Hyercube viewer: New displays and new data-types

Robert G. Clapp and Nelson Nagales

ABSTRACT

No single way to view seismic data is effective in all cases. Rather than building separate tools for each viewing approach, we added functionality to SEP's existing hypercube viewing tool. In addition to other functionality improvements, we added the capability to view wiggle traces, contours, out-of-core datasets, and datasets with different number of dimensions and size.

INTRODUCTION

Seismic viewing tools can be broken into two categories: batch viewing programs and interactive viewers. SEP has a long history of both program types. Static viewing programs can produce line graphs, wiggle traces, hidden line plots, contours, and raster images. These programs, in combination with utility programs that allow windowing and transposing of arrays offer the ability to create effective static graphics. For viewing and understanding multi-dimensional volumes, these tools are not as useful.

A series of SEP's interactive viewers (Ottolini, 1982, 1983, 1988, 1990; Clapp, 2001; Chen and Clapp, 2006) have also been developed at SEP. These viewers are more effective in viewing multi-dimensional volumes but have been limited in three key ways. The first shortcoming is their inability to produce high quality graphics and reproduce a given view of a dataset. Second, they were restricted to viewing a single (later a few) datasets that had to be identical in size and fit in memory. Finally, they only display data in raster format. Clapp et al. (2008) made an initial attempt in addressing the first shortcoming by recording, and allowing, replaying of mouse and keyboard actions.

In this paper we attempt to address the viewer's dataset and display limitations. We describe changes to the way the program interacts with datasets. These changes allow the viewer to handle datasets with different sampling and different number of dimensions, and even operate in an out-of-core mode. We show the new display options that allow the user to view the data as contour and wiggle plots. Finally we describe additional changes in the UI to the viewer and briefly discuss what might be added in the future.

DISPLAY OPTIONS

There are several possible ways to plot regular fields. SEP has generally used raster plots to display data. These plots have the advantage of being able to display relatively dense data within wide spatial frequency range. In some cases raster is not the ideal plotting option.

When examining waveforms it is often useful to display traces in a wiggle format. Wiggle format allows you to see more clearly the actual waveform recorded by the sensor. Figure 1 demonstrates hypercube's wiggle capabilities. In this case we are overlaying a velocity model on top of a migrated volume. In order to see the waveform each trace must take a few pixels of screen width, we chose to use a minimum of 8 pixels. If $8n > w$, where n is the number of traces and w is the width in pixels of the display we subsample the data volume until this criteria is met.

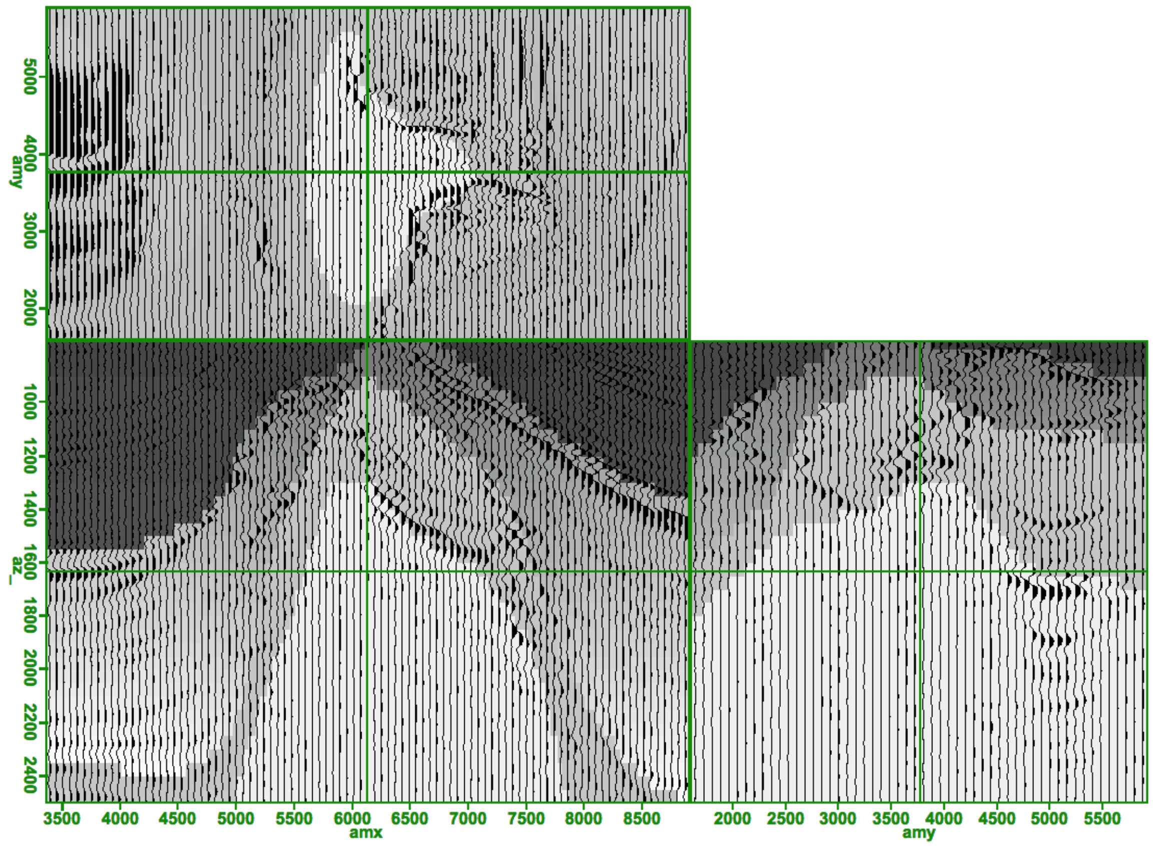

Figure 1: Velocity model overlain by a wiggle plot of a migrated volume. [**NR**]

For fields that vary smoothly it is often useful to contour the data rather than display directly each cells value. Figure 2 demonstrates this concept by drawing velocity contours on top of migrated volume. The program attempts to separate contour labels by some distance. Note the jagged nature of the contour lines. This is due to storing the velocity data as a series of bytes rather floats.

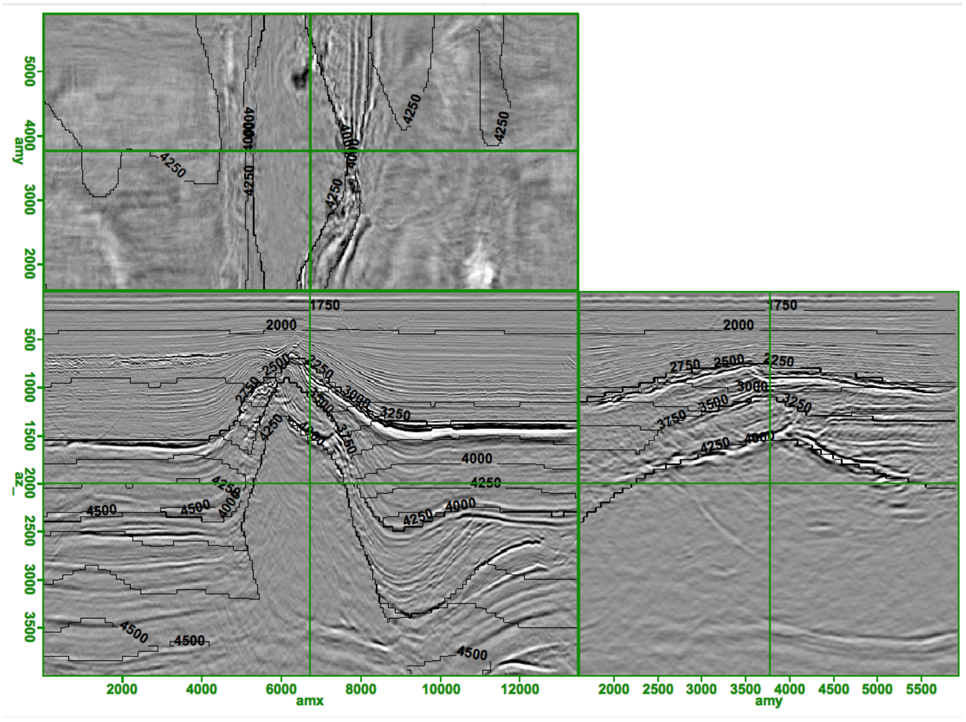

Figure 2: Migrated volume overlain by contours. Note the jagged nature of the contours due to the velocity volume being stored as bytes. [**CR**]

DATASET DEFINITION

How to define a dataset poses a challenge and is where using an object language proves most beneficial. The hypercube viewer uses a grid concept similar to SEPlib and SEP3D. The user, either directly, or indirectly, describes an n-dimensional grid in which the viewer operates. Each dataset is then described in terms of this overlying grid. Each dataset must fit within this grid description. A `io_func` class is responsible for reading the dataset from disk or creating the dataset. A `buffer` class is used to store the data, and the `dataset` class is responsible for handling requests from other portions of the program.

IO

For conventional datasets the `io_func` class is responsible for reading from disk a subsection of the controlling grid. It first converts from the grid coordinate system to the local dataset's coordinate system. This amounts to honoring the range of each axis requested. For example, imagine the grid is defined by the table below.

Axis	**n**	**o**	**d**
1	100	0	1
2	100	0	1
3	100	0	1

And the dataset is only 2-D (an example is a grid that define time, offset, and midpoint while the dataset contains only time and midpoint) with the following sampling.

Axis	**n**	**o**	**d**
1	50	0	.2
2	1	0	1
3	80	0	1

If there is a read request for the entire volume it will be converted into a read request for the 2-D subsection of the grid in which the dataset exists. Currently the `io_func` module can read SEPlib, RSF, SU, SEG-Y, and SeisPak formats from disk in this manner.

The `io_func` does not necessarily have to read from disk, and the data isn't necessarily static. For example imagine doing interactive NMO on 2-D dataset. Three datasets can exist: the original CMP, a semblance panel, and NMO corrected gather. In this case the grid contains 4 axes: time, offset, midpoint, and velocity. None of the three datasets contain all four axes. The latter two change depending user interaction. When the midpoint location changes semblance is recomputed. When a new velocity is selected on the semblance panel the NMOed data changes. To support this type of functionality the `io_func` has a `changed` boolean.

Buffer

The `buffer` class has two main responsibilities. It holds a subsection of the data and can return a 2-D slice of the subsection it holds in memory. Currently `buffer` class stores data as either a series of bytes of floats in a regular mesh, but it is not limited to this storage mechanism. The data could be stored in number of compressed formats. A dataset can be composed of several different buffers.

Dataset

A dataset is initialized by an `io_func` and creates buffers as it needs them. Currently there are four basic buffer types. The simplest is inherited from `incore_data`. As the name implies this data type is stored completely in memory and has the same number of axes as the controlling grid. The `created_data` class is for data that changes depending on the user's interaction with the program. An example of this is the semblance and NMO corrected datasets in the velocity analysis example. If a dataset does not contain all of the axes of the grid, it should be stored using the `partial_data` class.

The `outcore_data` class, as the name implies, is for datasets that are not stored completely in memory. This class reads in a 3-D subset of the domain based on the order the axes are being displayed in a given view. For example, given a 4-D volume and a view that is displaying the first, third, and fourth axis, a 3-D subset of the data will be read centered at the current position along the second axis. As long as the position is only being changed in these three dimension no further read requests are necessary. Several of these out-of-core buffers are created for each dataset allowing multiple views display multiple subsets and allowing switching back and forth between positions without having constant data read delays.

USER INTERFACE CHANGES

Several changes where made to the user interface beyond the plotting options discussed above. In this section we will highlight the most significant changes.

Mouse/status

As discussed in Clapp et al. (2008) how the hypercube viewer interacts with the mouse can be altered. The `main` tab, submenu `info` the mouse controls can be modified. Currently there are three options: zoom, navigate, and pick. The options refer to the action of the left mouse button. The effect of the middle and right button can be displayed in the status bar.

Status bar

The status bar is located at the bottom portion of the view window. It starts off displaying the current mouse controls. The `main tab`, submenu `info` changes the status bar to display either the current position, in grid coordinates, or the actual sample value that has been selected.

Overlay/display options

The `view` tab, submenu `data` has been modified so that both the primary displayed data and the data (if any) you wish to overlay can be changed easily.

How you wish to display the primary and overlaid data can be modified in the `Display` tab, `general` submenu. For each dataset you can choose to display using various color maps, in wiggle format, or with contours. The `Display` tab contains two additional submenus, `Wiggle` and `Contour`. The `Wiggle` submenu allows you to change the line color and whether, and what color, to fill positive and negative portion of the wiggle trace. The `Contour` submenu allows you to set the number of contours, the initial contour value, and the contour interval.

FUTURE DIRECTIONS

There are three active areas of development for the viewer. The first is the ability to partition space in N-D. This ability will enable the second area, interactive processing. Many processes need human guidance but that guidance often is more complex then modifying a single 1-D line. The final area is the ability to store additional data types. The ability to store data in a compressed form, such as curvelets, would enable very large volumes to be manipulated effectively.

CONCLUSION

We expanded SEP's hypercube viewer by addressing two of its weaknesses. First we add the ability to view contour and wiggle plots. Second we added the ability to view multiple datasets which exist in the same space but might not have the sampling, the same number of dimensions, or even be possible to completely hold in memory.

REFERENCES

Chen, D. M. and R. G. Clapp, 2006, Data-fusion of volumes, visualization of paths, and revision of viewing sequences in ricksep: SEP-Report, **125**.

Clapp, R., 2001, Ricksep: Interactive display of multi-dimensional data: SEP-Report, **110**, 163–172.

Clapp, R. G., D. M. Chen, and S. Luo, 2008, Hypercube viewer: 2008, **134**, 179–192.

Ottolini, R., 1982, Interactive movie machine user's documentation: SEP-Report, **32**, 183–196.

———, 1983, Movie cubes: SEP-Report, **35**, 235–240.

———, 1988, Movies on the Macintosh II: SEP-Report, **59**, 255–268.

———, 1990, Seismic movies on the XView graphics system: SEP-Report, **65**, 315.