Interactive Visualization of Complex Seismic Data and Models Using Bokeh

Article *in* Seismological Research Letters · February 2018

DOI: 10.1785/0220170132

Chengping Chai
Oak Ridge National Laboratory

Robert Herrmann
Saint Louis University

ELECTRONIC SEISMOLOGIST

Interactive Visualization of Complex Seismic Data and Models Using Bokeh

by Chengping Chai, Charles J. Ammon, Monica Maceira, and Robert B. Herrmann

ABSTRACT

Visualizing multidimensional data and models becomes more challenging as the volume and resolution of seismic data and models increase. But thanks to the development of powerful and accessible computer systems, a model web browser can be used to visualize complex scientific data and models dynamically. We present four examples of seismic model visualization using an open-source Python package Bokeh. One example is a visualization of a surface-wave dispersion data set, another presents a view of three-component seismograms, and two illustrate methods to explore a 3D seismic-velocity model. Unlike other 3D visualization packages, our visualization approach has a minimum requirement on users and is relatively easy to develop, provided you have reasonable programming skills. Utilizing familiar web browsing interfaces, the dynamic tools provide us an effective and efficient approach to explore large data sets and models.

Electronic Supplement: Interactive HyperText Markup Language (HTML) dispersion, model, cross-section, and waveform viewers.

INTRODUCTION

Multidimensional data and models are common in seismic analysis, but visualizing and interacting with them can be challenging. As seismic networks and temporary field deployments continue to grow (broadband and dense short-period networks), global seismogram archives continue to expand. Improved data lead to better and more detailed models of Earth's interior and earthquakes. Three-dimensional (3D) seismic-velocity models are common, archived and shared by the community (e.g., The Earth Model Collaboration; Incorporated Research Institutions for Seismology Data Management Center [IRIS–DMC], 2011). But 3D refers only to the spatial dimensions; each 3D position in the model usually includes information on *P*- and *S*-wavespeed, density, and often attenuation. The models are constructed using multiple dimensional data sets that, among other observations, may include station and earthquake locations and travel time or dispersion properties between them, seismograms recorded between many event-station pairs, surface gravity, or magnetics, etc. In each analysis, the relationship between observations and models is not simple, and tools that allow both the model builders and the model users to explore the connections between data and model efficiently and effectively are rare. Geophysical inversions are driven by composite misfit, model roughness (smoothing), and constraints, but resolution is related to detailed relationships between the observations and the model (e.g., Menke, 2012). Summary measures of resolution provide some useful information, but they seldom tell the whole story or satisfy those with experience, who often want to see what feature in the data produced a particular feature in the model. Through years of effort and experience, seismologists have developed methods to explore the models and to examine data fits. Modern open-source software packages provide an avenue to link model and data using established (or innovative) displays that enable researchers to explore models and develop a comfort with the model–data relationships. In the following, we describe our efforts to create a useful research tool to allow us to explore a 3D seismic model and the data (tomographically derived surface-wave dispersion and multibandwidth multi-ray-parameter *P*-wave receiver functions) that constrained the model. The result does not take us all the way back to the raw observations but provides an efficient way to explore a relatively large 3D model and immediately assess some of the features in the model for significance.

Visualizing multidimensional models is a challenge. Depth slices and cross sections are commonly used to show speed variations in 2D slices by representing seismic velocity or velocity perturbations with colors. Slices and cross sections are an effective way of communicating broad patterns in parameters such as wavespeed. To communicate details of rapid velocity changes, seismic-velocity models are often displayed using 1D line-plot profiles (usually with depth). However, both model slices and 1D line-plot profiles have drawbacks. Velocity line-plot profiles can only be used to compare a limited number of profiles at once, so they do little to communicate lateral trends within the models. Color slices are not a good way to present absolute parameter (e.g., velocity) values. Different preferences of color palette and our limited ability to recognize and or distinguish colors hinder the reading of absolute velocities from

 doi: 10.1785/0220170132

color-based plots. Exploring other high-dimensional data shares similar problems. For example, surface-wave dispersion models are 4D data sets with two dimensions related to spatial coordinates, one for period (or frequency) and another dimension for group- or phase-velocity values. Based on the purpose of research, either dispersion maps or dispersion curves may be more suitable for exploring or communicating variations in dispersion data.

Great effort has been expended developing flexible complex 3D visualization tools that are often valuable ways to explore broad features in higher-dimensional data and models. These tools are not always exactly what is needed and often have substantial learning curves and require some coding to import the data into the package (e.g., Pavlis *et al.*, 2012). Our approach has been to exploit the community experience and to use interactive, linked, and classic seismic displays linking complementary views of both the data and the model. Our approach is not limited to 3D visualization. We also developed tools to visualize 1D velocity profiles from Markov chain Monte Carlo inversions (Chai *et al.*, 2017).

Graphics Formats for High-Dimensional Information

A choice in any visualization effort is the graphics format. A good graphics format has the requisite resolution and is efficient and broadly supported across common computing platforms. Vector graphics formats such as the Portable Document Format (PDF) and raster graphics formats such as the Joint Photographic Experts Group (JPEG) format and the Portable Network Graphics (PNG) meet these criteria for the static display of 2D or perhaps 3D information. Communicating high-dimensional information with sequences of images in these formats is often inefficient. But these formats can be used as building blocks for more efficient tools to explore higher-dimensional information. The images must be packaged in a way to have reasonable efficiency and portability across potential user computing platforms.

One problem with high-end visualization packages is the need for custom software that is often hard to learn. An ideal implementation would leverage broadly used graphics display tools with which most users have experience: a web browser. Modern web browsers are sophisticated tools that are used daily by most computer users. They are available on every platform, benefit from strong competition that promotes speed and efficiency, and also adhere to clearly specified standards. Browsers are known for their support of the HyperText Markup Language (HTML). But modern browsers also incorporate exceptionally efficient JavaScript computing engines, support formatting with efficient Cascading Style Sheets (CSS), and provide interactivity that can be intuitive and efficient. Most importantly, minimal requirements are imposed on the user when a browser is used for data visualization.

JavaScript, CSS, and HTML learning curves used to be a barrier to implementing HTML for scientific visualization, but the online community has produced powerful tools that facilitate creative data presentation. Python packages such as Bokeh ease the development of HTML-based visualization tools significantly. More and more earth scientists are using Python for data processing, thanks to powerful Python packages such as ObsPy (Beyreuther *et al.*, 2010) and the language's intrinsic flexibility. As scientific programmers become more familiar with Python, utilizing python packages to develop HTML visualization tools is well within their reach for only a modest investment of time. Browser-based interactive visualization is an effective way of exploring large data sets locally, sharing data and results between collaborators, and presenting complex or high-dimensional data online.

Visualization is part of the exploratory analysis of data and models during scientific investigation. Directly sharing insights gleaned from that exercise remains a challenge, but this is changing. Most journals have sophisticated web-browser-based interfaces for finding, reading, and distributing scientific publications. Many journals prepare an Extensible Markup Language (XML)/HTML version of published articles. Ⓔ Electronic supplements allow article authors to share with their readers more information in support of their interpretations and conclusions, software, and data. HTML-based dynamic tools can be a part of electronic supplements. Developments in scientific publishing show an encouraging trend that more dynamic content will be supported in the future.

In the following, we illustrate the advantages and limitations of browser-based visualization using four examples: an interactive display of a surface-wave dispersion model (from Herrmann *et al.*, 2016), two dynamic displays of a 3D shear-wavespeed model (from Chai *et al.*, 2015), and an interactive display of three-component seismograms from multiple stations. Our goal is to illustrate the use of browser-based tools for seismic-model visualization, not to present a general-purpose plotting tool. Our approach requires that seismic researchers leverage open-source tools to develop custom animations. Different research groups have different preferences for both data formats and visual effects. General interactive tools (West and Fouch, 2012; Bezada, 2014) and 3D-visualization examples (Pavlis *et al.*, 2012) are available, but they have more requirements on either the user or developer side. Our point is that if you have reasonable programming skills, open-source packages provide an opportunity to explore and share your data and models with anyone who can operate a web browser.

What You Need

On the user side, a modern web browser and possibly a network connection are the only requirements. The data and structure behind the visualization are archived in a single HTML file. An Internet connection is used to obtain JavaScript libraries and CSS definitions, but these can be embedded into the file to drop a network connection requirement. A recent web browser is required because earlier versions of Internet Explorer do not support JavaScript well. Our tests show that popular browsers including Chrome (v.61), Firefox (v.56), Internet Explorer (v.11), Safari (v.11), Opera (v.48), and Microsoft Edge (v.20) render our interactive tools without problem. Most users are familiar with the interface elements (mouse hovering, dragging and clicking, sliders, etc.). For the visualization creator, four open-source Python packages were used in our examples. NumPy and SciPy are used for basic calculations (and are likely familiar to scientific python users). ObsPy was used to parse seismic waveform data. The HTML-based visualizations were generated using the package Bokeh (see Data and Resources). All four packages can be easily installed on different operating systems using Python package managers such as Anaconda or directly from the source code. All of these packages are easy to learn because detailed documentation and basic examples are freely available online and are well maintained. For example, many easy-to-follow Bokeh examples can be found at Data and Resources.

In our examples, dispersion and velocity models are imported from text files. Seismic waveform data are parsed from Seismic Analysis Code (SAC) files. Unlike waveform data, the data format for both the dispersion and velocity models is not standardized but are simple. Other data formats such as NetCDF and HDF5 can also be incorporated with minimal extra effort. For those interested in the details, the text formats we used are explained in this article's code repository (available in the Ⓔ electronic supplement to this article, see Data and Resource).

Four Examples

The interface of our visualizations should be easy for seismologists to navigate because our dynamic views are the integration of two traditional seismic-display images with commonly used interactive tools. We recommend that the reader open the HTML documents at this point; we discuss static views of the visualizations. Figure 1 is a screenshot of a surface-wave dispersion viewer. On the left is a dynamic color-coded map of Rayleigh-wave group speed for a selected period. On the right is a dynamic view of a dispersion curve at one spatial location. The enlarged green dot in the dispersion map (Fig. 1c) identifies the location corresponding to the dispersion curve. The dispersion viewer links lateral variations of group velocities with group velocity as a function of period, connecting traditional dispersion curves and dispersion maps in a responsive interactive display, an excellent way to explore a dispersion model effectively and efficiently. Using the slider (Fig. 1a,f), readers can explore the dispersion model in two different perspectives. Each figure includes details on the location of the sample, the period, etc. Clicking (or tapping on portable devices) on a dot in the dispersion map (Fig. 1c) will load the dispersion curve at the selected location. Plots of the map and the curve can be exported as the PNG format using the Save tool (in Fig. 1b,g). The dispersion curve display is dynamic, and the user can pan and zoom to examine specific details. Crosshair tools (Fig. 1b,g) provide two reference lines to help read values from the displays. Using the interactive interface, readers/users can explore the dispersion model quickly and in detail.

Figure 2 is a screenshot of the second example, a visualization of a 3D shear velocity model constructed using smoothed and interpolated receiver functions, surface-wave dispersions, and gravity observations (Chai *et al.*, 2015). The model viewer combines the map view and a traditional seismic-velocity model profile, which provides a holistic view of the 3D model. The shear velocity map (Fig. 2c) shows lateral shear-wave velocity variations at a selected depth; the 1D profile (Fig. 2h) illustrates the shear velocity changes as a function of depth. Similar to the dispersion viewer, two slider bars (Fig. 2a,f) are used to select the depth of the shear velocity map and location of the velocity profile. The profile location can also be chosen by clicking on the boxes in the shear velocity map (Fig. 2c). Labels (Fig. 2c,i) and the color bar (Fig. 2e) are updated with the selection. Interactive tools (Fig. 2b,g) allow a user to save the current views, zoom, and show reference lines. In addition, we included a button (Fig. 2j) for the user to download the selected 1D shear velocity profile as a simple text or in model96 format (Herrmann, 2013). With a few clicks, readers are able to download a 1D velocity profile of interest from the 3D model.

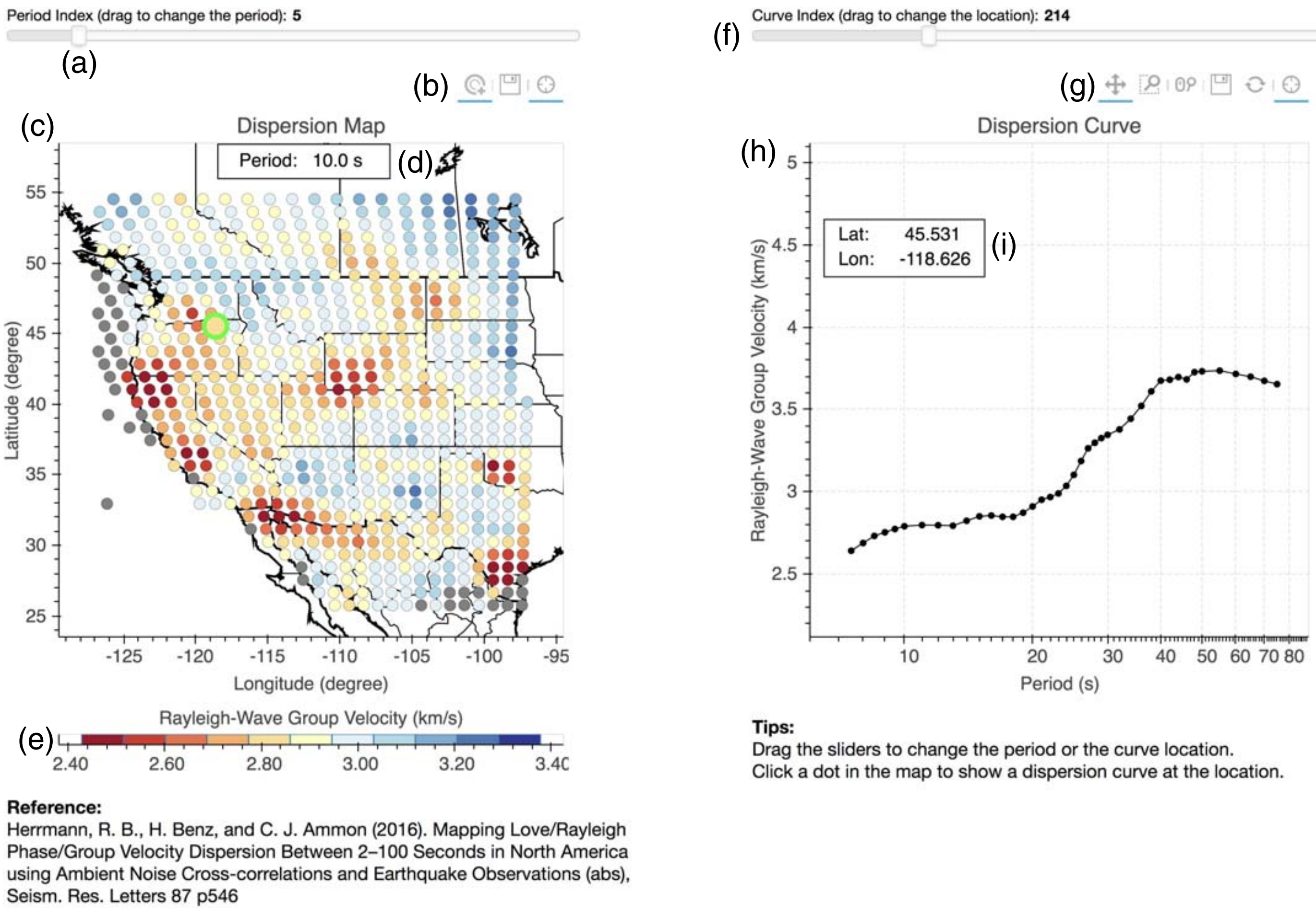
Besides the velocity profiles, cross sections are commonly used to show lateral and vertical variations in seismic-velocity structure along a linear profile. Figure 3 is a screenshot of the third example that shows a shear velocity map (Fig. 3c) and cross sections (Fig. 3g,j) side by side. Three slider bars (Fig. 3a,f,i) are provided to change the depth of the velocity map and location of the cross sections along latitude and longitude. Similar to first two examples, labels (Fig. 3d,h,k) update with the slider selection. After the selection, a user can save the selected views to disk.

For the last example, we visualize three-component seismograms from multiple stations and show station locations at the same time. A static view of the fourth example is shown in Figure 4. In Figure 4c, station locations are illustrated with triangles. The asterisk indicates the location of the corresponding seismic event. Clicking on a triangle or sliding the Station Index slider bar (Fig. 4a) will select a station. The right side of the figure (Fig. 4e,f,g) presents displacements recorded at the selected station. Interactive tools are provided to zoom-in, move, and save the seismograms.

The resulting HTML files of these four examples are available in the Ⓔ electronic supplement and online (see Data and Resources). A detailed manual is provided to help users reproduce these examples and possibly make changes. The manual also documents the data format used for all four examples, options that can be used to customize the visualizations, and other related information that may help users to create their own visualizations. Additional tools are provided in the repository, such as a Bash script to prepare boundary data for other regions.

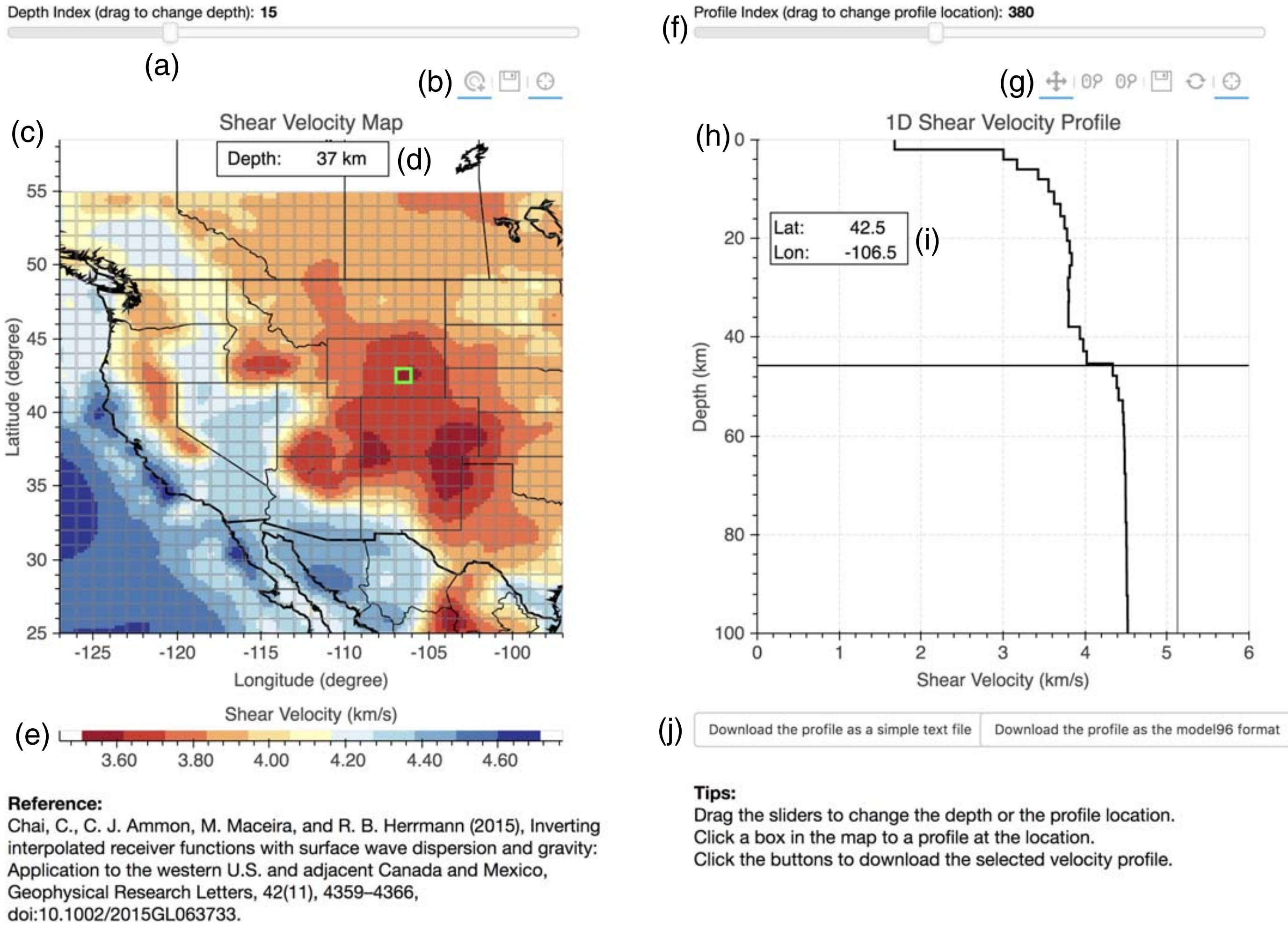
Visualization Creation Workflow

To create these visualizations, we followed a simple workflow. The codes rely on the Bokeh packages and are written in Python, similar to examples available on the Bokeh website. Our interactive plots, source code, and associated data files have been uploaded online for interested readers. The basic idea is to import and prepare the data, create the plots and user-interface elements in the display, and save the result to an HTML file. For the first example, the dispersion model



▲ **Figure 1.** A screenshot of the dispersion viewer identifying its user-interface elements. A dispersion map at 10 s and a dispersion curve are selected as examples. (a) The period index slider selects the period of the dispersion map. (b) Interactive tools for the map-view plot include tap (used to select a dot in the dispersion map), save, and crosshair from left to right. The underline indicates which tool is activated. (c) The dispersion map-view plot. Each dot represents a grid point, and colors indicate group velocity values. A dot can be selected by a single click (or tap on touch screens). The enlarged circle (with thick outline) shows the currently selected grid point. (d) A label to show the period of the dispersion map that will change when a different period is selected. (e) The color scale of the dispersion map that changes with the period. (f) The curve index slider bar that can change the location of the dispersion curve. (g) Interactive tools for the dispersion-curve plot include pan, box zoom, wheel zoom, save, reset, and crosshair (from left to right). (h) The dispersion-curve plot at the selected grid point shows Rayleigh-wave group velocities as a function of period. The dispersion curve changes when a different location is selected in the dispersion map. (i) Labels that show the latitude and longitude of the dispersion curve. The color version of this figure is available only in the electronic edition.

is imported from simple text files and organized for both map and curve views using Python. Map information, such as the political boundaries and shorelines, are also loaded from simple text files. Our online supplements include ready-to-use border data for the United States region in the package, and an example Bash script is also included to show how to extract border data from the Generic Mapping Tools (GMT). Initial (prior to user interaction) period and location values are selected, the data needed for the map and the curve are extracted from the dispersion model, and a map and curve display are created using Bokeh classes and functions (data source, figure, plot, etc.). Using Bokeh, we define interactive communication between elements of the display, including sliders (Fig. 1a,f) and the color bar (Fig. 1e). Interactive elements that change the data selection initiate extraction of the updated data from the data source, which then triggers the plots to update. When layout and design is complete, all the data and graphical components are translated to JavaScript and CSS by Bokeh and saved into a single HTML file that can be loaded into any popular modern browser. The HTML file can be sent to interested readers or hosted online. Once the readers open the HTML file in their browsers, the HTML file will present the data as designed (after downloading the required JavaScript and CSS automatically when they are not embedded). Flowcharts of the velocity model visualization are shown in Figure 5.

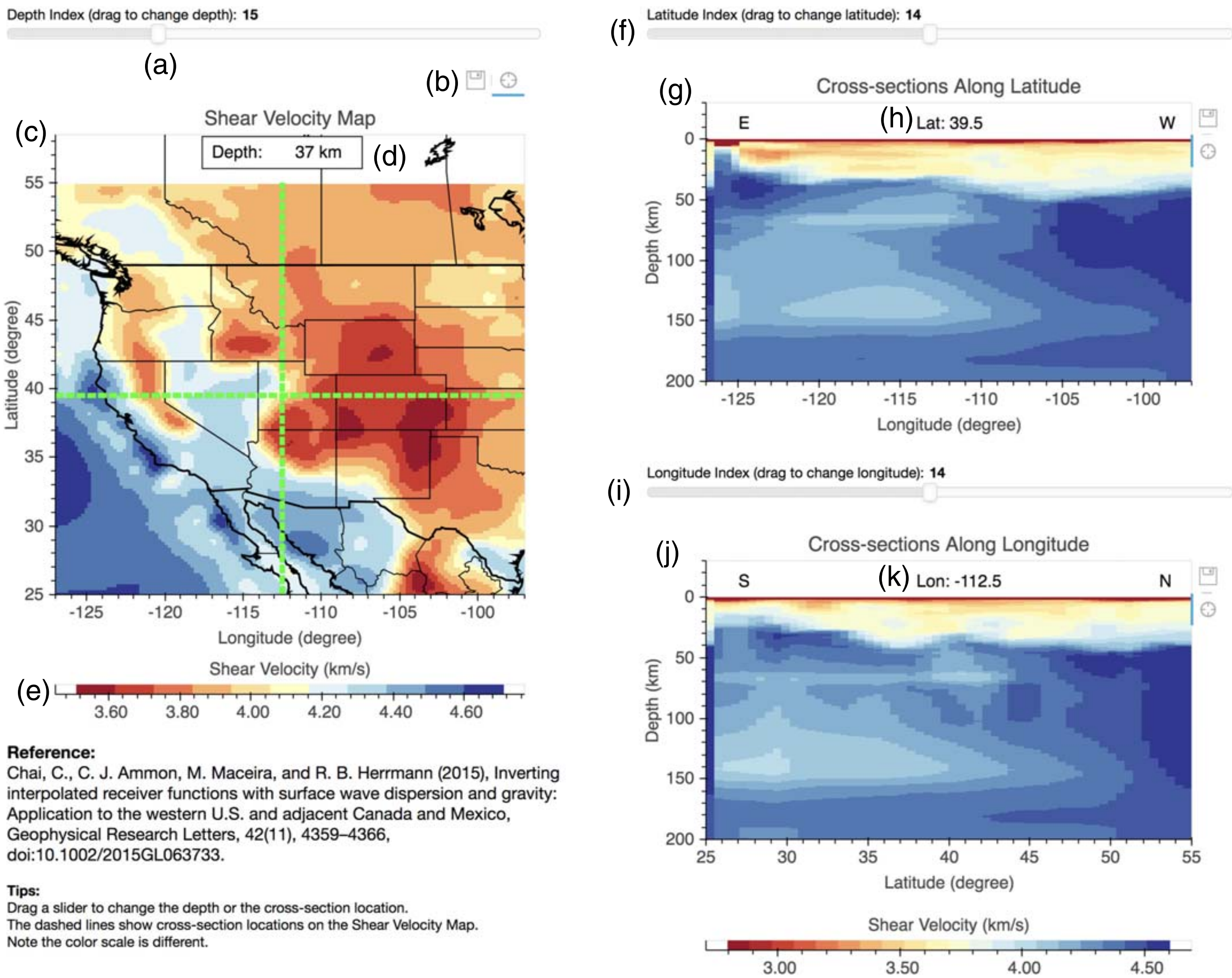


▲ **Figure 2.** A screenshot of the model viewer elements. (Left) A depth slice at 37 km and (right) a 1D shear velocity profile are selected as examples. (a) The depth index slider selects the depth of the shear velocity map. (b) Interactive tools for the map-view plot including tap, save, and crosshair from left to right. (c) A depth slice of the shear velocity map with boxes showing the grid, colors representing shear velocity values and the box with thick outline shows the selected grid point. (d) A label to show the depth of the shear velocity map that changes with the depth index slider. (e) The color scale of the shear velocity map that changes with depth. (f) The profile index slider selects the location of the velocity profile. (g) Interactive tools for the profile plot including pan, wheel zoom (*y* axis), wheel zoom (*x* axis), save, reset, and crosshair from left to right. (h) The 1D profile plot at the selected grid point on the shear velocity map. The profile changes when a different grid point is selected. (i) Labels to show the latitude and longitude of the 1D profile. (j) Buttons to download the selected 1D profile in two formats: a simple text and the model96 format (Herrmann, 2013). The color version of this figure is available only in the electronic edition.

Development of the model viewer is similar to the dispersion viewer with additional considerations on the convention of velocity depth slices and velocity-model extraction functionality. Most published maps of lateral velocity variation are interpolated to show a smoother model; we interpolated and stored model depth slices after importing the 3D model from a text file. Data for the selected depth slice is used to create the shear velocity map. The selected depth slice is linked to a slider control. To allow users to select a velocity profile based on location, a grid is plotted on the velocity map. The grid is linked to a velocity-model profile array that contains data for all the shear velocity profiles, so that the velocity profile can be updated when a grid point is clicked. Additional slider controls are added and linked to the model data source to allow several methods to sweep through the model. Interactive Bokeh buttons are added to each display to allow zooming and the exporting of the model.

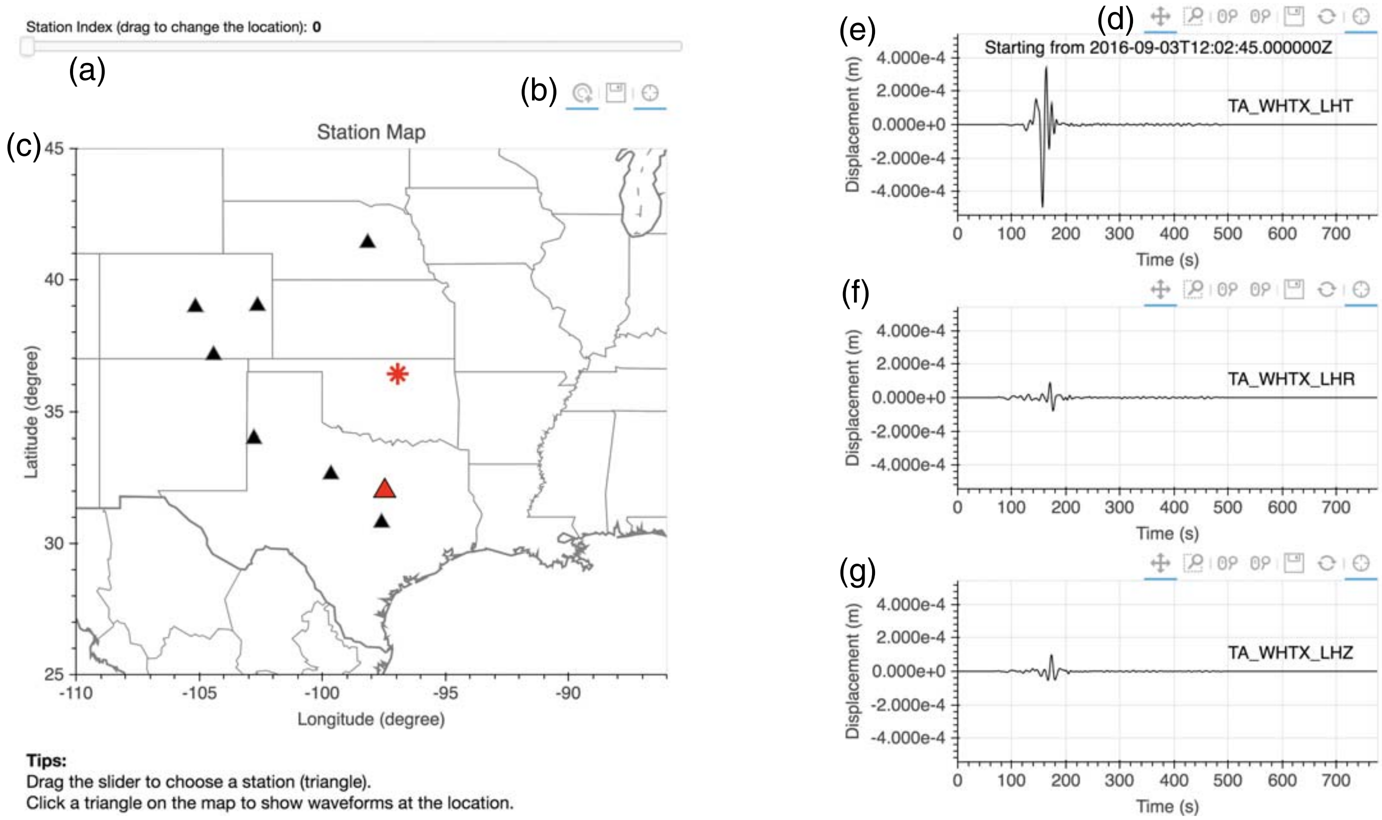
DISSUSSION AND CONCLUSION

Modern computer packages provide a number of powerful tools for the development of custom interactive portable displays of multidimensional seismic data and models. As an exploratory tool, such interactive displays allow researchers to explore their results with more detail and improved efficiency. Portability allows scientists to share their results in detail with



▲ **Figure 3.** A screenshot of the cross-section viewer indicating its elements. (Left) A depth slice at 37 km, (upper right) a cross section along latitude 39.5°, and (lower right) a cross section along longitude −112.5° are selected as examples. (a) The depth index slider selects the depth of the shear velocity map. (b) Interactive tools for the map-view plot, including save and crosshair from left to right. (c) A depth slice of the shear velocity map with colors representing shear velocities and dashed lines showing locations of the cross sections. (d) A label to show the depth of the shear velocity map that changes with the depth-index slider. (e) The color scale of the shear velocity map that changes with depth. (f) The latitude index slider bar that can change the location of the cross section along latitude. (g) The cross section along latitude with interactive tools, including save and crosshair from left to right, colors representing shear velocities. (h) A label shows the latitude of the cross section. (i) The longitude index slider bar that can change the location of the cross section along longitude. (j) The cross section along longitude, colors representing shear velocities. (k) A label shows the longitude of the cross section. The color version of this figure is available only in the electronic edition.

HTML files that can be viewed with any modern web browser. The file transports both the data (or model) and graphical tools for exploring or interacting with the model (exporting part of the model). Although an Internet connection is required for our interactive examples, embedding JavaScript libraries and CSS files in the HTML files, or providing the required files locally, allows offline scenarios. For those familiar with Python, the development time and cost for HTML-based interactive visualizations are significantly reduced, compared with the investments required for large commercial packages. All the required packages are open source and freely available online, which means everyone has access to HTML-based interactive plotting tools. Our examples show how users can effectively and efficiently explore 4D data sets. Interactive links between different displays create a powerful tool that can leverage community-familiar displays that evolved over many decades of visualization experimentation (in journal publications). Responsive interactive interfaces provide users quick access to



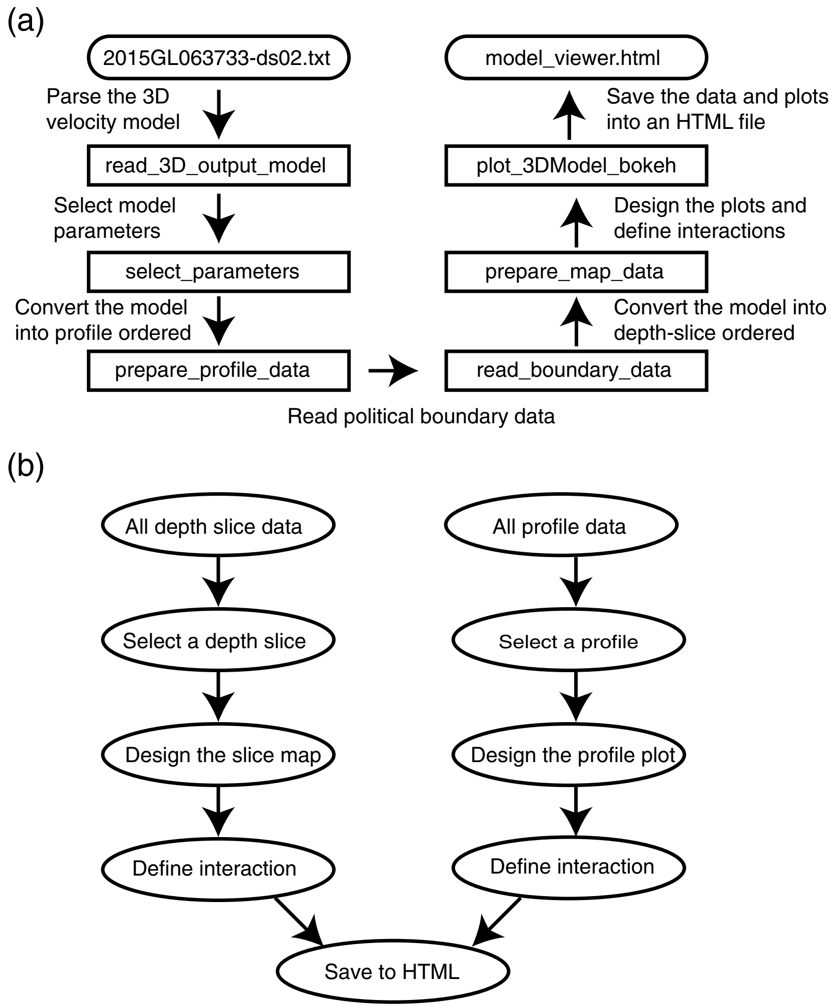
▲ **Figure 4.** A screenshot of the three-component seismograms viewer indicating its elements. (a) The station index slider selects a station to show seismograms. (b) Interactive tools for the station map plot, including tap, save, and crosshair from left to right. (c) A station map with triangles represent stations; the larger triangle indicates the selected one, and an asterisk for the event location. (d) Interactive tools for the seismogram plot including pan, box zoom, wheel zoom (*y* axis), wheel zoom (*x* axis), save, reset, and crosshair (from left to right). (e–g) Seismogram plots for three components. The color version of this figure is available only in the electronic edition.

interesting features of data and models. Our tools can be used to visualize seismic-velocity models, such as those contained in the IRIS Earth Model Collaboration (see Data and Resources).

The Bokeh package has some limitations. Map projections are not well supported, although more functions for geographical data may be included in future versions. Also, some of the interactive functionality that we included (clicks, slider bar, and buttons) use JavaScript snippets inside Python scripts to update the data of images. The embedded JavaScript snippets are easy to learn and well documented online. Performance could be a concern when many data sets were saved to an HTML file. However, we did not find this issue for the four examples we present (or in our research tools that linked a 3D shear-speed model with receiver function and surface-wave dispersion observations). An obvious issue that must be considered is computer security. We have created, used, and shared the dynamic visualizations described in this work, and they helped us explore our models. However, the files contain executable codes, so some care should be exercised when using and sharing these packages. Browser security is generally strong, so we do not consider it a major problem, but as with any computer package or application, security must be considered. Finally, advanced options such as arbitrary data slicing and real-time filtering can be difficult to implement because data processing libraries in JavaScript are limited and some calculations are time consuming. One way to bypass this issue is to process the data using Python and then save the results with Bokeh, as we did for the examples presented.

The intended audience of our interactive plots and examples ranges from undergraduate students to experienced researchers. The examples we presented provide a user-friendly way of exploring and selecting seismic models. Our view is that a little effort in creating a custom Bokeh-based visualization tool can be a valuable alternative to powerful but difficult-to-use packages that do not always allow simple, familiar display of seismic information (models and data). Rather than forcing our information into these packages, at least in some instances, combining traditional views of the information with interactive browser-based tools is a more effective way to communicate our results and allow others to explore our models with the detail that usually only the authors have enjoyed because it is laborious and impossible to communicate through traditional scientific publications.

As the interactive visualizations are intuitive to handle, they can easily be used for education and outreach purposes. The interface of the visualizations is easy to learn and provides



▲ **Figure 5.** Flow charts showing the design of the velocity model visualization in Figure 2. (a) The major steps involved in generating the visualization. Key functions used in the script (available online) are outlined with rectangles. Input and output files are identified with round rectangles. (b) The basic structure of the key function plot_3DModel_bokeh.

students access to complicated research results that are usually difficult to communicate. In general, the interactions provided by the visualizations offer a more interesting tool for students. Because the resulting HTML files are portable and accessible, students can study these visualizations at their own pace. Our examples and tools can help students efficiently learn recent developments in seismology.

DATA AND RESOURCES

NumPy v.1.13.1 (van der Walt *et al.*, 2011), SciPy v.0.19.1 (E. Jones, E. Elephant, P. Peterson, *et al.*, SciPy: Open Source Scientific Tools for Python, http://www.scipy.org, last accessed October 2017), ObsPy v.1.0.3 (Beyreuther *et al.*, 2010; http://www.obspy.org, last accessed October 2017), and Bokeh v.0.12.13 (http://bokeh.pydata.org, last accessed October 2017) were used to process and plot data. Anaconda was used to install Python packages (https://www.continuum.io, last accessed October 2017). Surface-wave dispersion data are extracted from the group velocity dispersion for North America of Herrmann *et al.* (2016) (http://www.eas.slu.edu/eqc/eqc_research/NATOMO, last accessed June 2017). The Generic Mapping Tools (GMT) v.5.2.1 was used to obtain border data (Wessel *et al.*, 2013; http://gmt.soest.hawaii.edu, last accessed June 2016). The source code, data, and HyperText Markup Language (HTML) files used in this article can be found at https://github.com/ccp137/DynamicViz (last accessed December 2017). A detailed manual is also provided online to help interested readers apply these tools to their own data or models. The HTML files are also included as an Ⓔ electronic supplement. For the tools in the Incorporated Research Institutions for Seismology (IRIS) Earth Model Collaboration, see http://ds.iris.edu/ds/products/emc (last accessed December 2017).

ACKNOWLEDGMENTS

The authors acknowledge the support by the U.S. National Science Foundation (Grant Numbers EAR-1053484 and EAR-1053363) and by the U.S. Department of Energy (Grant Number LDRD-20120047ER and Contract Number DE-AC05-00OR22725 with Oak Ridge National Laboratory). The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Government. The authors also acknowledge the developers of the open-source packages NumPy, SciPy, Generic Mapping Tools (GMT), ObsPy, and Bokeh. The facilities of Incorporated Research Institutions for Seismology (IRIS) Data Services, and specifically the IRIS Data Management Center, were used for access to waveforms, related metadata, and derived products used in this study. The authors also thank Editor-in-Chief Z. Peng, as well as the associate editor and three anonymous reviewers, for their positive and constructive comments and suggestions that helped improve the quality of this article.

REFERENCES

Beyreuther, M., R. Barsch, L. Krischer, T. Megies, Y. Behr, and J. Wassermann (2010). ObsPy: A Python toolbox for seismology, *Seismol. Res. Lett.* **81,** no. 3, 530–533, doi: 10.1785/gssrl.81.3.530.

Bezada, M. J. (2014). The waveform browser, an interactive and intuitive tool for exploring seismic event data, *Seismol. Res. Lett.* **85,** no. 1, 192–196, doi: 10.1785/0220130032.

Chai, C., C. J. Ammon, S. Anandakrishnan, C. Ramirez, and A. Nyblade (2017). Estimating subglacial structure using *P*-wave receiver functions, *Geophys. J. Int.* **209,** no. 2, 1064, doi: 10.1093/gji/ggx075.

Chai, C., C. J. Ammon, M. Maceira, and R. B. Herrmann (2015). Inverting interpolated receiver functions with surface wave dispersion and gravity: Application to the western U.S. and adjacent Canada and Mexico, *Geophys. Res. Lett.* **42,** no. 11, 4359–4366, doi: 10.1002/2015GL063733.

Herrmann, R. B. (2013). Computer programs in seismology: An evolving tool for instruction and research, *Seismol. Res. Lett.* **84,** no. 6, 1081–1088, doi: 10.1785/0220110096.

Herrmann, R. B., H. Benz, and C. J. Ammon (2016). Mapping Love/Rayleigh phase/group velocity dispersion between 2 and 100 seconds in North America using ambient noise cross-correlations and earthquake observations (abs), *Seismol. Res. Lett.* **87,** no. 2B, 546.

Incorporated Research Institutions for Seismology Data Management Center (IRIS-DMC) (2011). Data services products: EMC, a repository of Earth models, doi: 10.17611/DP/EMC.1.

Menke, W. (2012). *Geophysical Data Analysis: Discrete Inverse Theory: MATLAB Edition*, Academic Press, Elsevier, Oxford, United Kingdom, 263 pp.

Pavlis, G. L., K. Sigloch, S. Burdick, M. J. Fouch, and F. L. Vernon (2012). Unraveling the geometry of the Farallon plate: Synthesis of three-dimensional imaging results from USArray, *Tectonophysics* **532,** 82–102, doi: 10.1016/j.tecto.2012.02.008.

van der Walt, S., S. C. Colbert, and G. Varoquaux (2011). The NumPy array: A structure for efficient numerical computation, *Comput. Sci. Eng.* **13,** no. 2, 22–30, doi: 10.1109/MCSE.2011.37.

Wessel, P., W. H. F. Smith, R. Scharroo, J. Luis, and F. Wobbe (2013). Generic Mapping Tools: Improved version released, *Eos Trans. AGU* **94,** no. 45, 409–410, doi: 10.1002/2013EO450001.

West, J. D., and M. J. Fouch (2012). EMERALD: A web application for seismic event data processing, *Seismol. Res. Lett.* **83,** no. 6, 1061–1067, doi: 10.1785/0220110138.

Chengping Chai[1]
Charles J. Ammon
Department of Geosciences
The Pennsylvania State University
440 Deike Building, University Park
Pennsylvania 16802 U.S.A.
cchai.psu@gmail.com

Monica Maceira[1]
Oak Ridge National Laboratory
1 Bethel Valley Road, Oak Ridge
Tennessee 37930 U.S.A.

Robert B. Herrmann
Department of Earth and Atmospheric Sciences
Saint Louis University
3642 Lindell Boulevard, Saint Louis
Missouri 63108 U.S.A.

Published Online 14 February 2018

[1] Now at Department of Physics and Astronomy, The University of Tennessee, 1408 Circle Drive, Knoxville, Tennessee 37996 U.S.A.