Unsupervised Machine Learning and K-Means Clustering as a Way of Discovering Anomalous Events In Continuous Seismic Time Series

by

Elezhan Zhakiya

B.S., Massachusetts Institute of Technology (2016)

Submitted to the Department of Earth, Atmospheric, and Planetary Sciences
in partial fulfillment of the requirements for the degree of

Master of Science in Earth, Atmospheric, and Planetary Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2018

Signature redacted

Author .....
Department of Earth, Atmospheric, and Planetary Sciences
January 12, 2018

Signature redacted

Certified by .....
Bradford Hager
Cecil and Ida Green Professor of Earth Sciences
Thesis Supervisor

Signature redacted

Accepted by .....
Robert D. Van der Hilst
Schlumberger Professor of Earth Sciences,
Head of the Department of Earth, Atmospheric, and Planetary Sciences

Unsupervised Machine Learning and K-Means Clustering as a Way of Discovering Anomalous Events In Continuous Seismic Time Series

by

Elezhan Zhakiya

Submitted to the Department of Earth, Atmospheric, and Planetary Sciences
on January 12, 2018, in partial fulfillment of the
requirements for the degree of
Master of Science in Earth, Atmospheric, and Planetary Sciences

Abstract

Unsupervised k-Means clustering was implemented as a method for identifying anomalies in seismic time series. Sliding window approach was used for generating specific subsequences from the overall waveform. Dynamic Time Warping (DTW) was used as the method for comparing sesimic subsequences. DTW barycenter averaging (DBA) was used as the method for averaging multiple subsequences within a group of similiar shapes. Clustering is able to discover anomalously shaped parts of a seismic time series in a completely unsupervised fashion, without requiring anyone to input actual times of the events, any predetermiend examples of events, or any other parameters about the signal.

Thesis Supervisor: Bradford Hager
Title: Cecil and Ida Green Professor of Earth Sciences

Acknowledgments

Thank you to ERL.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Seismic waveforms are continuously recorded time series made by seismographs. Events like earthquakes are recorded in corresponding parts of the time series as subsequences of certain shapes (or rather a specific set of shapes). More importantly, these shapes carry with them the information about not just the earthquakes themselves, but also of everything encountered on the path between the source of the event and the seismic receiver. Sudden appearance of geologically interesting objects on these should thus result in anomalously shaped subsequences.

Discovery of these anomalous shapes can thus help with identifying all kinds of curious objects at or near the source. Thus identification of anomalies within seismic time series is of interest to us.

However, seismic waveforms are typically continuous recordings of months long data points, while the anomalous events of interest can occur on a one second time scale. This makes finding anomalies manually prohibitively difficult. Searching for an anomaly from a month long dataset can take upwards of several weeks or months of work. Additionally, there is no guarantee that the visual search for anomalies would yield all of the anomalies. We thus want a way of finding anomalies automatically.

In a continuous time series anomalies can occur at any point. Typically, seismic recordings show numerous events separated by lengthy stretches of noise. Since seismic events do not necessarily come at discrete known intervals, the selection of subsequences for anomaly identification presents yet another challenge: subsequences

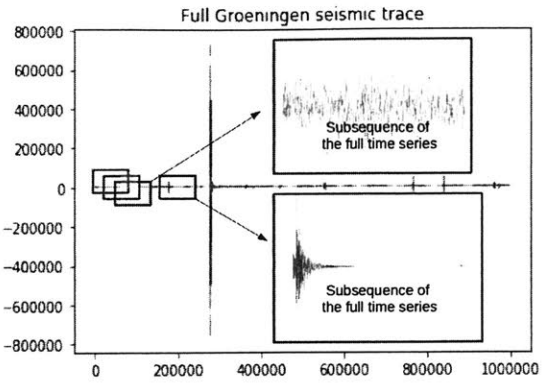

Figure 1-1: Seismic trace was broken into subsequences

must be generated from the time series on a continuous basis, with nearly each point selected as a starting point for a subsequence.

Subsequences were extracted using the sliding window method, which we explain in Section 2.1. The sliding window method breaks up the full time series into many subsequences of a fixed length, as shown in Figure 1-1

Anomalies, however, do not necessarily look like specific shapes within the waveform. We frequently do not have a predetermined notion of what exactly the anomaly is prior to studying it. Thus we cannot formulate an idea of a shape we would be looking for in a time series. Instead, we must formulate the notion of anomaly as signal or signals that look different from other signals commonly observed in the time series. Thus, we are challenged with the task of detecting events and anomalies without using simple template matching. Furthermore, we should opt out from using statistics about the signal as basis for anomaly identification.

Machine learning is an effective tool for creating programs capable of automatically recognizing complex patterns in data and extracting knowledge contained within those patterns. Machine learning relies on the use of datasets of input signals that collectively effectively summarize the process we would like the machine to learn. Machine learning is currently an expanding field of research, and is currently experiencing rapid adoption in the driverless cars industry, finance industry, and insurance industry.

Machine learning is divided into two categories: supervised and unsupervised.

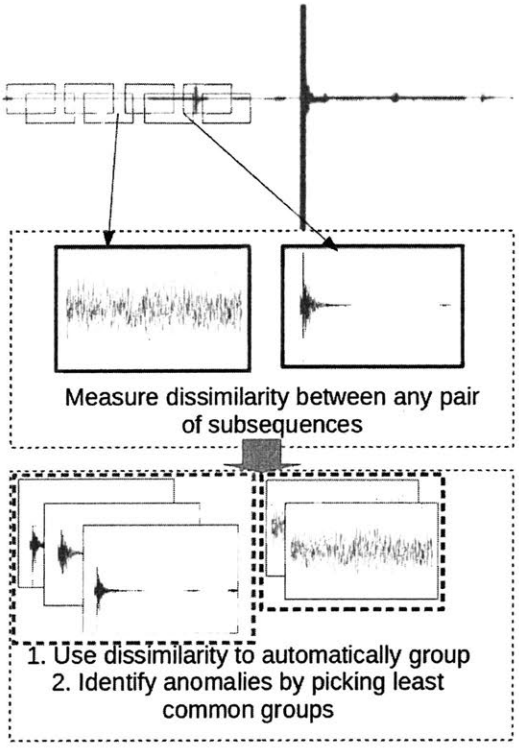

Figure 1-2: Proposed process for identifying anomalous chunks within seismic time series

Supervised machine learning methods require the use of known outcomes or labels to each input signal. Unsupervised methods only need datasets of input signals to function. Since we want to find one second-scale anomalies in continuous months long datasets, without any idea about the nature or form of these anomalies, and since we want to make the machine do it all on its own, we are left with unsupervised machine learning.

In the background section, we set up the framework behind the K-Means clustering algorithm - one of the main algorithms in unsupervised learning. Our plan for anomaly identification within time series is illustrated in Figure 1-2, and we give a detailed explanation of the process in Chapter 2.

We then talk about the need for using a special distance measure for subsequence data, known as Dynamic Time Warping (DTW). We then explain the need for using a specialized averging method for subsequences, called DTW Barrycenter Averaging

(DBA). We tested the effectiveness of DTW and DBA methods on our subsequences, and the result are presented in the methods section.

We then ran K-Means algorithm under fully-supervised, semi-supervised, and completely-unsupervised settings on our synthetic and real datasets. Our results are presented for each dataset in the results section. Finally, we make some recommendations for future work and areas of improvement in the conclusions section.

Chapter 2

Background

2.1 Finding Anomalous Parts out of a Seismic Waveform

Our data of interest was a continuous one month long seismic waveform. The time series consisted of multiple recorded events and noise. Some of these events were considered anomalous. Our task was to identify parts of this seismic waveform that contains anomalous events without relying on the expertise of a geophysicist.

The waveform was recorded at a 100Hz sampling rate, meaning that each second of the recording translated into 200 data points. Full one month of the waveform translated into 520M data points. Sub sampling the data was not preferred since many important features were present in the data at the highest frequencies. The pure volume of data was a significant challenge to us. Run time of our possible algorithms was one of our biggest considerations in our search for methods.

Example were generated using a sliding window approach - first subsequence was made up of the time series elements starting from the first and ending on element N, second subsequence starting from the second element and ending on element number N + 2, with N being the length of subsequences. Each unique element or short sequence of elements in the time series is thus contained within multiple subsequences in every possible position, but with the original order intact.

Next challenge was associated with anomaly detection. Our task was finding parts of the seismic time series that looked anomalous - but the exact character, shape, or form of these anomalies could not be defined ahead of time.

2.2 Machine Learning

The final goal of a machine learning algorithm is to either classify or asses a certain property about an object based on some input information about the object. Unlike traditional programming, however, machine learning achieves that without needing the programmer to include in his or her code a predefined model of the process. Instead, machine learning learns this model of the process completely on its own. For example, a traditional program for making decisions on the stock market would require the creator to encode some equations fundamental to finance, which the program would then use to make its decisions. In contrast, a machine learning algorithm would not need such an equation to be encoded in order for it to be successful at its task. Similarly, a driver less car running on machine learning does not need a team of programmers to encode a set of instructions on what to do when a car sees a pedestrian or red light. Machine learning achieves these things by training on the data.

A machine learning algorithm trains itself by looking at a dataset of examples. Each example is a set of inputs, and collectively the total set of all possible examples should encapsulate the entire process. There is always an outcome, either a class or a property, associated with each example.

If this class or property is available and is provided to the algorithm during training, then the algorithm is said to be a supervised machine learning algorithm. If the class or property is not provided, then the algorith is said to be an unsupervised machine learning algorithm. It is important to note that in an unsupervised setting, the outcome still exists and its correct determination is still the goal.

A supervised machine learning algorithm learns to correctly predict the classes or properties of objects by reducing its error in assigning outcomes to examples within

the training dataset. For every example in the training dataset, the algorithm looks at the input of that object, and assigns a hypothetical outcome to the object. This hypothetical outcome is compared to the real outcome, and the error is measured. The algorithm then changes its own parameters, usually guided by a heuristic based on the magnitude and the sign of the error. Since error minimization is its goal, over time the algorithm begins to assign hypothetical outcomes that are less and less different from the real outcomes. Thus by aiming to reduce the error, machine learning learns whatever it needs to make the correct prediction - and it learns any pattern present in the input that could help it reduce its error.

Using supervised machine learning for our task would require us to have a dataset of examples drawn from the time series, and would require each example to be labeled as noise, regular event, or an anomalous event. However, these anomalous events need not necessarily look like each other - they also do not need to be anomalous in the same way. We do not know what these anomalies should look like ahead of time. We could create a dataset manually by labeling every part as anomalous or not, but that would then limit us to discovering anomalies only of the same type as the ones in the original dataset. This method does not work for identifying new types of anomalies not encountered in the training dataset. Thus we would be limiting ourselves to only certain well-known anomalies, which is an undesired result. We need a way of recognizing all kinds of anomalies, not just ones frequently encountered previously.

2.3 Unsupervised Learning

Unsupervised learning draws patterns from data consisting solely of the input signal without the class or output property. It achieves that usually by grouping input examples that are similar together. Once all examples have been grouped based on their inputs, we can start to assign outcomes to new input examples based on the group they would most likely belong to. This process is also known as clustering.

Different variants of clustering algorithms include hierarchical clustering, partition clustering, and k-means clustering.

2.3.1 K-Means Clustering

The goal of k-Means clustering is to find an optimal assignment of subsequences to K clusters, such that for any given subsequence, its distance measure to the centroid of its current cluster is smaller than its distance measure to the centroid of any other cluster.

Once k-Means has been used to put the sub-sequences into clusters, the algorithm can classify a new seismic waveform subsequence as an anomaly/event/noise by determining which cluster centroids (representatives) matches the new waveform the closest based on our chosen similarity measure, dynamic time warping (DTW). In order to extract the cluster from our data and then classify a new subsequence, k-means must rely on a similarity measure. Here, we use the terms similarity measure and distance measure interchangeably to describe the overall shape difference between sub-sequences. For the lack of ambiguity, similarity measure is the same as distance measure, and is small for similar shapes and large for dissimilar shapes.

Note that a reliable similarity measure is the only thing that k-means needs to cluster the data - subsequences need not be pre- labeled for us to extract a meaningful pattern. This is highly advantageous since it lets us detect anomalous subsequences in the data without the need for per-determining what an ideal anomaly should look like, as discussed earlier.

K-Means works iteratively in two steps Initialization: Sample k random subsequences from the data to initialize k clusters. Step 1) K-Means iterates through every subsequence and re-assigns them to best-match clusters based on a distance measure between the subsequence and the centroid of the cluster. Step 2) After each subsequence has been re-assigned, the algorithm re-computes the centroid of each cluster.

The algorithm converges when subsequences no longer change their cluster membership. At this point all subsequences are placed in their best-match clusters.

Note that to determine the goodness-of-fit between a sub-sequence and a cluster, k-Means needs to measure the DTW between our queried subsequence and the centroid

of the cluster.

If Euclidean Distance (ED) is was our similarity metric of choice, then by default finding the element-wise mean of all the sub-sequences in a cluster would give us the centroid of that cluster that minimizes the ED to all other sub-sequences in that cluster.

Since we are using DTW, however, the element-wise mean of the sub-sequence is no longer guaranteed to be the centroid that would minimize the DTW distance to all other sub-sequences in the cluster. Indeed, if two sinusoidal time sequences were out of phase exactly, their element-wise mean would be a straight line at zero - which clearly does not represent either of the sequences.

K-Means clustering is a greedy algorithm - greedy algorithms may tend to a local optima rather than global optima. Depending on our initial conditions we might not be able to get to the right answer. The problem can be solved by adding a stochastic framework. We could use either a genetic algorithm (GA) or simulated annealing (SA). [Pakhira 2009] To implement GA or SA, each cluster would be parametrized using two quantities: the cluster centroid and the cluster variance, based on the sum of sqaure distances of each sub-sequence to the centroid. Then, we would assume the sub-sequences in a cluster to be normally distributed around the cluster centroid. Then, we would calculate the likelihood of any given sub-sequence belonging to any given cluster using the normal z-score likelihood, with the z-score being its distance from the centroid divided by the square root of the variance of the cluster. For each sub-sequence, a likelihood would be calculated for every cluster, and all likelihoods kept. The final cluster assignment of any given sub-sequence would then simply be the cluster with the highest likelihood value.

K-Means also suffers from an empty clusters problem. During step 1, it is possible that all subsequences within a cluster are re-assigned elsewhere, and no subsequence is assigned to the cluster. As a result, the cluster is left empty - and thus cannot participate in future iterations, since it has no centroid for subsequences to be compared to. There are several proposed solutions to the empty clusters problem. [Manoharan and Ganesh 2016] The proposed modified K-Means algorithm solves the issue of

empty clusters by changing the cluster centroid computation procedure. In the classical K-Means, centroid is calculated as the average of all members of the cluster. In the modified k-Means, centroid is calculated as the average of all current members of the cluster plus the old centroid of the cluster computed over the last step. In other words, cluster centroids from the previous iteration are considered to be a cluster member at the current step. If the cluster is empty, centroid is by default set to the same centroid as last time, since the average of a set consisting of a single element is the element itself. Manoharan and Ganesh show their modified K-Means algorithm converges to the same final answer as the classical k-Means algorithm.

2.4 Subsequence Time Series Clustering

Our goal is to identify the point of time in the time-series that is anomalous compared to the rest of the waveform Thus we are classifying sub-sequences, not entire time-series. We thus will a rolling window approach to extract the sub-sequences. Challenges with this task are: (1) that trivial matches due to the overlap resulting from the rolling window can occur. Rolling window approach guarantees that for every distinct event in the time-series, we will generate mutliple sub-seuences that contain that event, each in different spots. This can be problematic in cases where a sub-sequence contains the transition between two distinct signals. For a such subsequence it would be ambiguous which cluster it should belong to, and the distance measure (2) that the resulting dataset is large, computationally expensive. Our typical seismic waveforms contain millions of points, and thus present a significant performance challenge, when working with computationally expensive algorithms such as the k-means. (3) that the scale we will use here is of importance. Rolling window approach requries a set window length for subsequence extration. Thus anomalies at vastly smaller or larger scales than the window may not be fully identified using this approach. In this study, we limited ourselves to anomalies on a few seconds scale.

Definition 1: Time series is an ordered sequence of data-points Definition 2: A subsequence Z of a time series is a continuous portion of a time series T. A subsequence

Z of length m can be represented as a vector of m elements from T, arranged as [xi, xi+1,....xi+m-1] [Zolhavarieh 2014]

Biggest pitfall of the clustering algorithms is failure to obtain meaningful results: when the centroids of the final clusters all look similar to each other and do not truly represent clusters of truly different sub-sequences. The frequent result of trivial and undifferentiated cluster centroids caused some researchers to claim that clustering of time series subsequences is meaningless.[Keogh 2003] This problem arises when clustering results in cluster centroids that a) all look similar, and b) do not look anything like the subsequences contained in those clusters. Frequently these centroids could end up being simple sinusoidal shapes. When that happens, all centroids are identical and individual subsequences no longer change their cluster memberships. Though the algorithm technically converges, the result is a trivial equilibrium between identical generic centroids and subsequence that look nothing like the centroids, but yet stay in the cluster because all of the other cluster centroids are the same.

This problem arises due to algorithm converging upon simplistic identical centroids, which in turn results from the use of wrong methods for estimating centroids. Subsequence time series clustering researchers concluded that the solution to this problem is to use methods for determining the average member to resp sent a group that does so in accordance with the distance measure used. [Keogh 2013]

2.4.1 Dynamic Time Warping

Originally introduced in the speech recognition community, Dynamic Time Warping is a measure of similarity between temporal sequences of data robust to phase shifts and variations in speed associated with the sequences. DTW measures the distance between two time sequences along the shortest path that optimally matches the alignment between them. [Chiba and Sakoe 1978]

Traditionally used Euclidean Distance (ED) measures the distance between two sequences as the sum of squared difference between every Nth element in Z1 and the corresponding Nth element in Z2. Unlike ED, DTW does not require the squared distance to be calculated between every element in Z1 and the corresponding Nth

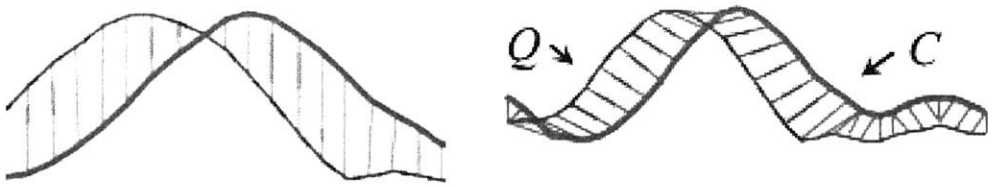

Figure 2-1: Comparison of DTW (Right) vs Eucledian Distance (Left) as distance measure between two subsequences [Ratanamahatana 2012]

element in Z2 - the difference is rather allowed to be calculated between each element in Z1 and its optimal match in Z2. The difference between ED and DTW can be visualized in Figure 2-1.

The resulting sum of those differences is the shortest distance remaining between Z1 and Z2 once the time-axis was allowed to be warped to minimize the total distance.

The basic steps of the DTW algorithm are: 1) Constructing an m-by-m matrix of element-wise distances, where the i,j entry of the matrix corresponds to a local cost measure between Xi in Z1 and Xj in Z2. Euclidean distance is commonly used as such local cost measure, though a variation that uses an absolute difference also exists.[Keogh 2002] 2) A warping path W is a set of matrix elements that defines a specific mapping between Z1 and Z2.[Keogh 2002] 3) The sum of all elements in W is the specific distance between two sequences measured along the warping path W 4) Pick the warping path W that gives shortest total sum of all elements it contains

Certain constraints must be placed on the DTW algorithm to ensure that the shortest path is a non-trivial answer. [Chiba and Sakoe 1978]:

1) Diagonality Constraint. Path W must start from (1,1) and end with (m,m). This means that DTW must account for the full lengths of the two sequences Z1 and Z2. Clearly, a shortest path can easily be found by simply only summing distances for a small part of Z1 or Z2. Such path results in a trivial result and does not provide a useful measure of similarity between sequences. Diagonality constraint is thus necessary to ensure that only the solutions accounting for the full lengths of Z1 and Z2 are identified. 2) Monotonicity constraint - indexes i and j of elements in Z1 and Z2 that lie on path W must monotonically increase. Once Jth element of Z2 is

included in path W, path can no longer contain any element of Z2 <J. In the m-by-m matrix, path W must propagate down and to the right, and never back up-wards and to the left. 3) Continuity constraint - all elements in Z1 and Z2 must be included, and the same pair of Xi and Xj cannot be used more than once.

These constraints can be illustrated in the Figure 2-2. Diagonality means the path must start in one corner and end up in the opposite corner. Monotonicity constraint means that the path must keep moving in the positive direction in both coordinates - it cannot suddenly turn back for a loop. In other words, the index of the elements being computed at every step must either increase or stay the same, but not decrease. Finally, continuity implies that it is not allowed to skip anything. For example, the path cannot suddenly go from the fifth element in the second sequence to the seventh element in the second sequence. This also implies that every single element in both sequences must be visited at least once.

While there are many options for path W for any two sequences, path W that minimizes the some of element wise distances along the path is the one to be used as a measure for similarity between two sequences. [Keogh 2002] This path gives the best alignment between two sequences, and its total sum over all the local costs contained in W gives the distance along that optimal path W. A local cost measure is defined as the distance measure used to compare two elements from two subsequences. A local cost measure can be symmetric, such as the square of the distance. Symmetric measure means that the same value is obtained regardless of the order of the two elements. As long as the local cost metric used is symmetric, DTW is also a symmetric measure between two subsequences.

Such path W can be found through calculating the total cost for every possible path through the M-b-M matrix subject to our constraints, and picking the one with the smallest cost. However, such an algorithm would require an exponentially large amount of time to compute and is thus unfeasible for large sequences.

An algorithm based on dynamic programming can be used to find the shortest path W in O(m*m) time. Cumulative distance(i,j) is found as the sum of the Euclidean distance(i,j) and the min euclidean distance of the adjacent neighbors (i+-1 and j+-1).

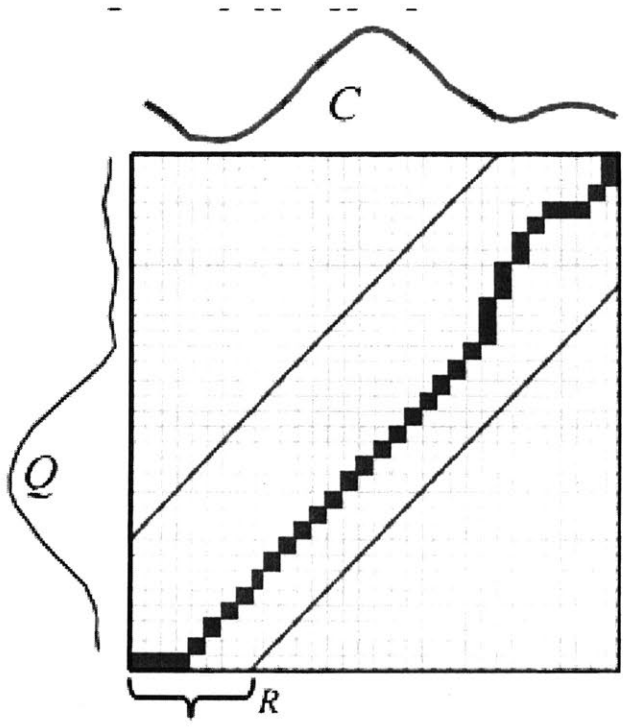

Figure 2-2: DTW constructs a matrix of element wise difference, with each cell being the difference between an element in Q with an element in C. Determination of optimal time warping for best alignment then becomes a search of the shortest path through the matrix, subject to the three constraints. [Ratanamahatana 2012]

[Oates 1999]

Rakhamon suggests it is the best one based on their review.[Rakhamon 2013]

2.4.2 DBA Averaging of a a Group of Subsequences

For k-Means clustering, an average of all subsequences within a cluster must be reliably computed. However, since we wanted to create cluster based on shape that is robust to phase shifts, a simple element-wise average would not work. Since we use DTW as the measure of distance between subsequences, our averaging method should arrive at a cluster average according to DTW distance.

Such method exists and is known as DTW barycenter averaging (DBA).[Petitjean 2011] DBA works by starting with an initial subsequence. DBA then calculates the cumulative DTW distance from the original subsequence to all subsequences to be averaged. DBA then aims to figure out the optimal warping of the allignment path between elements in the two subsequences to apply in order to minimize the cumulative DTW. Once such warping has been applied, the algorithm repeats itself and does so until converging on a final subsequence.

2.4.3 Improvements to Speed

Dynamic Programming gives us O(M*M) running time - this could be good enough for simply comparing similarity of two sequences. However, clustering algorithms require us to compute our distance measure many times over for the entire dataset.

Given the size of our dataset, DTW would be unfeasible based on the time it would take to run. Three main modifications exist to alleviate this problem: Sakoe-Chiba Band, Lower Bounding, and Early Abandoning.

Sakoe Chiba Band Constraint

Most common constraint used for DTW is the Sakoe-Chiba Band: warping path W must be contained within R cells of the diagonal. Effectively this constraint limits how far the algorithm is allowed to search to find the optimal alignment. By setting

the value of R to less than the length of a subsequence of m, we are no longer able to find the globally optimal path W if it lies beyond distance R of the diagonal. However, this significantly reduces the computation time from O(m*m) to O(m*w).

Lower Bound Keogh

DTW is expensive to compute for every pair of two sub-sequences in a dataset. Frequently, this distance is computed for clustering applications with the goal of identifying the closest matching cluster. It is necessary to compute DTW distance between z1 and z2 only in order to determine if the computed value is less than between z1 and any other subsequence(or cluster mean). Once it is determined that DTW between z1 and C1 is larger than DTW between z1 and some other previously computed distance, the actual value of DTW(Z1,c1) is not important. Thus it is common to use a lower-bound measure between z1 and c1 thatâĂŹs guaranteed to be smaller than DTW(z1,c1), if such measure is quicker to compute. Such measure is called LB-Keogh, and runs in O(m). [Keogh 2002] If LB(Keogh) distance between z1 and c1 is larger than the previous best candidate for closest match for z1, the computation of DTW(z1,C1) is unnecesary. If LB(Keogh) is smaller than the previous best DTW, only then the DTW(z1,c1) must be computed to determine if c1 is a better match for z1. This allows the algorithm to eliminate computation of DTW for most of the pairs.

Early abandoning

When LB-Keogh distance(z1,c1) is below the previous smallest DTW, we can use another step to potentially rule out c1 as a match for z1: lower bounding. While DTW is being computed, if the sum of the distances already exceeds the best-so-far match for z1-c2, then the computation can be abandoned.

While DTW between two sequences still remains expensive to compute, by using LB-Keogh and early abandoning we can avoid computing the actual DTW(z1,c2) for most of the cases. In fact, the algorithm ends up ruling out potential best matches using just LB-Keogh and Early Abandoning 99.9 of the time, and computing DTW

only about 0.01 of the time. [Rakthanmanon 2012]

2.4.4 Regularization techniques

Z-Normalization

In order to achieve meaningful clustering results on STS, the subsequences must be z-normalized. [Keogh and Kasetty 2003]

DTW is a measure of dissimilarity between Z1 and Z2 that depends on the magnitude of the values in both sub-sequences. Subsequences containing larger valued elements will naturally result in large value for DTW distance. Thus DTW is sensitive to the amplitude of subsequences in question. This leads to negative results by biasing the DTW towards subsequences consisting of smaller values.

This undesirable effect of amplitude should be removed via a normalization step. Z-normalization results in reducing the effect of amplitude on DTW computation. Z-normalized version of subsequence Z1 is obtained by reducing its each element by the mean of all elements in Z1 and then dividing the resulting values by the standard deviation of the set of the elements in Z1.

Tapering

In subsequence time series clustering, every example is a subsequence created by a sliding window. Sliding window moves along the time series and generates a subsequence out of the time series for every single possible starting point. Thus for any shape or event in a time series, it will be represented multiple times in the set of all subsequences - the same event will be in the middle of a certain subsequence, at the end of another, and at the start of yet another one. To function properly, clustering must be able to identify common reoccurring shapes or patterns within a time series. The subsequence examples should represent mostly and be associated with events that occur towards their middle, instead of at the edges. To accomplish this, tapering window was introduced.

Chapter 3

Clustering Methods

Since seismic waveform signals typically contain large amounts of data, calculating DTW and DBA averages, and clustering was a slow process. In order to get quick results, we used a synthetic dataset made up of the Ricker (Mexican Hat) signal and noise, shown in Figure 3-1

We also created and recorded several seismic events in a lab environment at low frequencies, shown in Figure 3-2. This signal does not contain any high frequencies, and thus we were able to sample it at a low rate without the risk of losing important features - this allowed us to run experiments quickly enough.

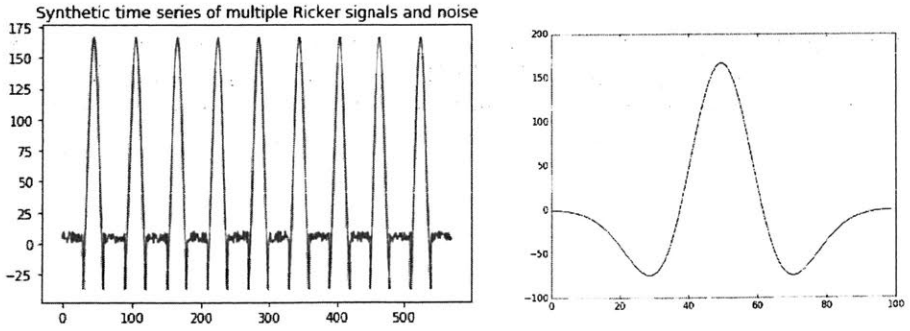

Figure 3-1: Synthetic dataset of Ricker signals and random noise. Figure below is an example of a view of a window centered on exactly one Ricker

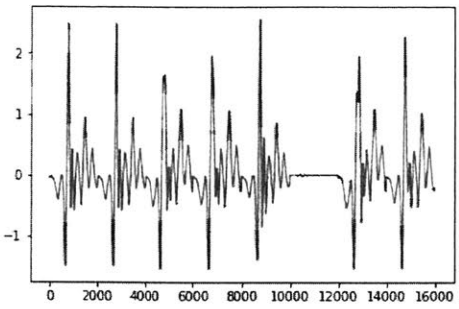

Figure 3-2: Seismic waveform recorded in a lab experiment environemnt

3.1 DTW as similarity measure between seismic trace subsequences

3.1.1 Different parameterizations of DTW

Sampling rate

The first parameter we set in discovering seismic waveform anomalies is the sampling rate. The original dataset has a sampling rate of 100 Hz - which means our complete seismic waveform is a vector, in which 200 elements represent a one second chunk. Our dataset is a month-long waveform, which translated to 2.6M seconds, or a 560M element vector.

We then generated 10 second-long subsequences from the waveform; each subsequence is thus a 1000 element vector. The subsequences were extracted using a sliding window approach, with a skip of 10 - first sub-sequence is the time series elements 1 through 1000, the second is elements 11 through 1011, and the final subsequence is 259999990th through 260000000th elements. In total, we had 26M subsequences to cluster.

DTW under Sakoe-Chiba Band constraint operates in O(M*R) time, meaning that it takes 1000*300 = 300000 operations to measure the distance between two subsequences, with a Sakoe-Chiba Band of 300. In on iteration of the k-Means clustering algorithm, distance is measured k times for each sub-sequence, since we measure every subsequence and cluster centroid permutation. With k=50 clusters and 260M

sub-sequences, one iteration requires 13B measurements of distances. However, since lower bounding and early abandoning are both utilized, true DTW distance is calculated for only about 0.01% of the cases. Lower bound Keogh runs in O(M) and thus takes 1000 operations to capture the measure between two sub-sequences.

Sakoe Chiba Band constraint

The Sakoe-Chiba Band constraint is the biggest speed-up to the calculation of DTW distance we have found. By limiting the optimal alignment search radius to 80% of the sub-sequence length (we earlier mentioned this as the search radius constraint, R), Sakoe-Chiba Band constraint is able to reduce one DTW measurement between two sub-sequences threefold. Sakoe-Chiba Band constraint has a trade-off: it is possible that the absolute optimal alignment between two given sub-sequences requires warping the time axis to over our 80% length ranges. In such case, result obtained through the use of the constraint does not represent the shortest path and is thus not a true measure of distance between the subseuqences.

However, our sub-sequences are generated through sliding windows, and thus, if such optimal alignment exists but is beyond a large range, then it is likely that the pattern found in one sub-sequence exists at the edges of the second sub-sequence. That pattern should then perhaps not be considered a prominent part of the second sub-section, and thus the result given by DTW Sakoe-Chiba was deemed to be a valid measure. DTW measured for the Ricker example over a full phase shift is illustrated in Figure 3-3

Z-normalizing

Subsequences do not preserve the overall amplitudes of the signal - thus some information is lost, and we are no longer able to find anomalies based on pure amplitude magnitude. However, each subsequence is z-Normalized relatively to itself alone, and thus shape is still preserved completely. We were interested in finding anomalies based on shape, and thus deemed z-Normalization to be an appropriate procedure to incorporate into our routine.

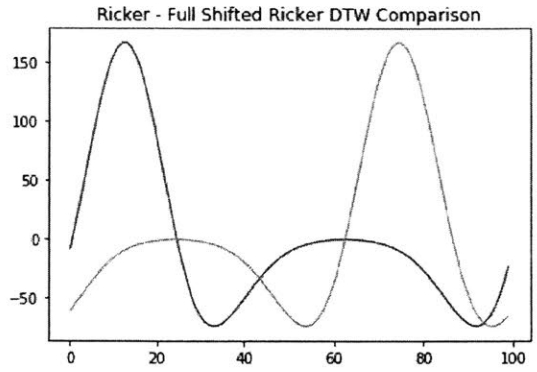

Figure 3-3: DTW distance measured between a Ricker sequence and a phase shifted Ricker sequence, DTW = 2629.7

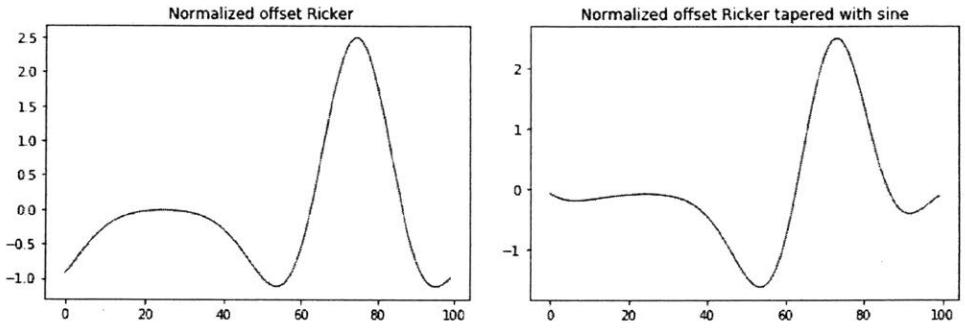

Figure 3-4: Comparison of a Ricker sequence with a Ricker sequence tapered with the sinusoidal signal

Tapering

A tapering window of a half period sine was applied to each subsequence prior to z-Normalization. Figure 3-4 shows that tapering causes the subsequence to be of smaller amplitudes near the edges.

Barrycenter averaging

DBA avergaing was chosen as the method for calculating a cluster representative. Cluster representative calculation method should accurately summarize the most predominant shape of all members of the cluster. Barrycenter averaging performs well in identifying a subsequence to act as a representative of all of the subsequences in this group, as shown in Figure 3-5

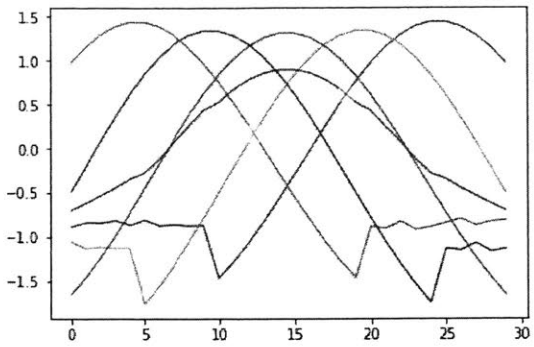

Figure 3-5: DTW barrycenter average centroid (brown) for a series of phase-shifted Ricker signals (purple, red, green, yellow, blue)

However, barrycenter is not robust to cases where the common shapes are offset significatnly. When a group of sub-sequences shares a similar shape, such shape should be present closer to the middle for each sub-sequences - for other cases barrycenter gives bad results.

One method to mitigate this problem is the use of sine-waves as tapering for sub-sequences - this tapering weighs the middle-of-the-sub-sequence pattern heavier. Due to the rolling window approach, inevitably âĂIJintermediateâĂİ sub-sections are generated - those that mostly contain the ending of an important pattern that came right before in its first half and the starting of a next important pattern coming after in the second half.

Chapter 4

Clustering Results

Here we present results achieved in the seismic anomaly detection problem using k-Means clustering and DTW as distance measure. We assessed the effectivenesses of DTW as a similarity measure between seismic subsequences under various parametrization and DBA as a method for obtaining representatives from cluster of seismic subsequences.

The final result of k means clustering is: 1. arrangement of sub-sequences into k clusters, and 2. centroids for the clusters The final arrangement of sub-sequences is the one that minimizes the distance for each sub-sequence to the centroid of the cluster the sub-sequence belongs to.

The centroid of the cluster acts as the representative for that cluster - this centroid represents the common patterns shared by all of cluster members. The centroids of clusters of most anomalous parts of the seismic waveform will be used to describe the discovered anomalous patterns.

4.1 1 Nearest Neighbor Classification using DTW

1-NN DTW classification was performed. The sub-sequences were all classified based on their closest neighbor out of our library of four types. Nearest neighbor classification using DTW is essentially template matching. True class or label of a subsequence is determined by looking at the class or label of a template subsequence. This can

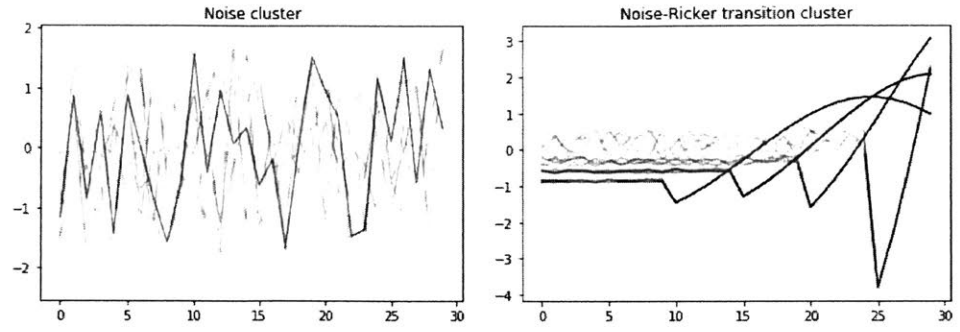

Figure 4-1: Subsequence classification result using nearest neighbor approach with DTW as distance measure. Left: all subsequences in the noise cluster (grey), and the centroid (blue). Right: all subsequences in the Ricker cluster (grey), and the centroid (blue).

also be thought of doing a single iteration of the K-Means algorithm, but with cluster centroids predetermined ahead of time, and kept unchanged always. Results are shown in Figure 4-1

This approach is not useful for anomaly detection - detecting an anomaly through this method would require the use of predetermined anomaly templates, which contradicts the nature of it being an anomaly. However, this method is useful as an intermediate check point on the data - it allows us quickly to assess the feasibility of using DTW and DBA for clustering subsequences within our specific application.

We provided the algorithm with four templates - Ricker, Noise, Ricker-to-Noise, Noise-to-Ricker. The first template is defined as the Ricker signal created through the scipy library in python. The second template is simply an array where each element is a uniform random variable within a range we pick ahead of time. Ricker-to-Noise and Noise-to-Ricker templates are defined as the subsequencs that capture the transition between the Ricker and Noise signals. If the nearest neighbor approach can correctly classify all of the present subsequences in the entire time series to their category, then DTW and DBA should be valid methods for us to use to capture shape similarity and identify prominent reoccurring shapes. This intermediate result confirms that DTW is a robust metric for comparing sub-sequences. Despite being out of phase most of the time, each subsequence was correctly measured to be closer in DTW space to its archetype.

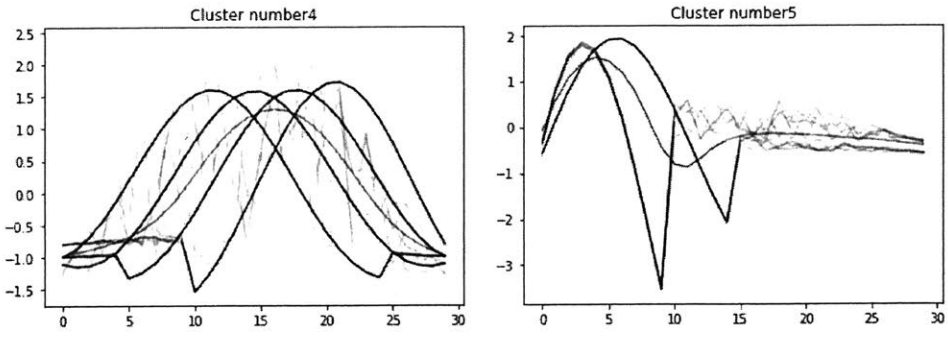

Figure 4-2: K-Means clustering result with clusters seeded by a user with DTW as distance measure. Figure left is a cluster of all subsequence determined to be Ricker. Figure right is a cluster of all subsequences determined to be the transition point between noise and Ricker

4.2 K-Means Clustering with Intelligent Seeding of Cluster Centroids

Next step towards a fully unsupervised k-Means was clustering, starting with a seed for each cluster centroid made up of one of the four archetype subsequences extracted ahead of time.

Unsupervised k-Means clustering with smart initial guesses for cluster centroids yielded reasonable clusters. It made a mistake in assigning a few noise subsequences to the Ricker cluster, but otherwise, most of the subsequences were correctly clustered together into the four main clusters. Each cluster centroid was found to be representative of the elements of the cluster, and the four obtained centroids correctly represented four main types of events in the dataset. Results are shown in Figure 4-2 and Figure 4-3.

Our dataset is a combination of Ricker and Noise signals, subdivided into subsequences starting at every 5th element. Thus the two dominant shapes in the data should be the Ricker and noise, as it is in 4-2.

K-Means was able to find a cluster of the Ricker and all of its offset variations, as shown in 4-3. It was able to correctly categorize every single Ricker variant into that second cluster. Note that only four clusters of subsequences with their centroids were plotted - this is because upon convergence, k-Means had arrived at six empty

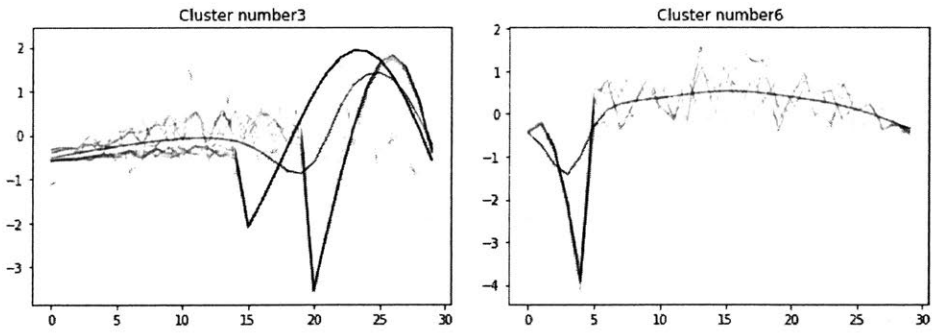

Figure 4-3: K-Means clustering result using nearest neighbor approach with DTW as distance measure. Left figure is a cluster of all subsequence determined to be a variation of Ricker signal and their centroid(blue). Right figure is a cluster of all subsequences determined to be the transition point between Ricker and noise and their centroid(blue).

clusters. Three clusters contain mostly either noise, or segments that are mostly noise and just partially Ricker. One cluster centroid is Ricker.

Surprisingly, clustering quality did not decrease drastically. A centered Ricker is a centroid for one of the clusters, and all of that cluster members are offset Ricker subsequences. Given that cluster centroids were initialized completely at random, this means that k-Means was able to extract the Ricker, Ricker-Noise, Noise-Ricker transitions as important patterns present in the time series. This means that we are able to extract important patterns out of a time series in a completely unsupervised fashion, and that these cluster centroids are meaningful summarizations of important shapes present in the data.

4.3 Completeley Unsupervised K-Means Clustering

While improving clustering results drastically, seeding presents a problem for detecting anomalies. Seeding requires an extraction of predetermined arc-type subsequences to serve as the initial guesses for the cluster centroids. However, with anomaly detection we have no real predetermined anomaly archetypes to extract. This is a problem because k-Means is a greedy algorithm, and can frequently become stuck on local optima for clusters.

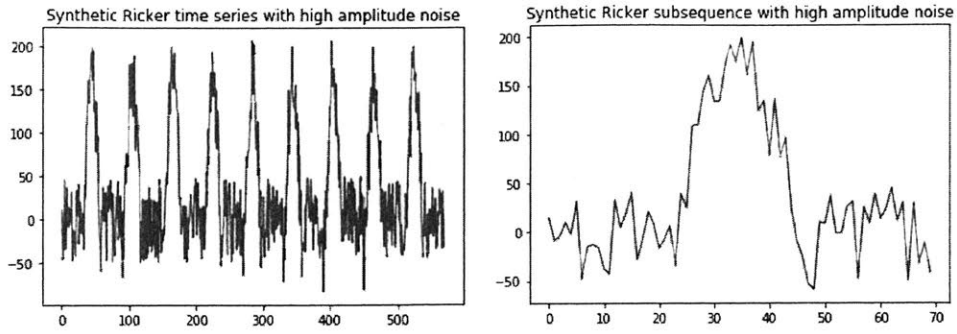

Figure 4-4: Full synthetic time series after noise has been added to the Ricker itself and noise amplitude increased.

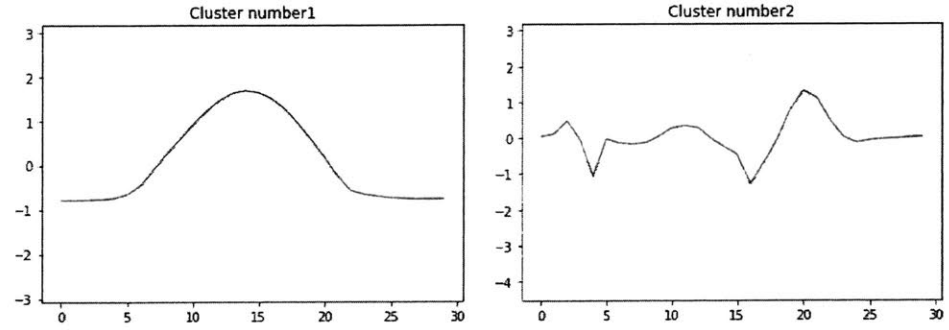

Figure 4-5: K-Means clustering results on synthetic ricker-noise dataset with absolutely no initialization done by the user. Figure to the right is the centroid (blue) and all subsequences (grey) within the noise cluster. Figure to the left is the centroid and all subsequences within the Ricker cluster.

We also added noise to the synthetic Ricker and noise time series to make the challenge more realsitic to real world applications. Full time series after adding the noise to the Ricker signal itself is shownn in Figure 4-4.

Thus we ran k-Means clustering algorithm with completely randomized initial cluster centroids. Cluster number was also set to 10 in order to leave more room for rare and anomalous subsequences to create a cluster of their own. K-Means converged after about 40 iterations. Results are shown in Figure 4-5.

Cluster centroids are initialized completely at random, so the method is completely unsupervised and does not require a predetermined understanding of what the common shapes of subsequences should look like. Cluster number two discovered was the Ricker itself - so the algorithm is able to extract all subsequences that are

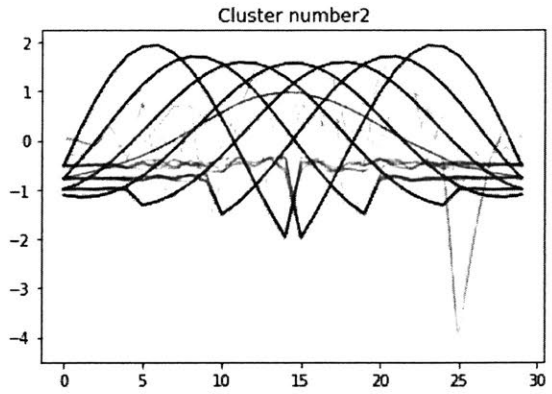

Figure 4-6: K-Means clustering results on synthetic ricker-noise dataset with absolutely no initialization done by the user: all subsequences determined to be different shifted versions of the Ricker (black), their centroid (blue), and misclassified noise subsequence (grey).

parts of the Ricker and determine that their common shape is the Ricker, without needing the Ricker needing to be a common shape or theme, as shown in Figure 4-5.

The unsupervised algorithm is thus able to extract Ricker as a meaningful pattern, and random noise at the edge of the Ricker as another meaningful pattern. On our synthesized seismic dataset, unsupervised K-Means clustering with randomized seeding is able to group the subsequences into the two clusters in Figure 4-7.

The centroid of cluster 1 appears to be anomalous relatively to the cluster 2 centroid. Subsequences contained within cluster 2 are indeed generated using a pulse, while cluster 1 subsequences were generated using a sinusoidal drive. Thus K-Means clustering algorithm is able to correctly assigns the two types of signals present to their respective clusters

Next, we attempted to run k-Means clustering using the Groeningen dataset. Groeningen dataset is information heavy and contains important features at high frequencies. Thus we first manually isolated six subsequences out of the data: three representing events, and three representing noise. We then measured pairwise DTW distances for all pairs within the six subsequences. Results are presented in Figure 4-8.

DTW measured between an event and an event is usually around 9-10; DTW

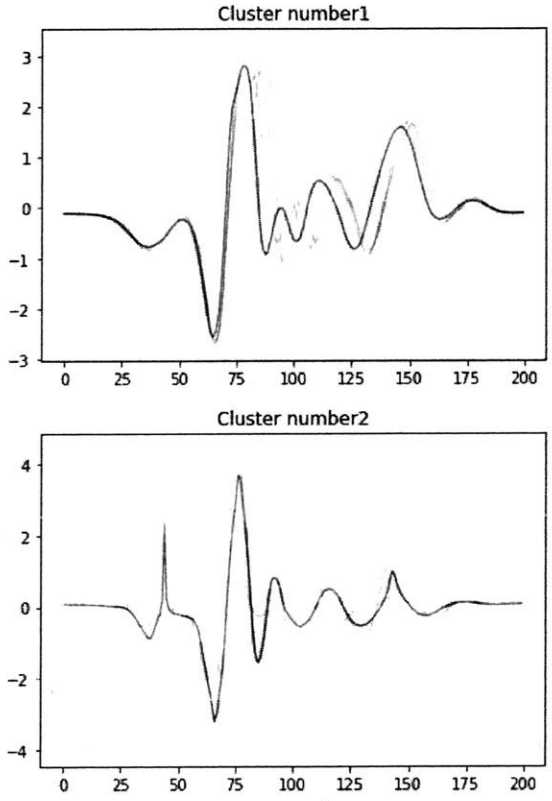

Figure 4-7: K-Means clustering results on seismic waveforms, with K=2 clusters, for an artifically created seismic waveform dataset. Left figure is cluster 1: all events of this type (grey) and their centroid (blue). Right figure is all events in this type (grey) and their centroid (blue).

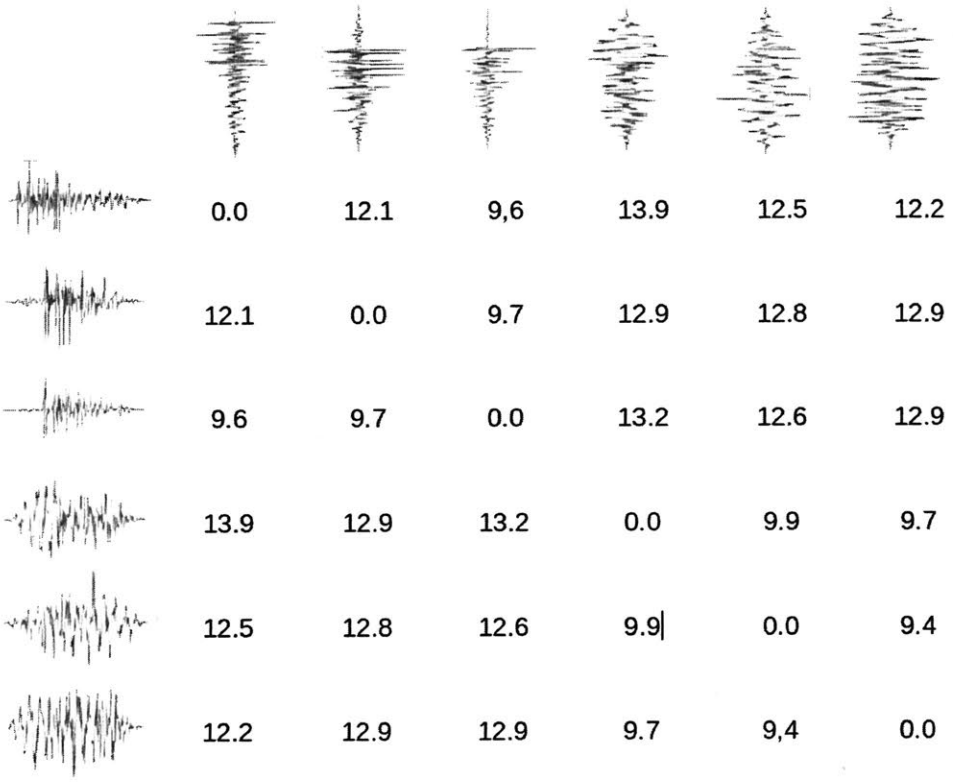

Figure 4-8: Comparison of pairwise DTW distances for selected subsequences of noises and events extracted from the Groeningen dataset

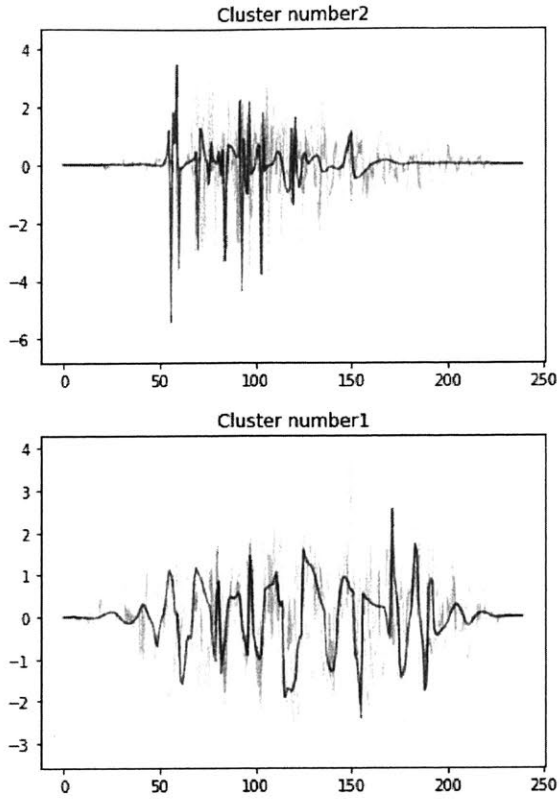

Figure 4-9: Result of K-Means clustering with fully random initialization after 20 iterations. Above are the centroid (blue) of and windows within the noise cluster. Below are the centroid of (blue) and windows within the events cluster.

measured between event-nosie is around 11-12, and DTW measured between two different noise windows was around 9.5. Thus DTW is able to sucesfuly see the difference between an event and noise. This gives us some hope for the ability of DTW to differentiate anomalous events.

Next, we ran a k-Means clustering algorithm on the six pre-extracted events and got results presented in Figure 4-9.

Here, K-Means was able to successfully classify each event and each noise as its correct category. This is an interesting result: the task of identifying events from noise is an active area of research. Typically, in machine learning this task is solved by using supervised learning, with the help of a pre-labeled dataset of noise and events. In this case, K-Means is able to correctly identify events from noise on a fully unsupervised basis.

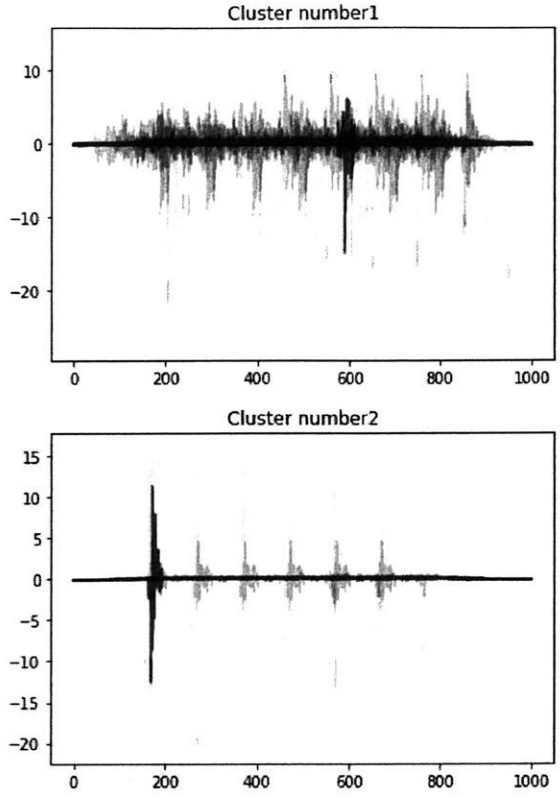

Figure 4-10: Result of K-Means clustering with full dataset with high resolution sampling. To the left is the cluster and cluster centroid (blue) for the first cluster. To the right is the cluster and cluster centroid (blue) for the second cluster.Above are the centroid (blue) of and windows within the noise cluster. Below are the centroid of (blue) and windows within the events cluster.

Finally, we ran unsupervised K-Means clustering on the full one month long Groeningen dataset (single component). Ten second long subsequences were extracted from the full one month time series. Our sampling rate was 100Hz and each susequence was thus 1000 points long. This would normally take an intractable amount of time, so we chose to censor: standard deviation was measured for each subsequence, and a threshold of about 500 was determined as a good separation for noise. Few events were present with standard deviation under that value, thus we deemed it a reasonable censor. We were left with 191 subsequences of length 1000. Each iteration took about 300 seconds, and we ran it for 10 iterations. Two final clusters are plotted in Figure 4-10.

Chapter 5

Conclusion

The final result of k means clustering is: 1. arrangement of sub-sequences into k clusters, and 2. centroids for the clusters The final arrangement of sub-sequences is the one that minimizes the distance for each sub-sequence to the centroid of the cluster the sub-sequence belongs to.

The centroid of the cluster acts as the representative for that cluster - this centroid represents the common patterns shared by all of cluster members. The centroids of clusters of most anomalous parts of the seismic waveform will be used to describe the discovered anomalous patterns.

1. Clustering is able to find anomaly in synthetic dataset; 2. DTW on real data shows DTW is able to distinguish anomalies form events and thus clustering should theoretically yield meaning ful results

The biggest drawback of the K-Means clustering and clustering algorithms in general is computational time. Seismic records are typically months long datasets, with events occuring at higher frequencies and over 1-5 second scales. Datasets are thus large. We have addressed all computational speedups possible to the process known so far.

Our first suggestion for future speed ups is to use parallel computing.

Ideally we need to come up with a dataset of various anomalies to judge performance, which allows us to objectively score different configurations and find the best set of parameters by maximizing accuracy. But for that we need a labeled dataset to

act as a quality check.

Next addition to the algorithm should be increasing it to 3D and multiple stations. 3D component data can be easily implemented - basically each subsequence would be represented as 3 vectors, corresponding to the three components. DTW would be calculated in the same way as before, except for that locally a simple element wise difference measure should be replaced by a euclidean difference.

In order to introduce multiple stations, however, we must change the DTW algorithm itself - namely we must allow for time axis warping between different station recordings such that the algorithm finds the optimal alignment itself. Lag and amplitude difference between the same event recorded in two different stations contains with it the information of the origin of th event. We suspect that the DTW path alignment can be used to extract this information. In other words, clustering multiple component multiple-station signals using DTW, algorithms should be able to cluster not just based on the nature of the event but also the origins of the event. So hypothetically, if we had two regular and two anomalous events, originating close to and far away from the stations, then it would be interesting to see if DTW based K-Means with K=4 is able to extract four clusters: normal event nearby, normal event distant, anomalous event nearby, and anomalous event distant.

References

Petitjean, Francois, et al. "A Global Averaging Method for Dynamic Time Warping, with Applications to Clustering." Pattern Recognition, vol. 44, no. 3, 2011, pp. 678-693., doi:10.1016/j.patcog.2010.09.013.

Zolhavarieh, Seyedjamal, et al. "A Review of Subsequence Time Series Clustering." The Scientific World Journal, vol. 2014, 21 July 2014, dx.doi.org/10.1155/2014/312521.

Keogh, E., et al. "Clustering of Time Series Subsequences Is Meaningless: Implications for Previous and Future Research." Third IEEE International Conference on Data Mining, doi:10.1109/icdm.2003.1250910.

Rakthanmanon, Thanawin, et al. "Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping." Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '12, 2012, doi:10.1145/2339530.2339576.

Dau, Hoang Anh, et al. "Semi-Supervision Dramatically Improves Time Series Clustering under Dynamic Time Warping." Proceedings of the 25th ACM International on Conference on Information and Knowledge Management - CIKM '16, 2016, doi:10.1145/2983323.2983855.

Zhao, Jiaping, and Laurent Itti. "ShapeDTW: Shape Dynamic Time Warping." Pattern Recognition, vol. 74, 2018, pp. 171-184., doi:10.1016/j.patcog.2017.09.020.

Ratanamahatana, Chotirat Ann, and Eamonn Keogh. "Three Myths about Dynamic Time Warping Data Mining." Proceedings of the 2005 SIAM International Conference on Data Mining, 2005, pp. 506-510., doi:10.1137/1.9781611972757.50.

Seto, Skyler, et al. "Multivariate Time Series Classification Using Dynamic Time Warping Template Selection for Human Activity Recognition." 2015 IEEE Sympo-

sium Series on Computational Intelligence, 2015, doi:10.1109/ssci.2015.199.

Manoharan, J James, and S Hari Ganesh. "A Framework For Enhancing The Efficiency Of K-Means Clustering Algorithm To Avoid Formation Of Empty Clusters." International Journal on Information Sciences and Computing, vol. 10, no. 2, 2016, pp. 22-31., doi:10.18000/ijisac.50163.

S. Rodpongpun, V. Niennattrakul, and C. A. Ratanamahatana, "Selective subsequence time series clustering," Knowledge-Based Systems, vol. 35, pp. 361-368, 2012.

A. Gorbenko and V. Popov, "On the longest common subsequence problem," Applied Mathematical Sciences, vol. 6, no. 113-116, pp. 5781-5787, 2012.

Keogh, Eamonn. "Exact Indexing of Dynamic Time Warping." VLDB '02: Proceedings of the 28th International Conference on Very Large Databases, 2002, pp. 406-417., doi:10.1016/b978-155860869-6/50043-3.

T. Oates, "Identifying distinctive subsequences in multivariate time series by clustering," in Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '99), pp. 322-326, San Diego, Calif, USA, August 1999

E. Keogh and S. Kasetty. 2003. "On the need for time series data mining benchmarks: a survey and empirical demonstration." Data Mining and Knowledge. Discovery 7, 4, 349-371.