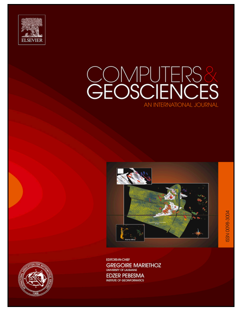
# Accepted Manuscript




Reducing process delays for real-time earthquake parameter estimation – An application of KD tree to large databases for Earthquake Early Warning

Lucy Yin, Jennifer Andrews, Thomas Heaton

PII: S0098-3004(17)30596-4

DOI: 10.1016/j.cageo.2018.01.001

Reference: CAGEO 4074

To appear in: *Computers and Geosciences*

Received Date: 29 May 2017

Revised Date: 2 December 2017

Accepted Date: 10 January 2018

# Reducing process delays for real-time earthquake parameter estimation – an application of KD tree to large databases for Earthquake Early Warning

Lucy Yin[1], Jennifer Andrews[2], Thomas Heaton[1,2]

[1] Department of Civil and Mechanical Engineering
California Institute of Technology
Pasadena, USA

[2] Division of Geological and Planetary Sciences
California Institute of Technology
Pasadena, USA

Corresponding Author: Lucy Yin

Department of Civil and Mechanical Engineering
California Institute of Technology
1200 E California Blvd
Pasadena, CA USA 91106
Tel: 1-626-841-9702
Email: lyin@caltech.edu

**Abstract**

Earthquake parameter estimations using nearest neighbor searching among a large database of observations can lead to reliable prediction results. However, in the real-time application of Earthquake Early Warning (EEW) systems, the accurate prediction using a large database is penalized by a significant delay in the processing time. We propose to use a multidimensional binary search tree (KD tree) data structure to organize large seismic databases to reduce the processing time in nearest neighbor search for predictions. We evaluated the performance of KD tree on the Gutenberg Algorithm, a database-searching algorithm for EEW. We constructed an offline test to predict peak ground motions using a database with feature sets of waveform filter-bank characteristics, and compare the results with the observed seismic parameters. We concluded that large database provides more accurate predictions of the ground motion information, such as peak ground acceleration, velocity, and displacement (PGA, PGV, PGD), than source parameters, such as hypocenter distance. Application of the KD tree search to organize the database reduced the average searching process by 85% time cost of the exhaustive method, allowing the method to be feasible for real-time implementation. The algorithm is straightforward and the results will reduce the overall time of warning delivery for EEW.

**Highlights**

- Presented a multidimensional binary search (KD) tree database structure for seismic data

- Reduced average searching time by 85% for real-time seismology predictions
- Suggested to directly predict ground motion information for Earthquake Early Warning due to accuracy
- Evaluated pros and cons of modeling approach and big data search approach for real-time seismology

## Introduction

Due to the advancement of information technology in the past few decades, Earthquake Early Warning (EEW) systems are able to analyze ground motions in real-time and provide alerts before the onset of the destructive wave at specific facilities (Heaton, 1985) (Allen et al., 2003). EEW is based on the principle that the damaging earthquake ground motion propagates more slowly than electronic information, so warnings can be successfully delivered immediately after detecting the first earthquake signals at a seismic station (Cua, 2005). The speed of the more damaging S-waves from earthquakes is about 3.5km/s, whereas electrically transmitted signals from the seismic network sensors travel at about $3.0 \times 10^5$km/s.

EEW is most beneficial for earthquakes causing a significant level of ground shaking, so the alert speed is critical to provide a warning to the most strongly affected areas close to the epicenter. Additionally, for high-cost user actions (such as halting industrial processes), the accuracy of ground motion predictions at user sites is important for the widespread adoption and use of EEW (Hoshiba, 2013). In general, the conventional algorithms use trained models to estimate earthquake source parameters (such as magnitude and hypocenter distance) from station ground motion observations, and then apply ground motion prediction equations to estimate the peak ground motion experienced at different user sites (Wu et al., 2007) (Zuccolo et al., 2016) (Kuyuk et al., 2014). The predictive models tend to compress the observed information into a few source parameters, which can overly simplify the behavior of wave propagation through the Earth (Meier, 2017). Significant error in final prediction results can be accumulated through the uncertainties in the underlying models (Bose et al., 2009) (Allen et al., 2009). As a result, for the purposes of a real-time EEW system, it is a challenge to create a simple model that fully captures all the attributes that influence the peak ground motion in a recorded waveform, such as magnitude, location, depth, soil type, local site condition, directivity, source radiation.

Fingerprint searching and template match methods are alternative approaches to EEW and have also recently been employed in other areas of seismology (Yoon et al., 2015). In the fingerprint searching method, important waveform characteristics are extracted from each earthquake record to form an extensive database of "earthquake fingerprints". During the occurrence of an on-going earthquake, the algorithm searches among the database for the most similar "earthquake fingerprints", and then estimates the source parameters or peak ground motions of the new event based on the searched records. A recently developed method, called the Gutenberg Algorithm (GbA) (Meier et al., 2015), applies the fingerprint-searching concept to EEW by abstracting the time-frequency amplitude information of the real-time seismic signal for various filter bands to create a large-scale database, and then estimates the earthquake source parameters such as magnitude and hypocenter distance for on-going earthquakes. In addition, the template-matching method in FinDer (Böse et al., 2012) compares observations with a database of theoretical spatial ground motion patterns to estimate earthquake source parameters and peak ground shaking at various sites. Another example of real-time earthquake monitoring algorithm using search engine method was developed for the estimation of source-focal mechanism (Zhang et al. 2014). Although the predictable variable in the example above are different, all of the methods share the common approach of searching among a pre-processed database.

One of the most important factors required of search algorithms is that the searched database needs to be sufficiently large in order to cover a wide range of potential earthquakes. In other words, if similar data to the target query are not included in the database, the searched result could be significantly off from the true value. As an example, the records in the databases should represent the natural distribution of earthquake occurrence as described by the Gutenberg-Richter relationship (Gutenberg and Richter, 1944); there should be many more small events than large ones because small size earthquakes occur more often than large earthquakes, so the search should returns reflect real earthquake likelihoods. Of course, the best strategy is to include all worldwide earthquakes recorded over a long period of time. While increasing the database promises to improve estimation accuracy, the trade-off is that the processing time of searching among a large database increases significantly due to the rise in comparison operations. A simple search of the Advanced National Seismic System (ANSS) Composite Catalog (http://www.quake.geo.berkeley.edu/anss/) reveals that 2090 shallow crustal earthquakes (depth <30km) over magnitude 2 occurred in California during 2015. Similar results are also indicated with searches on USGS/ComCat, Southern California Earthquake Data Center and other similar earthquake databases. If one wants to include all records from the network over years for all the earthquake events worldwide, the size of the database scales exponentially (Yu, 2016). As a result, the processing delay of the real-time search will significantly increase because the time required to query databases sequentially is proportional to the size of the database. While advances have been made in the development of such algorithms in EEW, very little attention has been paid to optimizing the processing time of large databases.

Database searching is often an application of the Nearest Neighbor (NN) search problem with the Euclidean metric. The problem is commonly encountered in many computational techniques such as event detection, pattern recognition, and data analysis (Bhatia, 2010). In general, we seek for a point in the database that minimizes the Euclidean distance to the target point (sometimes referred as the least square distance). The problem states that: for the target point $x = [x_{(1)}, \dots, x_{(d)}]$ and the i^th^ training point in the database $y_i = [y_{i(1)}, \dots, y_{i(d)}]$, we define the distance between x and $y_i$ to be

$$d(x, y_i) = \left( \sum_{k=1}^{d} \left( x_{(k)} - y_{i(k)} \right)^2 \right)^{1/2} \qquad [1]$$

NN searches for the $\hat{y}$ with the closest distance to the target point, mathematically represented as $\hat{y} = argmin_{y_i}\left(d(x, y_i)\right)$. In most cases, the k-Nearest-Neighbor (k-NN) search method is applied by finding the k closest training points to the target point; this method provides a more robust estimation that avoids outliers in the database. The corresponding parameters associated with the $\hat{y}$ are used to classify or estimate the parameters of interest for the target point.

In this study, we use a data structure, multidimensional binary search tree (KD tree) NN searching concept, to organize the seismic data, and evaluate the reduction of NN searching time for large datasets. KD-tree is a binary tree data structure that links the relative position of all the data points, so data with similar patterns cluster, thereby allowing the search procedure to become faster (Bentley J. L., 1975). Although it requires initial effort to construct the tree data structure, the searching process is quick. The goal is to introduce the concept of data structures in EEW to minimize the processing time for waveform record searching without loss of accuracy, and thereby earthquake alerts can be delivered to the sites of interest much earlier. The effectiveness of fast alerts is especially valuable in the proximity of the epicenter where the strongest damage occurs very quickly after event onset. In this study, we describe a searching procedure that uses the KD tree NN search method that identify the EEW fingerprints characterized by the Gutenberg Algorithm. We 1) evaluate the influence of database size on the prediction accuracy of the earthquake source parameters (magnitude and hypocenter distance) and peak ground motion parameters (PGA, PGV, PGD), 2) estimate the processing efficiency of the KD tree searching for databases with different sizes and extrapolate the future performance by scaling to larger data sets. The KD tree is well-established NN searching algorithm that has been implemented in a wide range of engineering and database applications (Bentley J. , 1979). Although the method parallel processing can reduce real-time latencies, the cost of allocating additional resources could be unfeasible in long term. Only by efficiently design the computational algorithms to optimize the processing time can EEW start to adopt the databases for real-time seismology applications, and the fingerprint searching algorithms with big data reveal their full practical potential.

## Data

Theoretical analysis of the KD tree searching shows the performance complexity being O(log N) verses O(N) for the linear sequential search, where N is the number of data points in the database (Friedman et al., 1977). Although the theoretical average search time of KD tree is much shorter than the linear sequential search, the performance varies depending on the distribution of the data. Our goal is to determine the searching efficiency of the KD tree method for our GbA seismic database. We ran a series of offline tests on the earthquake filterbank database to mimic potential performance of EEW using true seismic records. The dataset used is pre-processed by (Meier et al., 2015) for the GbA. The database consists of 182,805 near-site records with 9 feature dimensions in each record. Each of the feature dimensions represents the peak ground velocity in octave-wide frequency bands for a given ground motion record with a fixed time window. The frequency bands used in GbA features are shown in Table 1. GbA creates such a dataset table for every half-second increment in time after the P-wave arrival. In general, EEW tends to consider at least 3 to 4 sec data after the P-wave arrival for the trade-off of accuracy and time delay. For the purpose of this investigation we selected the database for a 10-sec time window because the predictions are stabilize with more data collection.

The collected earthquakes cover a large range of magnitude, spanning from M 2.0 to M 8.0, compiled from shallow crustal earthquakes collected from Japan, Southern California, and Next Generation Attenuation-West 1 (Chiou and Youngs, 2008).

| Feature Dimension No. | Frequency Band (Hz) |
|---|---|
| 1 | 0.09375 – 0.1875 |
| 2 | 0.1875 – 0.375 |
| 3 | 0.375 – 0.75 |
| 4 | 0.75 – 1.5 |
| 5 | 1.5 – 3 |
| 6 | 3 – 6 |
| 7 | 6 - 12 |
| 8 | 12 - 24 |
| 9 | 24 - 48 |

Table 1. Frequency bands for feature input in Gutenberg Algorithm. The GbA database consists of 9 feature dimensions. Each feature takes the observed peak ground velocity in the given frequency band.

## **KD Tree and Method**

### KD Tree

KD tree is a binary tree structure that stores the finite set of database points with k-dimensional feature space. In our case, we have 9 variables corresponding to 9-dimensions. The method involves two steps. First, we construct the tree to organize the information in the database. Then, the NN algorithm is applied on the KD tree to search to the most similar point to the target record during an on-going earthquake. In KD tree implementation, a point in the database is also called a node in the tree.

- Construction of KD-tree

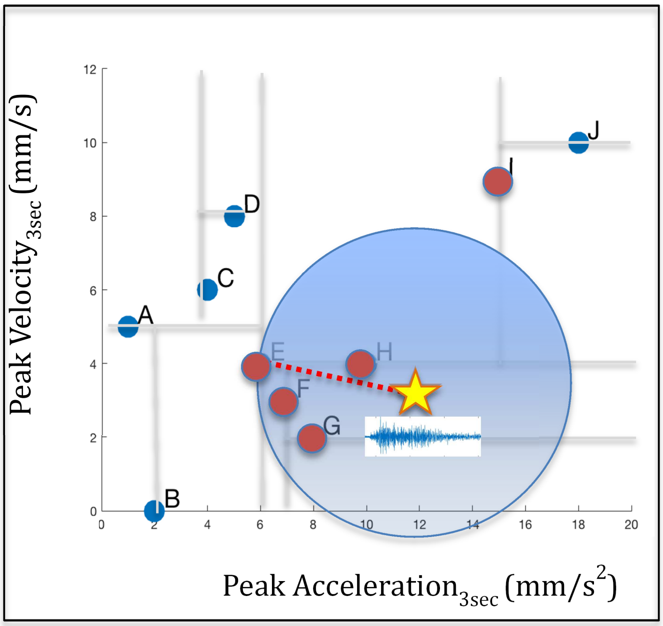
The construction of the KD-tree is a recursive process. Starting with the root of the tree, the first feature dimension (frequency band: 0.09375 – 0.1875Hz) is chosen as the splitting hyperplane. All nodes are ordered with respect to the value in this feature dimension, and the node with the median value is inserted into the root of the tree. All nodes with coordinates less than the median in the splitting hyperplane create the left subtree, and the nodes with coordinates larger than the median in the splitting hyperplane create the right subtree. All the feature dimensions rotate in becoming the splitting hyperplane to create the next level of subtrees.

- <u>Nearest Neighbor Search in KD-tree</u>

Starting with the root node of the tree, the nearest distance is initialized to be the distance between the target node to the root. Then recursively move down to the next level in the tree, and checks if the splitting hyperplane intersects with the hypersphere centered at the target record with a radius of the current nearest distance. If the node falls outside of the hypersphere created by the current nearest node (indicating the point is further to the target node than the current nearest node), then this node and any extended child nodes further away can be eliminated from the investigation. The process is repeated, recursively moving down to the next level in the tree until reaching the leaves of the tree. The searching time is reduced since large subsets of the database are not visited. Therefore, the average searching time in a KD tree is significantly lower, especially when the size of the database is large.

To better visualize the concept, Figure (1) demonstrates a KD tree structure for a 2-dimensinal featured database with 10 earthquake records described by the peak velocity and acceleration at initial 3 sec after triggering a station. The goal is to predict magnitude of the new event based on the velocity and acceleration recorded at the first 3 sec of the p-wave. We start the search process of the nearest neighbor of the target data (the yellow star) with node E, which is the root of the tree. The radius of the initial hypersphere is set between the target data and node E. In a 2D feature space, the hypersphere is simply a circle. Since the left branch (link between node A and E) does not cross the hypersphere, indicating all the nodes in the left subtree (node C, A, B, D) can be eliminated from the search because their Euclidean distance to the target point is clearly further than node E. This eliminates the computational effort of going through almost half of the database at the first step. Since the target node is closest to node H, the magnitude associated with node H (M=4.0) is the prediction result for the target node.

a)




b)

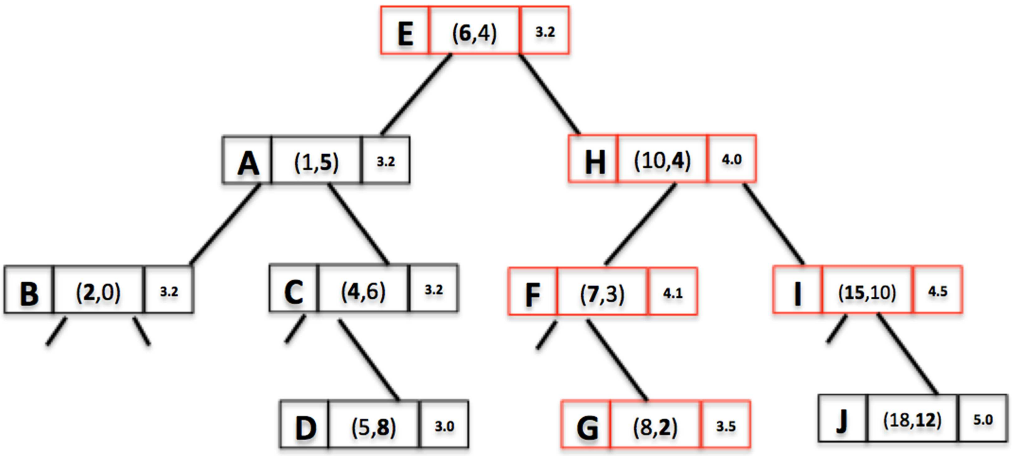



Figure 1: A 2-dimensional KD tree example: a) visual distribution of the database in feature dimensions, b) tree structure of the database. A database of 10 earthquake records (A - J) is organized using KD tree data structure (grey lines are the branches of the tree). As a comparison, the linear sequential search requires going through all 10 records, which doubles the computation effort.

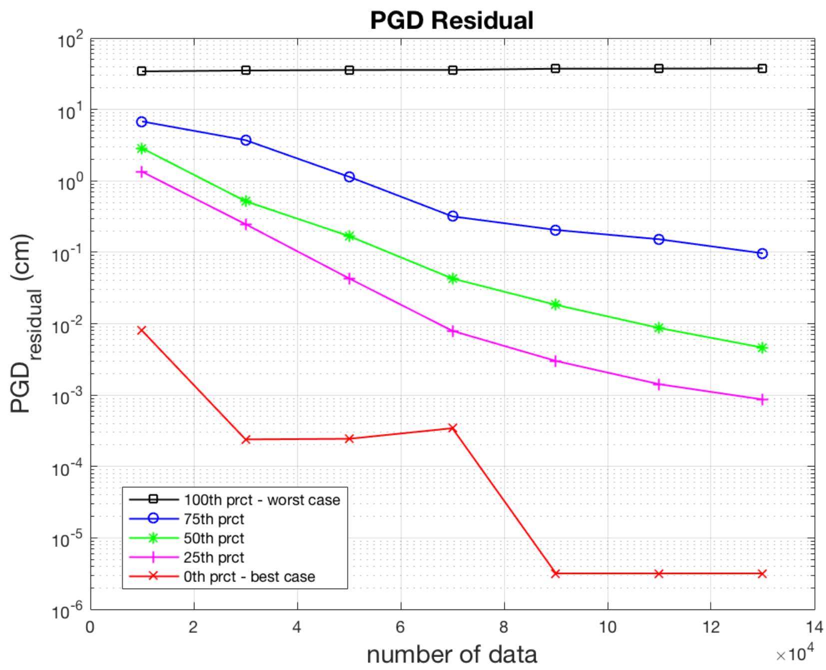
The algorithm can be easily extended to k nearest neighbor (k-NN) search to find k most similar points to the target point in order to give a more probabilistic estimate of target parameters. It requires two modifications. First, we need to keep track of all the current nearest points in an ordered queue with length k; if the queue contains fewer than k points, the subtrees on both sides need to be visited. Second, instead of comparing the splitting hyperplane with the hypersphere of the nearest point, we should check if the hyperplane intersects with the hypersphere of the last nearest point in the queue. If they intersect, the new node is inserted into the queue of k-nearest neighbors to the target point. At the end of the search, the algorithm returns k points from the database that are located with minimum distances to the target point.

Method

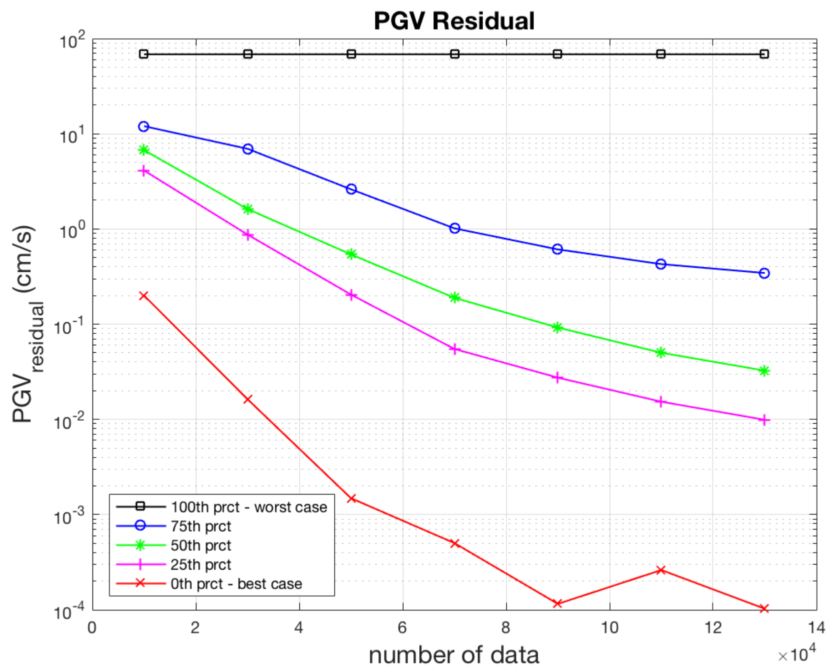
Since one of the ultimate goals of EEW aims to predict ground shaking, we extracted 500 records from the entire database to validate the prediction of earthquake source and ground motion parameters. The validation set was sampled uniformly with even spacing on the Peak Ground Acceleration (PGA) of the records. The reason is to cover the full spectrum of ground shaking intensity, in order to mimic all circumstances that could be encountered in the future. The performance of parameter estimations is evaluated with different dataset sizes. The estimated seismic parameters include station-specific ground motions: Peak Ground Acceleration (PGA), Peak Ground Velocity (PGV), Peak Ground Displacement (PGD), and earthquake source parameters: magnitude, hypocenter distance. The procedure first requires a 30-NN search in the Euclidean distance defined in Eq (1), and then a prediction using the Gaussian mean of the corresponding parameters from the 30-NN matched records. The value 30 is chosen to match the original model parameter in the GbA. Later, we compared the searching time of the KD-tree search to the Linear Sequential search, in both the CPU time and the number of operations.

## Results

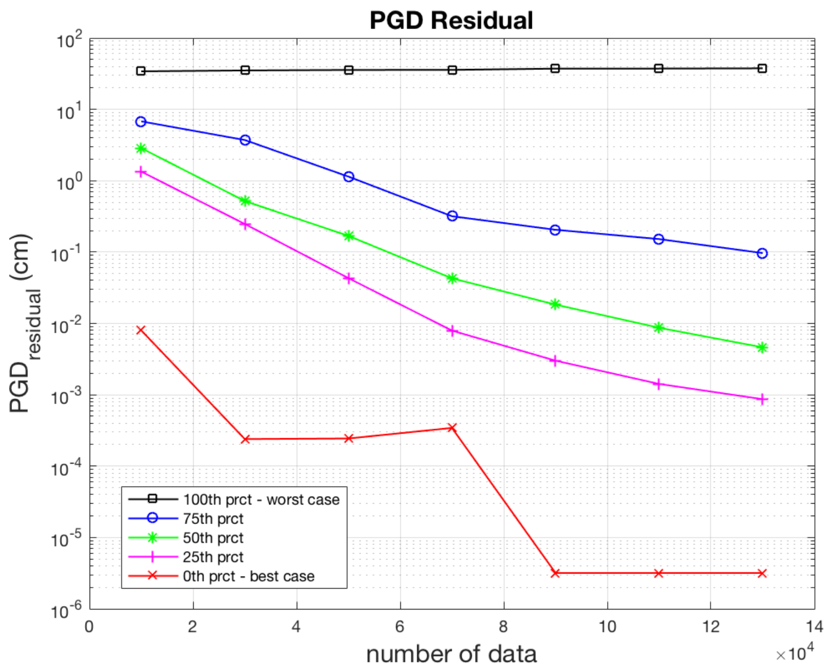
We computed the earthquake parameter estimation error of the validation set for databases with different sizes. Figure 2 shows the 100th, 75th, 50th, 25th, and 0th percentile residual errors for the estimated PGA, PGV, and PGD of the 500-validation dataset, respectively. The residual error is defined as the absolute difference between the true observed parameter and the predicted parameter. The 50th percentile is the average residual errors; the 100th and 0th percentile indicate the maximum error and minimum error, respectively. As expected, the residual error decreases as the database size increases on average. The 50th percentile is not flattened near the largest given database size showing that the residual errors might not yet reached the global minimum; this suggests that the estimation accuracy could further be improved by increasing the size of the database. Since there is always a possibility of outlier data regardless how large the database gets, the maximum error residuals are not affected by the size of databases as shown in the 100th percentile line in Figure 2. Statistically, there will always be residuals on the predictions, unless the features are truly uniquely diagnostic. Of course, if a sufficiently large database were compiled, the probability of encountering outliers would decrease.


PGD Residual
$PGD_{residual}$ (cm)
number of data
$\times 10^4$
100th prct - worst case
75th prct
50th prct
25th prct
0th prct - best case

a)


PGV Residual
$PGV_{residual}$ (cm/s)
number of data
$\times 10^4$
100th prct - worst case
75th prct
50th prct
25th prct
0th prct - best case

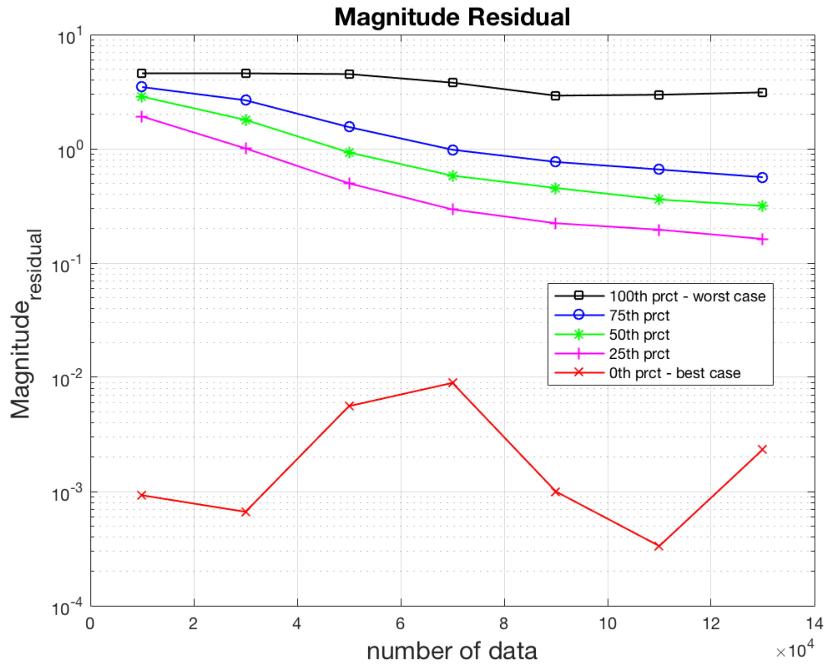
b)



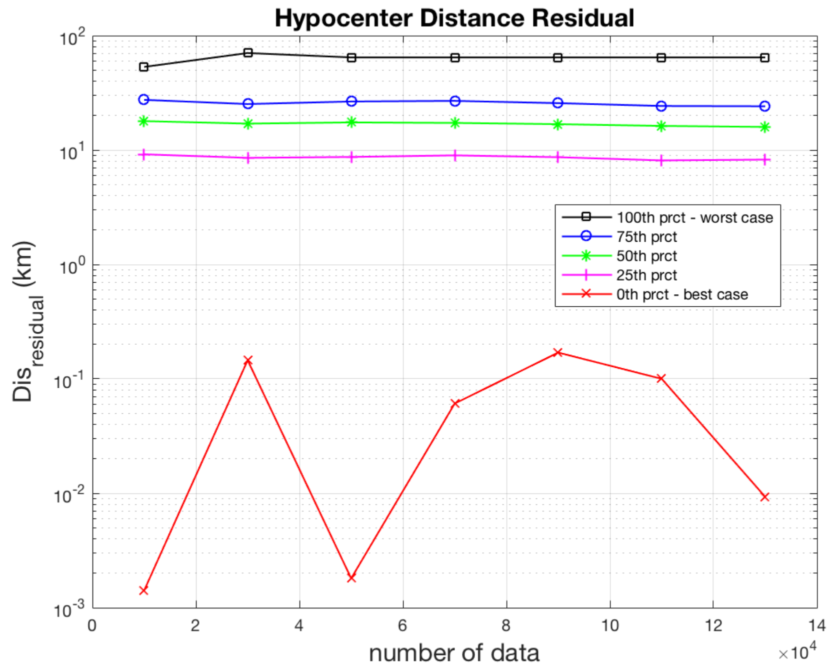

c)

Figure 2. Ground motion residuals for the 500-validation dataset with different database sizes. a) Peak Ground Acceleration, b) Peak Ground Velocity, c) Peak Ground Displacement residuals are given in absolute ground motion units. The lines show the percentile according to the legend. The 50$^{th}$ percentile is the average residual error; the 100$^{th}$ and 0$^{th}$ percentiles indicate the maximum and minimum errors respectively.

We also estimated the earthquake source parameters using the databases: magnitude and hypocenter distance. Although ground motion parameters are more useful outputs for EEW alerts, predicting source parameters is the conventional approach in real-time seismology (Minson et al., 2017). Figure 3 shows that the size of the database has less impact on hypocenter distance than magnitude estimation. Since hypocenter distance predictions from the observed waveform are a result of source energy and soil properties, the additional constraints might be necessary. For example, seismicity location forecast could be introduced as prior knowledge to reduce the uncertainties in earthquake location estimation (Yin et al., 2017). This analysis implies that it is essential to select data features intelligently to characterize the parameters we are aiming to predict. Frequency band features might be more suitable to predict the ground motions than source parameters, since local site effects may be implicitly being accounted for.




a)




b)

Figure 3: Source parameter residual for the 500-validation dataset with different database size. a) Magnitude, b) hypocenter distance residuals are given in absolute units. The lines show the percentile according to the legend. The 50th percentile is the average residual error; the 100th and 0th percentiles indicate the maximum and minimum errors respectively.

Through the performance analysis for databases with different sizes, we conclude that large databases can help to provide more accurate ground motion estimations for EEW. Next, we compare the computational time difference for the 30-NN search using the KD tree methods for each validation test. The implementation is in Matlab. For comparison, a Linear Sequential search method is also implemented as a base case. The Matlab function follows the pseudo code concept from the Appendix with optimization modules that efficiently process the data. In Figure 4, the solid lines show that the average CPU search time of a database with 130, 000 points is about 0.2 sec for the Linear Sequential search method and 0.03 sec for the KD tree search method; the significant reduction in time reduces computational effort by 85%. Although the Linear Sequential search is capable of handling the real-time processing with limited delay using the current size of the database, a significant delay would be introduced as the database size rapidly increases in the future. The dashed lines show extrapolated computational time up to double of the current database size. The results anticipate that the advantages of the KD tree application would be emphasized in the future as global seismic databases are growing significantly (Yu, 2016).

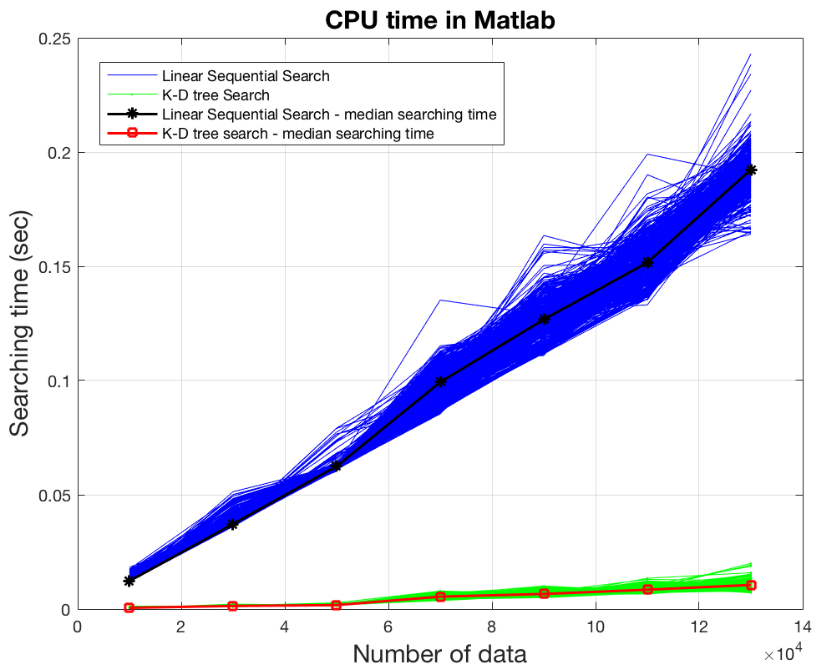



Figure 4: CPU searching time for different database sizes using linear sequential search and KD tree search. The implementation is in Matlab.

The measured operational time for the searching process varies significantly between different software languages and implementations; different optimization modules with parallelization might also bias towards one method over another. Implementations in C++ tend to be much faster than Matlab. In order to compare the true efficiency of the method across all platforms, we further compared the number of data points visited for both NN search algorithms. Since the majority of the searching time is made up by the visit to each data point to compute the Euclidean distance to the target point, the fewer data points visited ensures less time effort. In the Linear Sequential search, the operation is required for all the data in the database in a serial manner. However, in KD tree, subsections of the database can be eliminated depending on the distribution of the tree structure and location of the target point. As shown in Figure 5, the number of data points visited in the KD tree for each validation varies; on average, the KD tree approach only visits about 10% of the entire database to find the closest data point to the target, confirming the performance in CPU searching time in Matlab. In the worst-case scenario, all the data points are visited, which leads to the same operational complexity as the exhaustive approach (linear sequential search).

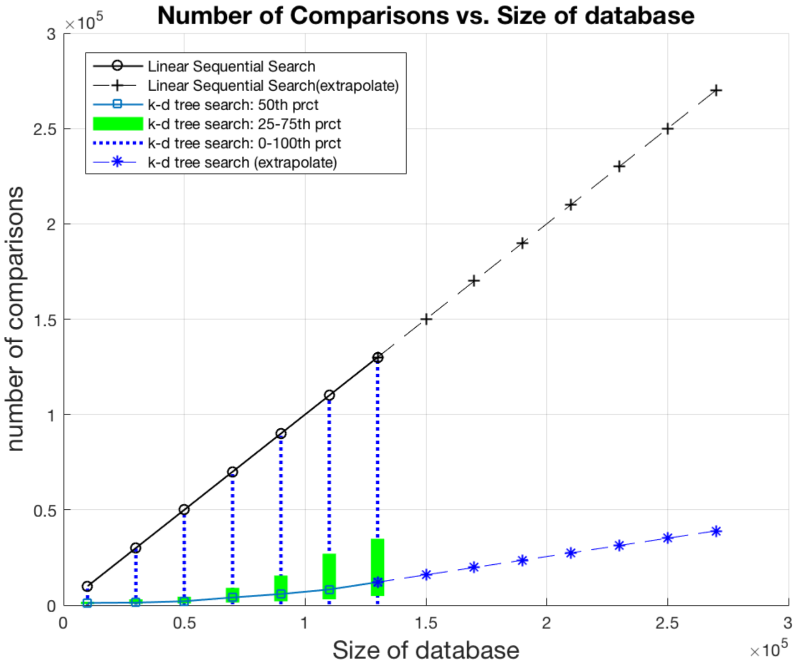



Figure 5 Number of data points visited for linear sequential search and KD tree search. The dashed lines are extrapolated to estimate the performance for larger database in the future.

**Discussion and Conclusion**

In this study, we evaluated the viability of earthquake fingerprint searching methods for EEW, using database structure to reduce searching time for large databases. Specifically, we evaluated the GbA as an example of the EEW fingerprint search algorithm. We found that database size is a critical factor in providing reliable predictions of ground motion (PGA, PGV, PGD) and source parameters (magnitude and hypocenter distance) for EEW. We also present the KD tree approach to reduce the searching time, so that large database searching is feasible for real-time implementations in EEW. By empirical validation, we demonstrated that the searching time using KD tree can be approximately 85% less than the exhaustive approach for the GbA EEW earthquake database. (Strauss et al, 2017) has studied extensively on the cost-benefit effects of a warning system in the United States; the study has shown that the number of injuries from earthquakes can be reduced by more than 50% if EEW can provide timely and accurate alerts.

One of the potential applications of the database searching method is to directly estimate peak ground motions from the observed ground motions for any given site in real-time seismology application such as EEW; it avoids the multi-step modeling errors that could be accumulated through source parameter estimation and the ground motion attenuation relationship, since the final errors can lead to significant uncertainties in the final shaking information. Ideally, the goal of EEW is to serve as an alarm for severe ground shaking in real-time rather than source characterization. The fingerprint searching methodology could also be extended to tackle other challenges in EEW, such as event detection (i.e. earthquake/noise discrimination). In such a problem, characteristics of additional ambient noise and teleseismic records need to be incorporated in the database. This would vastly increase the database size, since incorporating many different types of noise, teleseisms, regional events, and calibration/maintenance signals could potentially be huge. The vision is to be able to accomplish efficient searching for large databases, so that these novel EEW methods are feasible in real-time in the future.

Although we emphasized the importance of having a large number of data, a question is often raised about what should be the minimum size of database in order to get reasonable accurate solutions. Assuming the standard deviation of $\log_{10}$(PGV) estimation of 0.309 by (Kanamori, 2007) is acceptable, the database size needed to achieve this marginal error of ground motion in EEW is about 70 000 to 100 000 data points, as shown in Figure 2b). The (Kanamori, 2007) study focuses on two EEW parameters, $\tau_c$ and $P_d$, that are extensively used in the existing EEW algorithms, such as Onsite (Bose et al., 2009). The minimum database size calculated varies with geological region, event types, predictive parameters, etc.

Creating a database for real-time seismology is not simple. In addition to the sizes of databases, feature engineering also significantly affects the prediction results. Selecting parameters that correlate to the predictive results requires extensive scientific domain knowledge. In the observation of local earthquake records, the higher frequency band features are more informative than the low frequency features because the high frequency amplitude of ground motion decays rapidly with distance (Hanks & McGuire, 1981) (Kong & Zhao, 2012). Although it is out of the scope of this study, we plan to further investigate in the effects of using a weighted Euclidean distance in the Nearest Neighbor Search to emphasize the high frequency information as a significant attribute in the feature space. Continuous monitoring and modifying of the features will help to improve the performance of the system. As the number of features increases, the process time saved by KD tree search decreases (Andoni & Indyk, 2008). For features over 20 or 30 dimensions, alternative approximation to approach high dimensional searching, such as Locality Sensitive Hashing, would be more appropriate [e.g. (Yoon et al., 2015)].

EEW is an interdisciplinary project that involves collaboration among different scientific and engineering communities. The accuracy and speed of rapid earthquake source parameter algorithms has significantly improved over the past decade, but are potentially limited by the simplification involved in model parameterization. The earthquake fingerprint searching techniques have the capacity to guide the development of EEW to a new phase with the assistance of better computational power and data mining techniques.

**Acknowledgement**

We would like to thank Dr Men-Andrin Meier for providing the preprocessed seismic database from GbA used in this study (obtained on September 2016), including Japanese waveforms data obtained from http://www.kik.bosai.go.jp, The Next Generation Attenuation – West 1 data obtained from http://peer.berkeley.edu, and southern California waveform from Southern California Earthquake data center obtained from http://www.scedc.caltech.edu. All of the simulations and figures are generated with Matlab 2016a. This research was supported by the Gordon and Betty Moore Foundation grants 3023 and 5229, and USGS/NEHRP Cooperative agreement G16AC00355. This research was also supported by the Natural Sciences and Engineering Research Council of Canada's (NSERC) Postgraduate Scholarships-Doctoral Program.

## Bibliography

Allen, R., & Kanamori, H. (2003). The potential for Earthquake Early Warning in Southern California. *Science , 300*, 786-789.

Allen, R., Brown, H., Hellweg, M., Khainovski, O., & Lombard, P. (2009). Real-time earthquake detection and hazard assssment by ElarmS across California. *36*, 2009.

Andoni, A., & Indyk, P. (2008). Near-Optimal Hasing Algorithms for Approximate Nearest Neighbor in High Dimensions. *Communications of the ACM , 51* (1), 117-122.

Böse, M., Heaton, T. H., & Hauksson, E. (2012). Real-Time Finite Fault Rupture Detector for large earthquakes. *Geophysical Journal International , 191*, 803-812.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Comm. ACM , 18*, 509-517.

Bentley, J. (1979). Multidimensional Binary Search Trees in Database Applications. *IEEE Transactions on Software Engineering , SE-5* (4), 333-340.

Bhatia, N. (2010). Survey of Nearest Neighbor Techniques . *International Hournal of Computer Science and Information Security , 8* (2), 302-305.

Bose, M., Hauksson, E., Solanki, K., Kanamori, H., & Heaton, T. (2009). Real-time testing of the on-site warning algorithm in southern California and its performance during the July 29 2008 Mw 5.4 Chino Hills earthquake. *Geophysical Research Letters , 36*, L00B03.

Cua, G. (2005). *Creating the virtual seismologist: developments in earthquake early warning and ground motion characterization.* Ph.D. Thesis, California Institute of Technology, Department of Civil Engineering.

Friedman, J., Bentley, J., & Finkel, R. (1977). An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Methmatical Software , 3* (3), 209-226.

Gutenberg, B., & Richter, C. (1944). Frequency of earthquakes in California. *Bulletin of the Seismological Society of America , 4*, 185-188.

Hanks, T., & McGuire, R. (1981). The Character of high-frequency strong ground motion. *Bulletin of the Seismological Society of America , 71* (6), 2071-2095.

Heaton, T. H. (1985). A Model for a Seismic Computerized Alert Network. *Science , 228* (4702), 987-990.

Hoshiba, M. (2013). Real-time prediction of ground motion by Kirchhoff-Fresnel boundary integral equation method: Extended front detection method for earthquake early warning . *J. Geophys. Res , 118* (3), 1038-1050.

Kong, Q., & Zhao, M. (2012). Evaluation of Earthquake Signal Characteristics for Early Warning. *Earthquake Engineering and Engineering Vibration , 11*, 435-443.

Kuyuk, H., & Allen, R. (2014). Designing a Network-Based Earthquake Early Warning Algorithm for California: ElarmS-2. *Bull. Seismo. Soc. Am.* (104), 162-173.

Meier, M., Heaton, T., & Clinton, J. (2015). The Gutenberg Algorithm: Evolutionary Bayesian Magnitude Estimates for Earthquake Early Warning with a Filter Bank. *Bulletin of the Seismological Society of America , 105* (5).

Meier, M. (2017) How 'Good' are Real-Time Ground Motion Predictions from Earthquake Early Warning Systems? *Journal of Geophysical Research* (122), 5561–5577.

Minson, S., Wu, S., Beck, J., & Heaton, T. (2017). Combining Multiple Earthquake Models in Readl Time for Earthquake Early Warning. *Bulletin of the Seismological Society of America* .

Strauss, J., Allen, R., (2017). Benefits and Costs of Earthquake Early Warning. Seismological Research Letters (87), 765 – 772.

Wu, Y.-M., Kanamori, H., Allen, R., & Hauksson, E. (2007). Determination of earthquake early warning parameters, tc and Pd, for southern California. *Geophys. J.Int.*

Yin, L., Meier, M., & Heaton, T. (2017). Making Earthquake Early Warning faster and more accurate using ETAS seismicity models as Bayesian Prior. *16th World Conference on Earthquake Engineering.*

Yoon, C., O'Reilly, O., Bergen, K., & Beroza, G. (2015). Earthquake detection through computationally efficient similarity search. *Science Advances* , 1-13.

Yu, E. (2016). Products and Services Available from the Southern California Earthquake Data Center (SCEDC) and the Southern California Seismic Network (SCSN). *AGU Fall Meeting*, (pp. S53A-2817).

Zhang, J., Zhang H., Chen, E., Zheng, Y., Kuang, W., Zhang, X. (2014). Real-time earthquake monitoring using a search engine. *Nature Communications 5:5664*.

Zuccolo, E., Bozzoni, F., & Lai, C. (2016). Regional Low-Magnitude GMPE to Estimate Spectral Accelerations for Earthquake Early Warning Applications in Southern Italy. *Seismological Research Letters* .

**Appendix – Pseudo Code**

KD Tree Construction

```
Function construction_kdtree(points in database, depth)
        Split_axis=depth mod k_dim;
```

```
        Median = select median from fingerprints(split_axis)
        Leftdatabase={fingerprints in database| fingerprints(split_axis)<median}
        Rightdatabase={fingerprints in database| fingerprints(split_axis)>median}

        %create node
        node.location=median
        node.left= kdtree(leftdatabase,depth+1)
        node.right= kdtree(rightdatabase,depth+1)
```

<u>Searching in KD tree</u>

```
Function search_kdtree(target, node, nearest_dist)
Split_dim= split dimension at the depth of the tree
Hyperplane_dist=target(split_dim)-node(split_dim)

If nearest_dist > |hyperplane_dist| then
        curr_dist:= distance between target and curr_node
        If curr_dist<nearest_dist then
                nearest_dist:=curr_dist
                nearest_fingerprint=curr_node
        search_kdtree(target, curr_node.left, nearest_dist)
        search_kdtree(target, curr_node.right, nearest_dist)
else
        if hyperplane_dist<0 then
                search_kdtree(target, curr_node.left, nearest_dist)
        else
                search_kdtree(target, curr_node.right, nearest_dist)
```

**Highlights**

- Presented a multidimensional binary search (KD) tree database structure for seismic data
- Reduced average searching time by 85% for real-time seismology predictions
- Suggested to directly predict ground motion information for Earthquake Early Warning due to accuracy
- Evaluated pros and cons of modeling approach and big data search approach for real-time seismology