Visual Exploration and Analysis of Time Series Earthquake Data

A. Chourasia
9500 Gilman Dr., MC 0505
La Jolla, CA - 92093
amit@sdsc.edu

K. B. Richards-Dinger
900 University Ave.
Riverside, CA - 92521
keith.richards-dinger@ucr.edu

J. H. Dieterich
900 University Ave.
Riverside, CA - 92521
dieterichj@ucr.edu

Y. Cui
9500 Gilman Dr., MC 0505
La Jolla, CA - 92093
cui@sdsc.edu

ABSTRACT

Earthquake hazard estimation requires systematic investigation of past records as well as fundamental processes that cause the quake. However, detailed long-term records of earthquakes at all scales (magnitude, space and time) are not available. Hence a synthetic method based on first principals could be employed to generate such records to bridge this critical gap of missing data. RSQSim is such a simulator that generates seismic event catalogs for several thousand years at various scales. This synthetic catalog contains rich detail about the earthquake events and associated properties.

Exploring this data is of vital importance to validate the simulator as well as to identify features of interest such as quake time histories, conduct analyses such as calculating mean recurrence interval of events on each fault section. This work[1] describes and demonstrates a prototype web based visual tool that enables domain scientists and students explore this rich dataset, as well as discusses refinement and streamlining of data management and analysis that is less error prone and scalable.

CCS CONCEPTS

• **Human-centered computing** → **Visualization** → **Visualization application domains** → Geographic visualization

• **Information systems** → **Information systems applications** → **Spatial-temporal systems** → Geographic information systems

KEYWORDS

Visualization, data management, earthquake simulators

1 INTRODUCTION

Earthquake simulators are computer codes that can resolve the discrete fault-slip events across the scale (magnitude, space and time) needed to track the state evolution for the brittle regions of the solid Earth. We will develop and apply the most capable earthquake simulators to investigate brittle deformation, fault interaction, and earthquake predictability. The need to generate 10^5 -10^7 earthquakes in simulations spanning ≥104 years precludes full representation of inertial dynamics in the simulations, so the more advanced simulators incorporate quasi-dynamic approximations such as radiation damping.

RSQSim incorporates rate-state constitutive properties. It has unique capabilities to deterministically model short-term clustering together with long-term statistical properties of earthquakes [1, 2]; and to represent the different modes of slip observed in nature. Through the use of analytic approximations, and a computational approach based on sliding state transitions, RSQSim is very efficient numerically [3]. This efficiency enables repeated simulations of long earthquake catalogs (10^5 to 10^7 events) with outer scales of the dimensions of regional plate boundaries, and sufficiently resolved inner scales to permit detailed simulations of the evolution of system state through occurrence of frequent small earthquakes. This study focuses on visualization, analysis and data management aspect for the data produced on high performance computing resources by the RSQSim simulator.

Table 1: Data properties for a sample catalog

Size	2.4 GB
Files	2 ascii, 9 binary files
Catalog time duration	From 50k to 90k years
Number of events	5,970,621
Event variables	12
Number of fault patches	260,051
Fault patch variables	11
Number of event actions	19,127,461
Event action variables	8

2 DATA WRANGLING

2.1 Source Data

The data is in form of a time series catalog containing millions of earthquake events with varying magnitudes and occurrence intervals within the entire catalog that spans tens of thousands of years. Metadata for a sample catalog is listed in Table 1. An earthquake catalog is generated for each simulation scenario for a given time period. Each catalog consists of events (earthquakes), patches (a geometrically well defined portion of a fault) and action (relationship between event and patches and their properties). A brief overview of event, patch and action information is as follow

a) Event information: This columnar text data includes implicit event ID, time, event magnitude; event origin coordinates in UTM projection [4], event duration, name of fault section and few other properties.
b) Patch information: This columnar text data includes implicit fault patch ID, fault patch geometry either as triangles or rectangles, its coordinates in UTM projection, fault section ID and the name to which this patch belongs.
c) Action information: This data is a set of binary files that record the action for every given earthquake event and its effect on affected patches and associated properties such as change in slip, stress, etc.

2.2 Data Transformation

Working with source data required parsing, restructuring and in memory indexing, this requires processing time before visualization or analysis to be conducted. Iterative refinements to the implementations of raw data handling improved from 20 minutes to 2 minutes on a MacPro workstation with 2x 2.26 Ghz Quad Core Intel Xeon processor and 16 GB memory. However this duration was not ideal as it was still too slow when application would restart. Furthermore, this implementation was single user oriented and could not be easily deployed on the web.

The source data can be mapped to a relational database in a straightforward manner by representing the data with three tables; one for events, second for patches, and third for actions that includes relationship between former two. SQLite [5] database was chosen based on following considerations

a) The database will be written once, rarely modified, but read repeatedly for visualization and analysis
b) SQLite simplifies data management, but keeps notion of a file, this is desirable as the domain scientists are used to file handling
c) SQLite does not requires database server setup, this lowers setup hurdles for domain scientists
d) SQLite database is portable on multiple platforms, thus can be easily generated on diverse computing resources and shared easily with others
e) SQLite drivers are available in many languages, thus can be easily used from variety of languages such as R [6] and Python [7]

The data transformation was conducted using Python and validated to match source data. During the data transformation geographic data was translated from UTM projection to EPSG:4326 projection [8] and stored in GeoJSON format [9] for quick retrieval later. The data translation process takes approximately 20 minutes on a workstation mentioned previously; the resulting database is almost twice the size, mainly due to addition of ID columns that were implicit in source data and extra computed columns as well as translated geo-referenced data, which is more verbose.

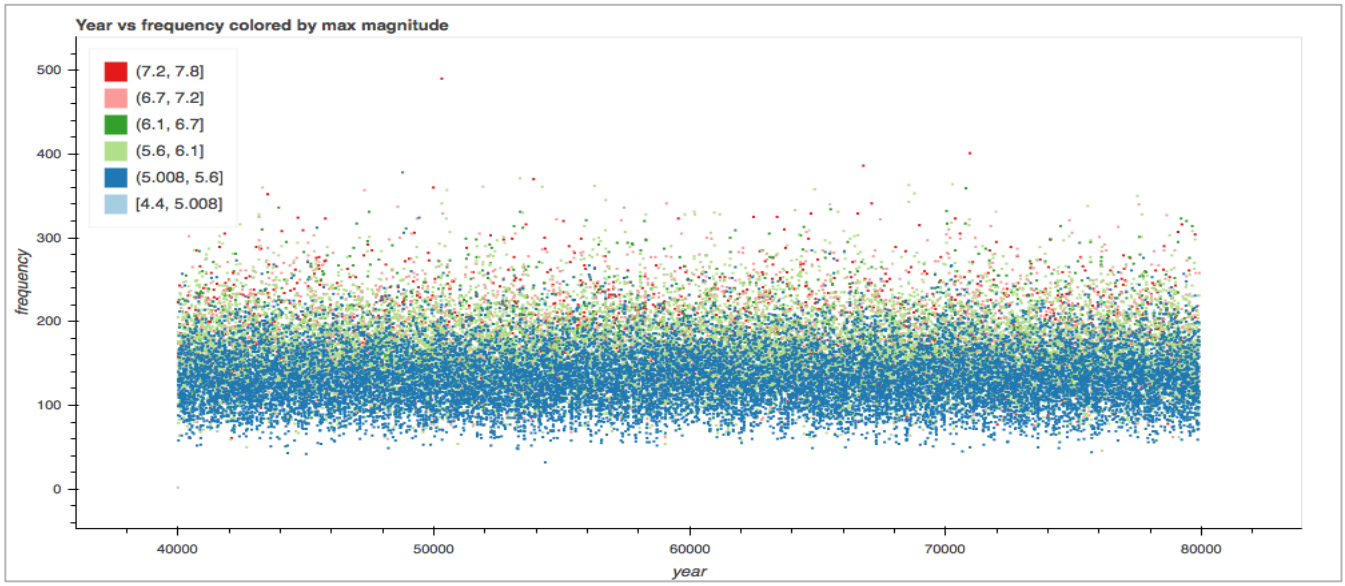

Figure 1: Frequency distribution of quakes for 40,000-year period. Magnitude range is indicated by color.

3 VISUALIZATION

3.1 Tasks

The domain scientists had developed a method to view the data in 3D using R scripting. As most quake events affect very few fault patches the 3D aspect was deemed to be of less interest, instead we focused on developing a projected 2D visualization that would allow the domain scientists to accomplish the following

a) Browse events and view them in geographic context
b) View the fault patches affected by a given quake
c) View chronology in space and time of events that precede and follow a selected event
d) Filter events based on magnitude and number of affected fault patches
e) Provide a web interface to easily investigate and disseminate this data

3.2 Visual encoding

To accomplish the set tasks, we employed map visual idiom (see Fig. 2) that implicitly represents geographic spatial data and line plot idiom (see Fig. 3) that allow us to represent time series data in abstract form. The map shows geographic context with roads, cities and other landmarks, the data encoded using circle markers and displayed at its geo-referenced locations. Events on map are shown by polygons with shape corresponding to their magnitude in whole number. Events with magnitude less than 4 are shown as triangles, event with magnitude between 4 and 5 are shown with a square, events between 5 and 6 are shown as pentagon and so on. The selected event is shown with red color polygon, clicking on this polygon presents a popup window containing contextual information associated with this event as shown in Table 2.

Figure 2: Map showing selected event with a red circle and corresponding affected fault patches in blue. Previous events are displayed with grey circles connected by a grey line that indicates their chronology with respect to the selected event. Similarly next events are displayed with purple circles connected by a purple line. Visibility control toggles for various layers are on top right.

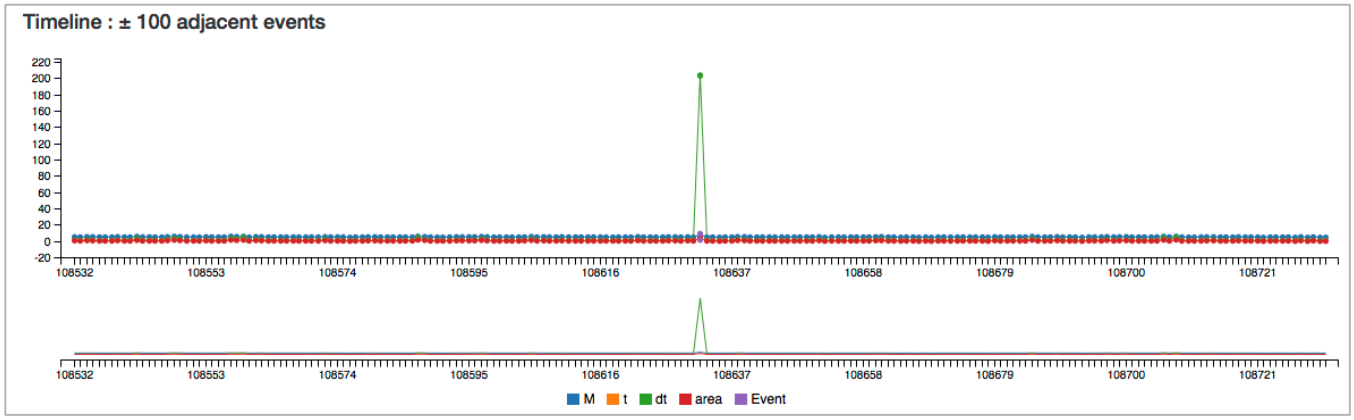

Figure 3: Line plot shows magnitude, duration and area of 100 previous and 100 next events that are adjacent the selected one shown in the center. Clicking on any event on the time series loads the chosen event in the map above, enabling swift exploration.

Similarly previous and next events adjacent to the selected event are displayed as grey and purple polygons respectively, these polygons are connected via a trail line to indicate event chronology. Affected fault patches are projected in 2D and are displayed as explicit geometry in blue color. All these elements including the base maps are separated into layers such that they can be either displayed or hidden on demand.

Lastly, a timeline chart below the map displays event chronology of ± 100 events adjacent to the selected event. The timeline chart includes few properties such as magnitude, area and duration of events. The timeline and the map interface are linked such that when an event is selected in the timeline the map is updated.

Table 2: Pop up information for a selected event ID

Event properties	Value
Event ID	108632
M-Magnitude	7.650801
M0-Moment (Nm)	3.76879000084e+20
T0-Time (s)	1.60341098492e+12
DT-Duration (s)	203.089111328
Patch ID	184145
X-Coord (m)	448831.639048
Y-Coord (m)	3797705.13249
Z-Coord (m)	-7542.424242
Area (m^2)	3.66856720806
Patches affected	5194

3.3 User interface

The catalog consists of millions of events, thus a careful user interface design was needed to accomplish event selection. We implemented the graphical users interface (GUI) which could be subdivided into a composite of three sections (see Fig. 4).

The top section allows event selection via event ID (Fig. 5) or time. In case the of event selection via time which is natively stored in seconds, on interaction we show the time in human readable format as year:day:hour:second (see Fig. 6). Both ID and time based selections offer sliders for quick pick and text input for precise input. Additionally increment and decrement buttons at various scales are provided for swiftly changing the selection.
The middle section of the GUI allows filtering of selected event by magnitude and number of patches affected. The corresponding sliders for these filers offer the user an ability to set upper and lower range as desired. The Map trail and Time slider allow the user to customize the number of events to be shown on map and time series chart respectively.

The bottom section of the GUI provides checkboxes for applying filters on time series, auto fitting of all displayed events on the map and an ability to set auto submission rather than submitting changes manually after desired changes.

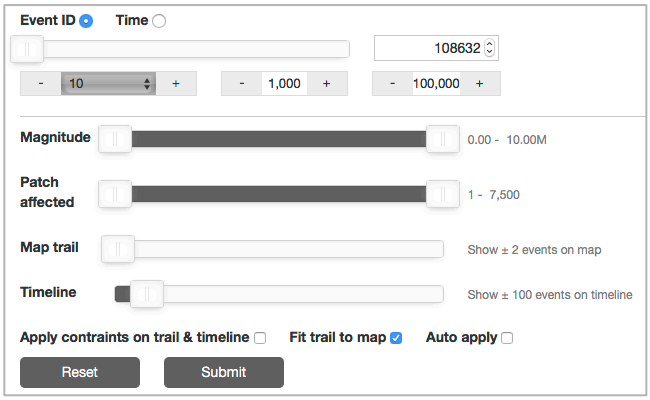

Figure 4: Snapshot of full graphical user interface, showing selection by event ID. The event can be further filtered via magnitude and patch sliders. Trail and time series sliders allow customization of ancillary items for display.

Figure 5: Selection by event ID snapshot. The interface allows users to provide input via a slider (for easy scrubbing), text input box (for precise numeric input) as well as increment and decrement buttons at preselected scales for swift exploration. A drop down selection list enables choice for additional scales.

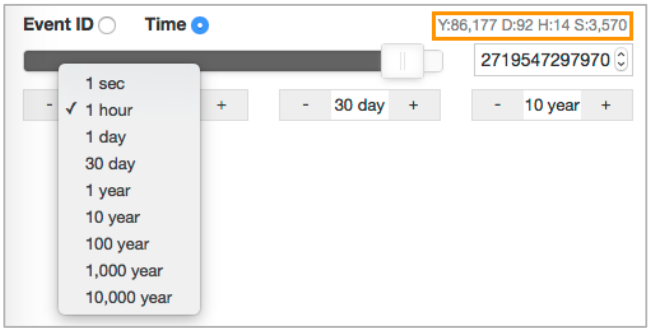

Figure 6: Selection of event by time. The interface allows users to provide input via a slider (for easy scrubbing), text input box (for precise numeric input) as well as increment and decrement buttons at preselected scales for swift exploration. A drop down selection list enables choice for additional scales. On user interaction human readable time in year:day:hour:second format is shown in the orange box.

3.3 Implementation

Custom visualization is implemented using Python, JavaScript and HTML as a server-client application. The server side handles processing and responds with renderable data to the client. The client (web browser) fetches the served data and displays the result along with the user interface. The application relies on LeafletJS mapping library [10] that provide geographic environment. Flask framework [11] in Python is utilized for web serving and query response interaction. The selection and filters are mapped to appropriate database queries on server side. Folium Python module [12] is used to generate custom map with custom code that fuses data for selected event and fault patches ondemand. The timeline chart is generated using the C3JS JavaScript library [13] and linked with map visualization. The graphical user interface is created using noUISliderJS JavaScript library [14], which provides mobile friendly and range sliders that are not natively available with HTML5. The web application is deployed using Gunicorn [15] web server.

4 RESULTS AND DISCUSSION

We have developed and demonstrated a web based visualization application [16] that displays an interactive map with set of layers that includes base map, selected event, trail events (previous and next events adjacent to the selected event), trail event line (line connecting previous, selected and next events) and affected fault patches. These layers can be toggled on/off to reduce clutter and the map could be zoomed in/out or panned.

A graphical user interface allows the users to interactively select, browse, search and filter events of interest, refer to section 3.3 for interaction details. Users can spatially view location of chosen event (see red polygon in Fig. 2) and its chronology on a map (see black and purple polygons connected by a grey and purple lines in Fig. 2) as well as inspect other calculated properties such as area, duration by clicking on these markers. The users can also view the patches affected by the event spatially (see series of projected blue triangles Fig. 2). Finally the event chronology is shown as line plot, which allows easy comparison and trend assessment of magnitude, duration and area (see Fig. 3) adjacent to the selected event. Each event in this timeline in linked to map visualization, such that when the user clicks on any event, the clicked event is updated on the map. This linking enables swift navigation and exploration.

In addition to the interactive visualization, the newly implemented data management scheme can be used for scientific analysis. One such analysis is implemented and the results were validated with the domain scientists. The analysis is computation of mean recurrence interval of quake on each fault sub section from a total of 2,606.

The following steps illustrates the computation process

i. Find all unique fault sections in fault geometry table
ii. For each section find event record in event table
iii. Arrange records from ii in chronological order
iv. Calculate time duration between successive records by computing pairwise difference of all records in iii
v. Compute mean of all items in iv
vi. Repeat ii–v for each section

This analysis using the original workflow with R scripts consumes more time compared to the new one that uses database. The original analysis required loading, parsing and structuring of raw data, which is not need by new analysis, as it relies well structured database. More importantly the new analysis does not need to load entire data into memory as the database is kept on disk. This demonstrates that the analysis-using database will offer better scalability and convenience for domain scientists.

The domain scientists are currently using this web based visualization interface for data exploration and also considering its use for pedagogy purpose in classroom.

5 CONCLUSIONS

To conclude, we have developed and demonstrated an interactive web based application for visualizing of large time series earthquake data. The application not only allows easy browsing and filtering, but also is also easily accessible by multiple concurrent users. Furthermore, we have transformed the original data handling to a modern environment that lends itself to error reduction as less code is required for custom parsing and filtering, easier data management, easy dissemination and most importantly scalable data analysis and visualization.

6 FUTURE WORK

We would like to refine and extend this work for the entire scientific pipeline; this includes writing output directly to the database from simulation, rather than translating it as a post-processing step. Refine the visualization to include graphical spatial region selection. Provide an ability to download filtered data and perform custom queries on the web application. Finally, refactor all existing analysis routines to use database queries.

ACKNOWLEDGMENTS

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1053575. The National Science Foundation grant number EAR-1135455 and Keck Foundation grant number 005590-00001 also supported this work.

REFERENCES

1. Dieterich, J. H., Applications of rate- and state-dependent friction to models of fault slip and earthquake occurrence, In: Treatise On Geophysics, Vol. 4. Elsevier, Oxford, *Vol. 2 Earthquake Seismology, Elsevier*, 107-129, 2007.
2. Dieterich, J., and K. Richards-Dinger, Earthquake recurrence in simulated fault systems, *Pure Appl. Geophys*., 167, 1087-1104, 2010.
3. Pekurovsky, D., A. Chourasia, K. B. Richards-Dinger, B. E. Shaw, J. H. Dieterich, and Y. Cui (2016). Performance enhancements and visualization for RSQSim earthquake simulator. Presented at the 2016 SCEC Annual Meeting. Palm Spring, CA, Sep 11-14, 2016.
4. Dutch, S. The Universal Transverse Mercator System. Retrieved Jun 8, 2017 from https://www.uwgb.edu/dutchs/FieldMethods/UTMSystem.htm
5. SQLite – SQLite Home Page. Retrieved Jun 8, 2017 from https://sqlite.org/index.html
6. R: The R Project for Statistical Computing. Retrieved Jun 8, 2017 from https://www.r-project.org
7. Python. Retrieved Jun 8, 2017 from https://www.python.org
8. WGS84: EPSG Projection – Spatial Reference. Retrieved Jun 8, 2017 from http://spatialreference.org/ref/epsg/4326
9. GeoJSON, 2016. Retrieved Jun 8, 2017 from http://geojson.org
10. Leaflet – a JavaScript library for interactive maps. Retrieved Jun 8, 2017 from http://leafletjs.com
11. Flask (A Python microframework). Retrieved Jun 8, 2017 from http://flask.pocoo.org
12. Folium: python data leaflet.js. Retrieved Jun 8, 2017 from https://github.com/python-visualization/folium
13. C3.js | D3-based reusable chart library. Retrieved Jun 8, 2017 from http://c3js.org
14. noUISlider – JavaScript Range Slider | Refreshless.com. Retrieved Jun 8, 2017 from https://refreshless.com/nouislider
15. Gunicorn – Python WSGI HTTP Server for UNIX. Retrieved Jun 8, 2017 from http://gunicorn.org
16. Chourasia. A..Visualization for RSQSim. 2017. Retrieved Jun 8, 2017 from http://vis.sdsc.edu:5555