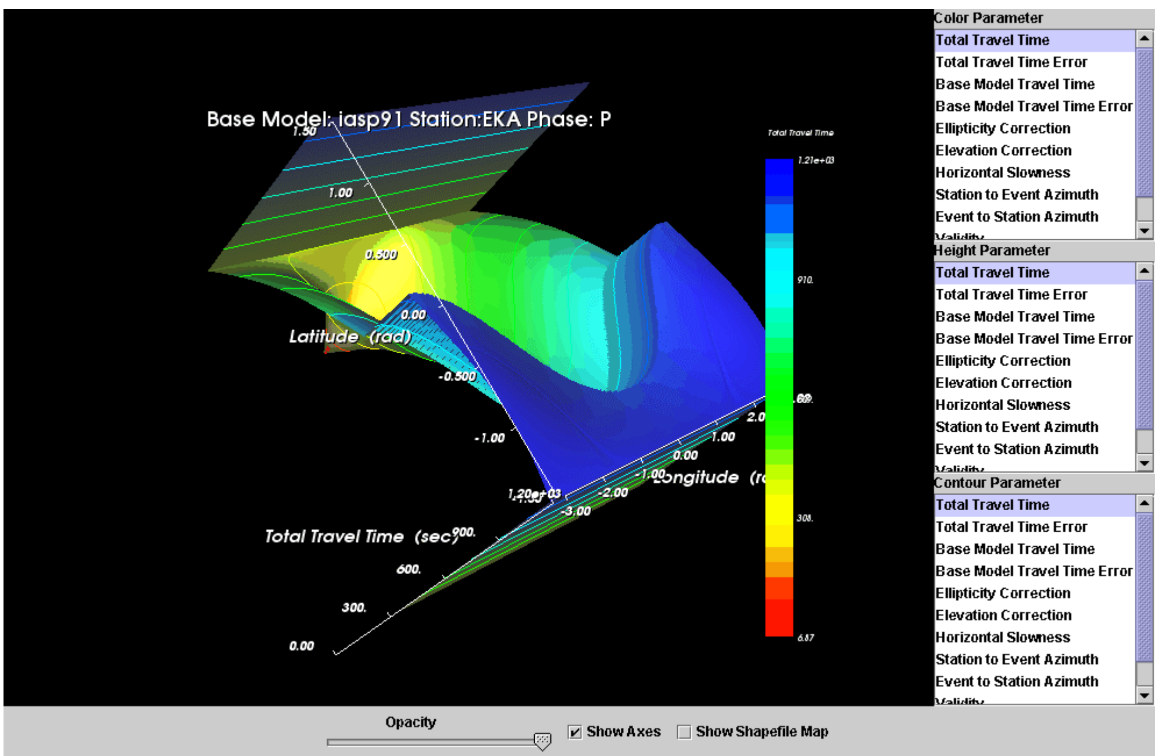

Figure 5. A typical screen shot of the DM travel-time model view displaying the base model component.

Seismic Event Location Tool

The Seismic Event Location Tool (LocOO) is a seismic event location code that computes event locations and associated uncertainty based on travel time, azimuth, and horizontal slowness observations (Ballard, 2002). LocOO was written using an object-oriented approach. It is fundamentally an infrastructure for a seismic event locator. This code relies on four modules to actually implement seismic event location, (1) a data input module, (2) a seismic observation prediction module, (3) a location algorithm module, and (4) a results output module. Currently, the only location algorithm implemented is a linearized least-squares inversion algorithm (Geiger, 1910; Bratt and Bache, 1988), enhanced with Levenberg-Marquardt damping (Levenberg, 1944; Marquardt, 1963) to improve the performance in highly nonlinear regions of model space. The location algorithm also computes multidimensional uncertainty estimates regarding the best-fit event location (Jordan and Sverdrup, 1981), including non-linear uncertainty bounds that are not necessarily elliptical in shape (Wilcock and Toomey, 1971; Ballard, 2003). The only prediction module currently implemented is the Parametric Grid Library (Hipp et al., 1999). LocOO was developed in parallel with PGL and continues to keep up with all of its new features and capabilities, including utilization of complex data models and elimination of master files. In short, any capability that is implemented in PGL, and that can affect seismic event location, can be accessed with LocOO.

The most important improvement in LocOO version 2.0, released in July 2004, is that the architecture of the code was redesigned to allow simultaneous, multiple event location algorithms to be implemented. To date, no such algorithms have been incorporated into LocOO, but the internal data structures in the code are now in place to handle such algorithms and we expect to add one more multiple event location algorithm in the coming year. In addition, the internal structures of the code were streamlined to simplify future incorporation of new location algorithms.

Another improvement in the code was the replacement of the NNSA schema database access modules used in version 1 with two new database access modules. The first is implemented on top of the new OCCI features available in Oracle 9i, and the second uses DBToolBox functions (Ballard and Lewis, 2004) for input and output to database flat-files and XML data sources.

While these features are not readily apparent to the everyday users of LocOO, they should significantly enhance the stability and robustness of the code, and facilitate future modifications and enhancements.

CONCLUSIONS AND RECOMMENDATIONS

In this paper, we have described the enhancements, made in 2004, to the integrated set of NNSA software known as PGSS. Parametric Grid Software Suite tools were developed to take advantage of the improvements, in the ongoing development, of the GNEM R&E earth model, and the tools' uses span data population and storage, data management and viewing, and seismic event location.

Future versions of the PGSS will be developed to support new enhancements to the GNEM R&E earth model. Some of the planned improvements are as follows.

1. Continued transition of the PGL mid-level geometry representation toward the new *DataModel* paradigm described above.
2. Additional DM GUI improvements include the following:
   - Enforcement of the correct tree structure text and alphabetization of entries.
   - Layout and content restructuring of the attribute panel to hold table displays of object attributes.
   - Additional right-click menu items updated to access commonly used functions, such as searching, visualization, etc.
   - Expanded error/alert/warning notifications using enhanced font and color-coding techniques.
   - Incorporation of a separate tree node pane displaying navigable FDB object references and dependencies for the selected object.
   - Provisions for generic FDB object creation functionality that will allow a user to define any FDB supported object as a keyword based text input file. This capability will provide another population input path when there are no tools able to provide a specific required flavor of a requested object.

- Capability to support point interpolation in the GUI allowing a user to enter a set of geographic interpolation coordinates to perform object interpolation and subsequently output the results to a table.

3. Integration of selected DM functionality directly into the Sandia National Laboratories KB Navigator tool.
4. Addition of one or more multiple event location algorithms to LocOO.

REFERENCES

Ballard, S. (2002), Seismic Event Location Using Levenberg-Marquardt Least Squares Inversion, Sandia National Laboratories, SAND2002-3083.

Ballard, S. (2003), "Seismic Event Location: Dealing with Multi-dimensional Uncertainty, Model Non-Linearity and Local Minima", in *Proceedings of the 25th Seismic Research Review – Nuclear Explosion Monitoring: Building the Knowledge Base*, LA-UR-03-6029, Vol. 3, pp. 193-202.

Ballard, S. and J. E. Lewis (2004), "DBTools: A Suite of Tools for Manipulating Information in a Relational Database", in *Proceedings of the 26th Seismic Research Review: Trends in Nuclear Explosion Monitoring*, Orlando, FL (Sept. 21-23, 2004).

Bassin, C., Laske, G. and Masters, G., (2000), The Current Limits of Resolution for Surface Wave Tomography in North America, EOS Trans AGU, 81, F897, 2000.

Bratt, S. R. and T. C. Bache (1988), Locating Events with a Sparse Network of Regional Arrays, *Bull. Seis. Soc. Am.*, v. 78, pp. 780-798.

Geiger, L. (1910), Herdbestimmung bei erdbeden ans den ankunftzeiten, K. Gessel. Wiss. Goett. v. 4, pp. 331-349.

Hipp, J.R., Young, C. J., Moore, S. G., Shepherd E. R., Schultz C. A, Myers S. C., (1999), Parametric Grid Information in the DOE Knowledge Base: Data Preparation, Storage, and Access, Sandia National Laboratories, SAND99-2277.

Hipp, J.R., Simons, R.W., Jensen, L.A., (2002), "The GNEM R&E Parametric Grid Data Software Suite: Tools for Creation, Access, Management, Viewing, and Export", in *Proceedings of the 24th Seismic Research Review - Nuclear Explosion Monitoring: Innovation and Integration*, LA-UR-02-5048, Vol. 1, pp. 283-291.

Jordan, T. H. and K. A. Sverdrup (1981), Teleseismic Location Techniques and their Application to Earthquake Clusters in the South-Central Pacific, Bull. Seis. Soc. Am., v. 71, pp. 1105-1130.

Levenberg, K. (1944), A method for the solution of certain non-linear problems in least squares, *Quart. Appl. Math.*, v. 2, pp. 164-168.

Marquardt, D. W. (1963), Journal of the Society for Industrial and Applied Mathematics, v. 11, pp. 431-441.

Schroeder, W., Martin, K., Lorensen B., (2003), The Visualization Toolkit, 3rd Edition, Kitware, Inc., February 2003, ISBN: 1930934076

Wilcock, W. S. D. and D. R. Toomey (1971), Estimating Hypocentral Uncertainties for Marine Microearthquake Surveys: A Comparison of generalized inverse and grid search methods, Marine Geophysical Researches, v. 13, 161-171.