PAPER E

TOMOGRAPHIC IMAGING AND MODELING SOFTWARE

Gary Mavko, Mike Fitzpatrick, Caroline Lambert, Pierre Samec, Jerry M. Harris

Seismic Tomography Project

SUMMARY

During the last few years we have been developing software for displaying, modeling, processing, inverting, and interpreting crosswell tomographic data. Part of this software, which we call the TIMS (Tomographic Imaging and Modeling Software) System, is directly aimed at providing the tools needed to process raw crosswell seismic data into interpreted tomograms.

An important part of our philosophy has been to keep the TIMS software simple and portable and to rely on features supplied by widely available standard software including FORTRAN, C, LINPACK, and the UNIX shell. Data is stored in a machine-independent and network compatible form called netCDF. We have attempted to keep numerical algorithms relatively separate from the I/O, recognizing that in the future we may wish to use these algorithms in other environments such as Landmark Openworks, which has been heavily developed at Stanford by Pierre Samec. A working prototype TIMS system exists and has been used successfully to process a real field data set. Additional work over the next few months will focus on standardizing a number of stand alone programs that have been developed for specialized plotting, modeling, and inversion experiments, as well as completing documentation.

The TIMS development was largely sponsored by the Gas Research Institute. Additional work, including much of the forward modeling and integration work done by Samec, and the inversion and image enhancement studies by Harris, Lazaratos, Mavko, and Michelena also received support from other projects, including the SRB (Stanford Rock and Borehole Project). Some of the research in these areas is reported elsewhere in this volume and in the SRB volume. In this note we describe primarily the TIMS system.

INTRODUCTION

TIMS was developed to address the growing need both in industry and academia to be able to handle, display, process, and interpret single well and crosswell seismic data. Many of the needs are the same as in surface seismic, and well known techniques for signal processing like spectral analysis, filtering, gaining, and agc are immediately applicable. However, there are enough differences -- including trivial ones like the

source and receiver geometry and fundamental ones like the emphasis on direct arrivals instead of reflected arrivals -- that a new system is desirable.

When we started work on TIMS in the Spring of 1989 we already had crosswell data in house, so we had an immediate need for working software. At the same time we recognized the importance of planning the software for future growth, consistency, and flexibility. We have tried to design an approach that allows tools to be added quickly and that still will evolve gracefully as our needs and the available technology change. For example, the algorithms are modularized so that in the future they can be used with different I/O routines and/or with a different database strategy. A prototype system is now working that we have used successfully to process, display, and invert a real data set.

FEATURES OF THE TIMS SYTEM

The System has both **interactive** and **batch** capability (See Figure 1). **Interactive** capabilities are desirable for quick and convenient processing and display of small pieces of data. As with surface seismic processing, there is no single sequence that works well on all data sets. Testing for optimum filtering, gaining, noise rejection, and so on, is almost always necessary, and the testing is most effective when done interactively. For travel time analysis, there is no substitute for interactive picking and pick editing. **Batch** capabilities allow subsequent processing of very large pieces of data, and it is most convenient when the batch programs apply the identical algorithms that are used in the testing. At present batch processing streams are constructed using UNIX shell scripts to string together the various applications modules. The System allows the various processes to extract subsets of the data in a file or to default to reading and processing all of the data in a file.

The System is able to handle **heterogeneous data** sets. Although we expect source and receivers to be nominally equally spaced, they won't be so in practice. The research nature of crosswell seismic surveys naturally results in variable spatial coverage, variable trace lengths, and variable source types. Three component data are also likely to be common. For the seismic traces, sample rate is about the only thing we assume to be fixed for any given data set. Well logs, travel time picks, velocity models, and tomographic images will also be necessary parts of the data set.

A critical aspect of modeling, interpretation, data analysis, and processing is the data formatting and i/o. A typical situation in industry and academia is that networking hardware and software are in place to allow free access to data among various researchers, but differences in data format required by various analysis, display and processing tools end up hampering the exchange. In order to reduce this handicap early on, we (Pierre Samec) undertook a study of data handling strategies.

After considering several alternatives we adopted a public domain data access interface which was developed under the sponsorship of the National Science Foundation (see Figure 2). This purpose of this interface, called the Network Common Data Form (**netCDF**), is to allow one to create, access, and share scientific data in a form that is self-describing and network-transparent. "Self-describing" means that a file includes information defining the data it contains. "Network-transparent" means that a file is

represented in a form that can be accessed by computers with different ways of storing integers, characters, and floating-point numbers. Using the netCDF interface in new software for scientific data access, management, analysis, and display can improve the reusability of the software for other data sets and by other users. The netCDF software provides common C and FORTRAN interfaces for applications and data, and it has been tested on both UNIX and VMS operating systems, so it appears to be highly portable. netCDF is an interface to a library of data access subroutines for storing and retrieving data. In simple terms, to a programmer, netCDF looks like a set of working subroutines somewhat analogous to the the FORTRAN statements OPEN, READ, and WRITE, but of course more powerful.

Another issue is portability. Since we have affiliations with a broad group of potential users and contributors to the System, and because we have a wide variety of hardware to work with, we have attempted to maintain wide portability to the extent that we do not seriously impact performance. To accomplish this, the system is based on:

- UNIX
- X-windows
- netCDF
- FORTRAN and C

We have attempted to keep the software modular, so that the basic numerical algorithms are independent of the input, output, and trace handling. Similarly many of the primitives called by the numerical algorithms are public domain standards such as LINPACK. These can always be optimized for speed on a particular machine by vectorizing, parallelizing, or coding in machine language; at the same time reliable FORTRAN equivalents will be immediately available for any new machine. The biggest portability flaw that we struggle with is the variability in the way that FORTRAN programs call C subroutines, and vice versa, and machine-dependent requirements when FORTRAN and C are linked together with I/O routines.

We have also tried to keep the software development simple. To the extent possible we have relied on existing features of UNIX, the netCDF common data form, and the C programming language, rather than to duplicate them with a complex central program. For example, C allows for dynamic memory allocation and flexible input options via command line parsing. UNIX shell scripts allow varied job streams to be constructed from the same stand-alone programs used for testing. The flexible machine-independent storage format of netCDF coupled with UNIX Network File System (NFS) allow networked computer resources to be efficiently shared.

DESCRIPTION OF AVAILABLE SOFTWARE

TIMS will include (1) **software** and (2) **documentation**. The software, which is described below, will include high level applications programs, low level utilities, graphics tools, and example shell programs for creating and running batch jobs. We envision that at some point in the future, we may evolve toward a main driver routine in place of or in addition to the shell programs if i/o performance requires it. The documentation will include (1) user manuals explaining how to install and use the

delivered software and (2) programming manuals explaining how to write applications modules that will be compatible with the system.

SOFTWARE

There will be six major types of software associated with the System:

- **Applications modules**
- **Picker**
- **Plotting software**
- **Utility software**
- **netCDF i/o software**
- **shell utilities**

Applications modules. Applications modules perform the generically recognized, high level seismic processing tasks. These are what the user of the system will see and use. There are two broad categories: Trace processing modules that perform signal processing tasks very much like those in surface reflection seismology, and post-trace modules that create and process travel time picks, perform ray tracing, and do inversions. The trace processing applications include, for example, correlation, agc, deconvolution, dip-filter, Radon transform, up and down going wave separation, corridor stack, migration, and trace plotting. The post-trace applications include first break picking, ray tracing, 2-D inversion, and tomogram and log plotting. All of the trace processing applications will be oriented toward processing a "slice" of data: a source gather, receiver gather, a common offset panel or a time slice. Some will operate on only one trace at a time within the slice, so that the order of data going in does not matter. Examples of these modules are bandpass filtering, agc, deconvolution. Other modules will operate on entire slices at a time. Examples of these are fk-filtering, fk-spectral analysis, and multichannel deconvolution. Most commercial surface seismic processing packages are trace oriented and require special care to "collect" traces into slices for multichannel processing. Our approach has tremendously simplified the process of creating new applications.

Picker. The picker is an interactive program that allows the user to display crosswell seismic data on the video screen, identify wave arrival events, and 'pick' the arrival times by use of the mouse. The user can pick the time of each event on each trace, one at a time, or can instruct the program to interpolate or extrapolate the user's picks automatically, under the user's guidance.

Plotting Software includes tools for plotting on various devices including terminals, laser printers, and versatec color plotters. A variety of data types can be displayed including seismic traces, rays, travel time picks, well logs, and inverted velocity tomograms.

Utilities. Utility software includes a wide range of low level routines for handling and manipulating data within applications and plotting modules. We sometimes refer to these also as primitives. The user of the System will generally not be aware of utilities, but a programmer will use them routinely. The utility routines are intended to be

useful for commonly repeated tasks and will generally not be unique to any one application module. Examples of utility software include:

- **i/o routines.** These are subroutines to read and write the commonly used subsets of data, like source gathers, receiver gathers, and time slices, as well as more general data like scalars and 1-D, 2-D, and 3-D arrays.
- **vector routines.** Seismic processing is highly vector-oriented. We are building a set of vector routines using for example LINPACK. These are written in FORTRAN, and when compiled can take advantage of vector and parallel operations, if available.
- **signal processing primitives.** These include commonly used functions like convolution, fft, filter design, correlation.
- **plotting primitives.** These are the commonly used inner routines of the plotting software listed above.

netCDF software is the basis of our i/o. netCDF is a public domain item. Although we will not 'deliver' it, we will supply information on obtaining it, on using it, and on interfacing it with our System.

shell utilities. These will include example scripts for creating batch jobs as well as tools for creating and compiling new applications modules.

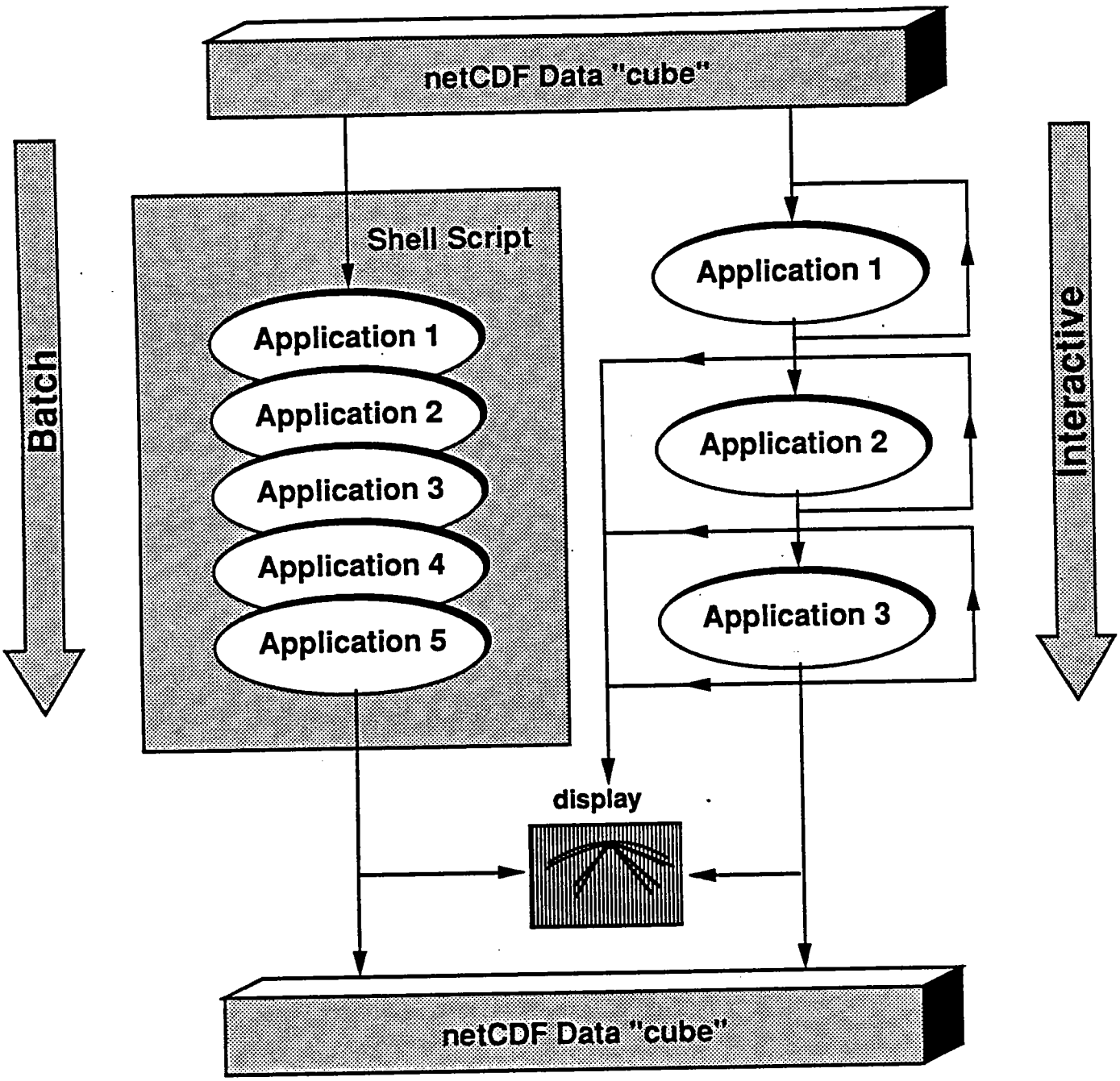

Figure 1. The TIMS system consists of a set of applications modules which read data from the netCDF file and deposit results back in the netCDF format. Applications can be called individually in an interactive mode, or combined using a UNIX shell script to create a batch processing stream.

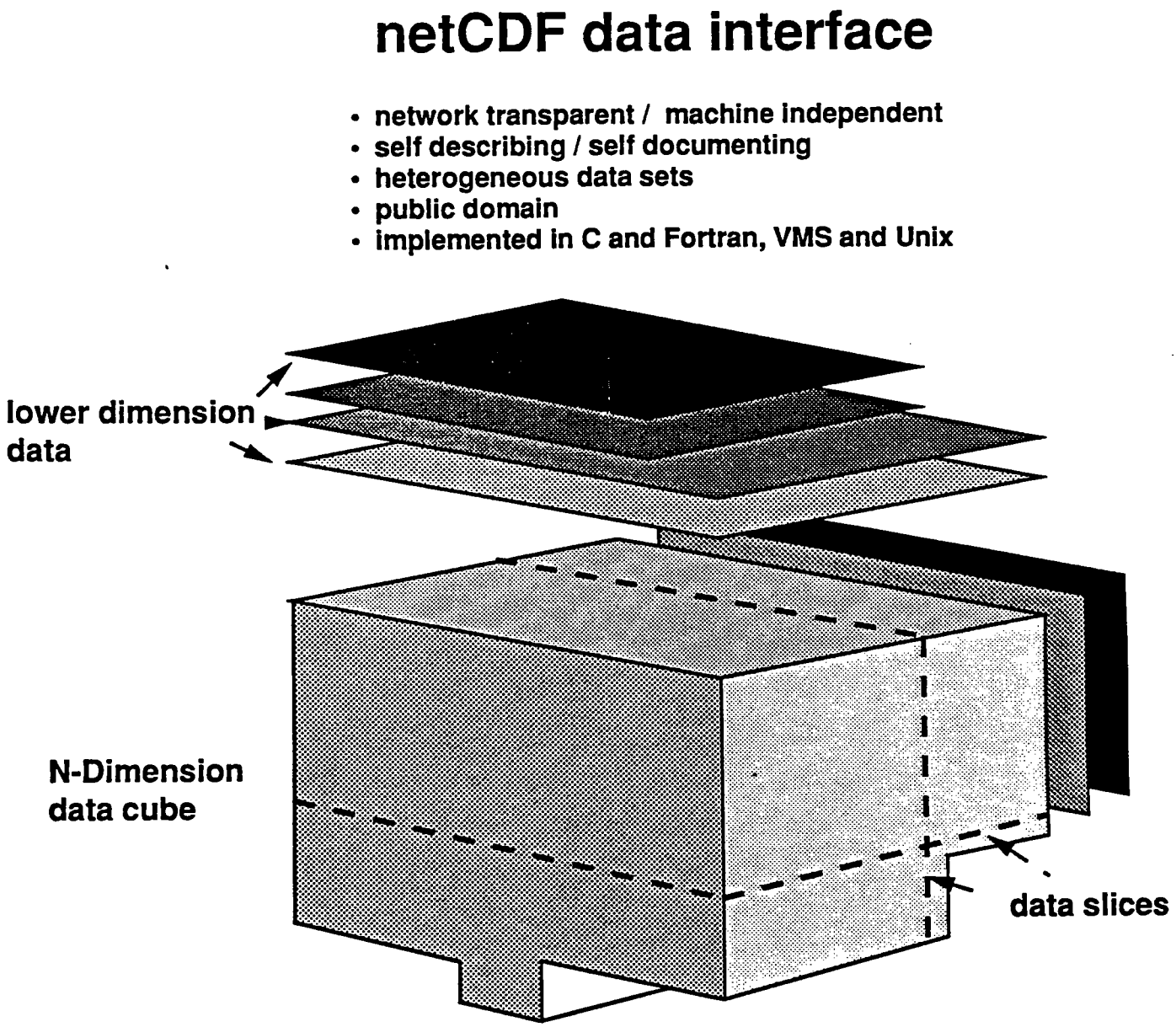

Figure 2. The netCDF common data form allows flexible, self-documenting, machine independent storage of heterogeneous data.