Multi-Scale Time Activity Data Exploration via Temporal Clustering Visualization Spreadsheet

Jonathan Woodring and Han-Wei Shen

***Abstract*— Time-varying data is usually explored by animation or arrays of static images. Neither is particularly effective for classifying data by different temporal activities. Important temporal trends can be missed due to the lack of ability to find them with current visualization methods. In this paper, we propose a method to explore data at different temporal resolutions to discover and highlight data based upon time-varying trends. Using the wavelet transform along the time axis, we transform data points into multi-scale time series curve sets. The time curves are clustered so that data of similar activity are grouped together, at different temporal resolutions. The data are displayed to the user in a global time view spreadsheet where she is able to select temporal clusters of data points, and filter and brush data across temporal scales. With our method, a user can interact with data based on time activities and create expressive visualizations.**

***Index Terms*— Time-varying, time histogram, filter banks, wavelet, animation, transfer function, clustering, k-means, visualization spreadsheet.**

I. Introduction

DATA points in a time-varying data set exhibit different activities over time, which characterize the temporal trends of underlying features. We define a temporal trend as the characterization of the change in value of a data point over time in a series. This can be classified in a variety of ways, such as when a trend begins, when it ends, the rate of change, and the value over time. This is true for all time-varying data sets, and it seems natural to classify data points based upon their activity to find temporal hot spots.

Traditional data exploration deals with classification based upon temporally static value quantities. When time is not considered, patterns and correlations based on temporal activity can be missed. For example in animation of volumes, if the transfer function doesn't map a change over time to a visible range or the mapping does not highlight the dynamic activity, the user will miss the change or it will go ignored. It has been noted that change blindness can occur when visually perceptual phenomena change too slowly to be detected [23]. For rendering a single time step, most methods do not have a concept of mapping based on time activities or inter-correlated time activities.

Only until recently has scientific visualization attempted to tackle classification based upon temporal activity. Fang et al. developed a system to classify and segment medical data based on a distance metric from the time activity curve vector [5]. Akiba and Ma [1], [2] use the time histogram to create time-varying transfer functions based upon the time profile of a data set. Our method builds upon these concepts to facilitate time-varying data exploration, such that the data are explored and classified based upon multi-scale temporal activities.

Jonathan Woodring and Han-Wei Shen are with the Computer Science and Engineering Department at The Ohio State University, E-mail: woodring@cse.ohio-state.edu, hwshen@cse.ohio-state.edu.

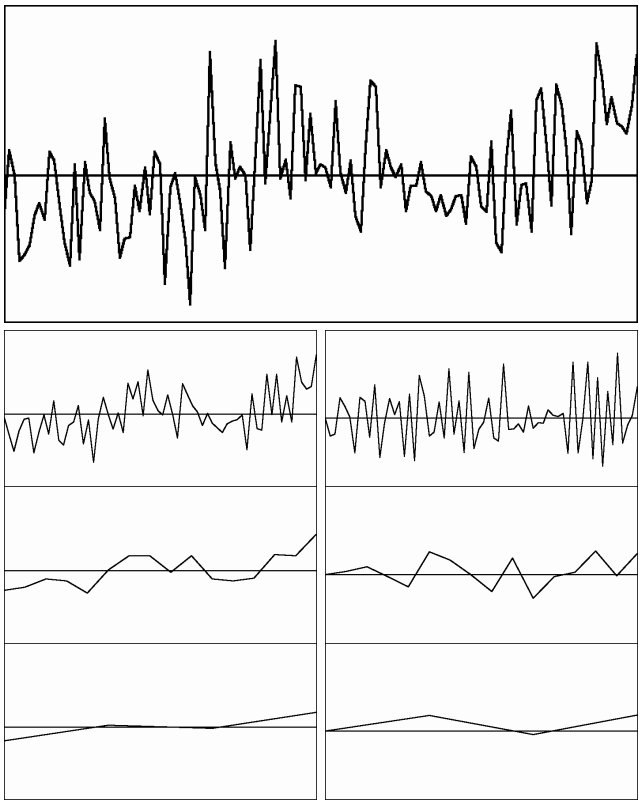

Fig. 1. The corrected annual US temperature data from 1880 to 2006. The top image is the original data. The second row through fourth rows are selected wavelets from applying the Haar wavelet transform to the original data, showing 2 year, 7.875 year, and 31.5 year trends. The left column are low-frequency wavelet coefficients and the right column are high-frequency wavelet coefficients.

Our goal is to allow explorative capability to find trends of different temporal scales in data to create visualizations based on temporal activity. Similar to multi-scale spatial classification [20], temporal activity can occur at different temporal scales. To explore the data, we utilize the wavelet [19] to transform data points into a set of time series curves grouped into different wavelet or filter bank levels. By filtering the data into different frequency bands, a user is able to visualize her data at different scales of activity, and find data that share similar temporal trends at particular time scale.

The data can then be explored by the activity in different frequency bands and wavelet coefficients, as can be seen in Figure 1, by applying the Haar wavelet transform to the recently corrected US annual temperature data from 1880 to 2006 from NASA. The top image is the original data, while the bottom images are selected wavelet coefficients after Haar wavelet transform. Going

down the rows, larger (slower) temporal trends can be seen in the data. The lower-left image shows that there was a rise in temperature in the first 30 years and a similar rise in temperature in the last 30 years. The high-frequency coefficients in the same frequency band in the lower-right image show the same trend, minus the DC component, and it can be clearly seen that the rise in temperature in the first 30 years and last 30 years is roughly equal in magnitude.

For regular data with multiple data points, we group data points into clusters of similar temporal activity using a hybrid clustering through k-means and SOM. The user then has a baseline to start exploring their data based upon temporal activity. Using an intermediate visualization, the wavelet clusters are displayed to the user in a visualization spreadsheet, so she can explore the global temporal trends present in the data [3], [12], [13], [17], [30]. The user is able to select, inspect, and filter [4], [25] time series clusters through the use of multiresolution time histograms [6], [10], [16] displaying the contents of each cluster. Selected and filtered clusters are combined together into a visualization by using boolean operations, time series operators, and animation [29]–[31].

II. Related Work

We utilize the wavelet to decompose time series data into a series of wavelet or filter bank levels. Wavelets have been used previously as a compression and level-of-detail reduction [19] scheme for large data sets. Recently, Lum and Ma have used filter banks as a method to classify spatial data based on different frequency ranges [20]. In information visualization, the Fourier transform is used to decompose line graphs into different scales to do optimal banking to 45 at different temporal resolutions, where banking to 45 attempts to find an optimal aspect ratio for 2D graphs to maximize the angles between line segments to 45 degrees for perceptual visualization enhancement. [8]. We use the wavelet transform to extract multi-resolution temporal trends. There has been use of multiresolution histograms for image recognition and matching [6], [10]. Jain and Merchant apply the wavelet transform to the color histograms for images to build multiresolution histogram pyramids for image retrieval. Hadjidemetriou et al. apply spatial image filtering to acquire multiresolution histograms for image matching. Our use of multi-scale time histograms is most like the latter, we filter the data along the time axis and use clustering [7], [15] to match temporal trends.

Brushing, linking and multiple data views have been used in the visualization as a means to be able to draw conclusions between multi-variate data. Ggobi and the like are the latest incarnations of multi-variate data brushing and linking [4], [25]. Our use of brushing and linking is applied in the selection of data points across temporal scales. The use of spreadsheet formats are likewise used throughout visualization to simultaneously display multi-views of data [3], [12], [13], [17], [30]. We use visualization spreadsheets to present the data that are acquired through wavelet transformation and clustering. The use of graphical widgets and user interfaces for are prevalent in classification and transfer function design [14]. The combination of wavelet transformation, clustering, time histograms, and brushing and linking create a complete user interface to explore and classify data by temporal activity.

There have been various research efforts in visualizing time series data in the area of information visualization. For instance, van Wijk and van Selow combined cluster analysis and calendar based visualization to identify standard daily patterns throughout the year [27]. Weber et al. [28] proposed a method to visualize time-series data based on spirals. The spiral graphs are effective for detecting periodic patterns for large scale data sets. Hochheiser and Shneiderman [9] proposed a Timebox widget to specify query constrains on time series data. Lin et al. [18] introduced VisTree, a time series pattern discovery and visualization system to help analyze data from aerospace applications.

Creating transfer functions and visualizing time-varying volumetric data is one of recent interest [21]. There has also been other work in automatic and semi-automatic methods of transfer function creation for time varying data sets [11], [22], [26]. More recently, Akiba and Ma have used time histograms to be able to classify data based upon their time series profile [1], [2], [16]. Their method is better at feature tracking over space. Akiba and Ma's assumption is that data point populations will aggregate in value space and move together in value space. In this paper, we make the assumption that interesting data points have similar temporal trends. While our method is better at finding and isolating data points based on similar activity, we utilize their method for creating dynamic transfer functions for data points based on extracted temporal features. Our assumption is more closely linked to Fang et al. [5]. They use a method to segment medical data based upon calculating distance metrics from a time series profile curve [5]. We also extract data points based on time activity, but across several temporal scales with the ability to query and explore at different resolutions. This work of temporal activity selection provides the data input to volume combination methods over space and time to create final visualizations [29]–[31].

III. Methodology

In order to allow the user to find temporal trends in her data, we model each data point or position in time series data as a 1D time signal. A *time series curve*, *time activity curve* [5], or *time curve* is a vector representing the value over time at a particular data point. The time series vector for a data point is a vector of t elements ordered by time, where t is the number of time steps in the data set, and the value of each element is the value at a time step of that data point. Assuming the data value is scalar over time, if we were to plot the value over time in a 2D graph, the line curve would comprise its graphically representative *time curve*, like the top graph of Figure 1.

In a 2D time curve line graph, the user is more likely to be able to detect changes in value over time, or detect trends and anomalies present in the time signal, compared to animation or volume visualization. This is due to the global time awareness and value change that is displayed by 2D plotting. The entire time sequence is available for viewing at a glance. In volume rendering, comparatively, the user has one slice in time displayed at any given moment. Value change or trend detection depends heavily on the capability of the transfer function to display the trends, the speed that an animation is played at, and the memory of the user. Furthermore, trend detection in animation can be easily missed due to opacity, occlusion, visual memory, change blindness [23], or any number of other perceptual factors.

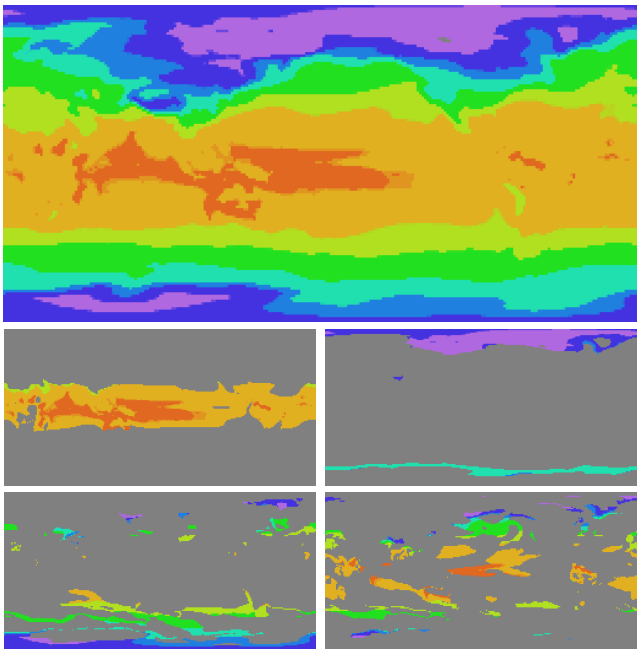

Fig. 2. 2 meter atmospheric temperature data from CCMS 3.0. The top image is the temperature for the entire world for the first time step. The second row shows the data points from two clusters from a low frequency band wavelet clustering. The left image shows a clustering of a high temperature, which is near the equator, and a low temperature clustering, which is near the North Pole and waters around Antarctica. The third row shows two clusters of high wavelet coefficients in the same frequency band. The cluster on the left has a decrease in temperature followed by an even greater decrease, and the second cluster has an initial rise in temperature followed by a dip.

To further aid the user, we decompose each time series curve into wavelet levels or a series of filter bank levels, as in the bottom graphs in Figure 1. By using the wavelet transform, we filter the time activity curve of each data point into several scales of temporal activity. This allows the user to explore the activity that takes place at different temporal scales. In the displayed wavelet coefficients for Figure 1, we show 2 year, 7.875 year, and 31.5 year trends that are present in the single data point. The user can then explore and classify the data based on various trends that occur at different scales in the data. As in Figure 1, the different 2 year, 7.875 year, and 31.5 year trends that are masked by high-frequency spikes in value are revealed through filtering, like multi-scale banking to 45 and multi-scale volume exploration [8], [20].

One data point is easy to plot, like the aggregate US annual temperature data, but when dealing with millions of data points, such as in time-varying volumetric data, it becomes intractable for a user to visualize the time curve for every data point. To expedite the user search process, we utilize clustering [7], [15] to form temporal summary data for each frequency band and wavelet coefficient type. Clustering is run on groups of wavelet vectors separated by frequency band and wavelet coefficient type to form cluster groups. This means each data point will be grouped with other data points that share similar temporal activity in a particular frequency band and wavelet category.

An example of the different clustering that can take place in data is shown in Figure 2, which from a 6000 time step series of 2 meter atmospheric temperature data of the world. The rendered image for the entire data is shown on top, colored by the values

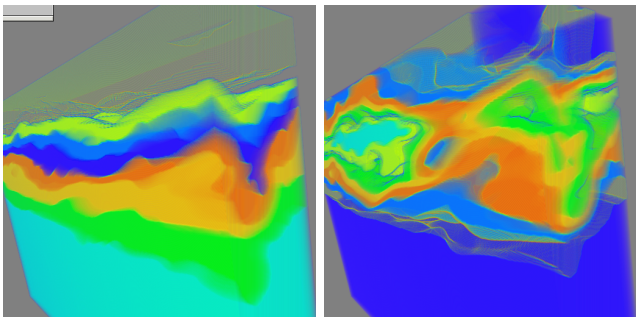

Fig. 3. Examples of temporal clusters from combustion data. The data points are colored by the temporal cluster that they belong to, so that all of the data points with the same color have similar temporal trends. The left image shows the temporal clusters in the mix fraction variable, and the right image shows the temporal clusters in the OH variable.

of the first time step out of the series. The data was wavelet transformed, and then clustering was performed separately in each frequency band and wavelet coefficient type. The clusters shown in the example are from a low-frequency band, showing 3000 time step trends, or showing the trends in the first and second half of the data. The second row shows two different clusters based from clustering the low wavelet coefficients, one cluster of a high temperature area and one cluster of a low temperature area. The third row shows two groups based on clustering the high wavelet coefficients. The left image shows data points of decreasing temperature, followed by an even greater decrease in temperature. The right image is a cluster of data points that has an increase in temperature, followed by an decrease in temperature. A second example of classification or separation of data points by temporal clustering can be seen in Figure 3, using two variables of a combustion data set. The image shows all of the temporal clusters found in the data, and the data is colored by the temporal cluster.

The data point clusters are shown to the user in a spreadsheet format [17], with the centroid curve and thumbnail volume rendering of the data in the cluster. This is the primary user interface which displays all of the time trends present in the data and allows the user to select the clusters she wishes to explore for visualization. Data cells are organized by frequency band and wavelet category, and sorted by derived properties such as cluster distance or population overlap on cell selection. Spreadsheet cells can then be filtered by centroid similarity or spatial overlap to reduce the data complexity and increase the summarization of the time series data. A small spreadsheet is shown in Figure 4. This spreadsheet shows all of the low wavelet coefficient clusters in the frequency band that represents 15.25 time step trends for a 122 time step data set.

Upon cell selection, the cluster is placed on a detail spreadsheet, displaying all of the wavelet coefficients at different time scale of the selected cluster, along with other selected clusters. The wavelet coefficients in a cluster are rendered in the form of the *time histogram* [16]. A time histogram is a compact representation of a series of histograms over time. A histogram for one time step is represented by a vertical strip that spans the value range of the data points, with each bucket rasterized as a rectangular area with the bucket count displayed with color or intensity. The horizontal axis represents time. Figure 5 shows an example of one time histogram cell from a series of time histograms for a cluster.

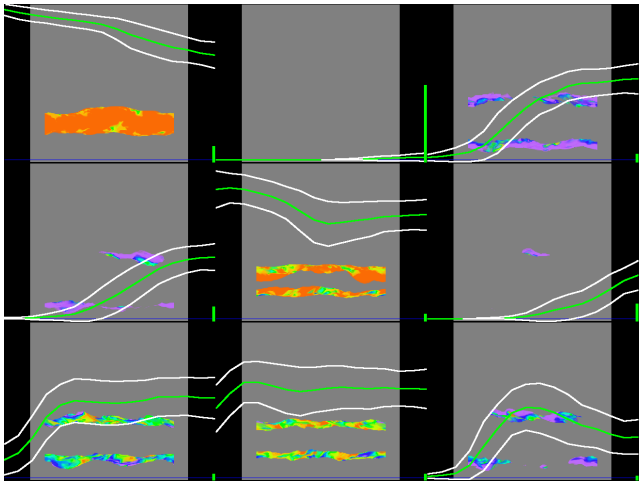

Fig. 4. A cluster spreadsheet featuring the mix fraction variable of a 122 time step combustion data set. The spreadsheet is only showing clusters for 15.25 time step trends for the low wavelet coefficients. The thumbnail of the data points for each cluster is rendered, and the centroid time curve of the data points is rendered on top in a green curve.

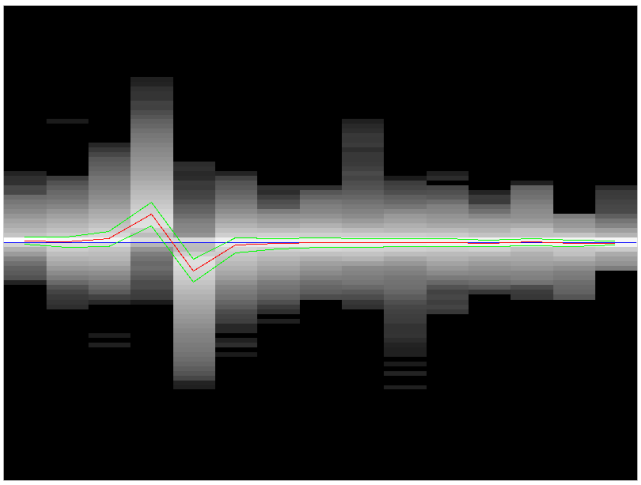

Fig. 5. A time histogram for a cluster from high wavelet coefficients of 15.25 time step trends in the χ variable of a 122 time step combustion data set. The high coefficient value over time for all data points in the cluster is shown by the time histogram. Count intensity is rendered in a log scale to emphasize outliers, since the centroid is shown. The red curve is the centroid of this cluster. Since this is a high wavelet coefficient curve, from the centroid, we can see there is a zero crossing, meaning there is a temporal peak in the data at that time period in that frequency band.

The time histogram is able to display the value distribution and variance of the different data points over time in a cluster, for each wavelet type and frequency band. This allows the user to have a better understanding of data contained within each cluster, and to compare the data with other cluster's time histograms.

The user can further refine the data point selection in a cluster to direct the exploration process. We allow the user to brush and link [4], [25] over the time histogram plots to make selections. She can filter data points by time curves and trends, as seen in Figure 6. Figure 6 shows several time histogram views of the same set of data, where the user has brushed a selection in one time histogram. She can see the results of her brushing in the other time histograms of the same set of data, but at different temporal scales of activity and wavelet coefficient types. Additionally, the

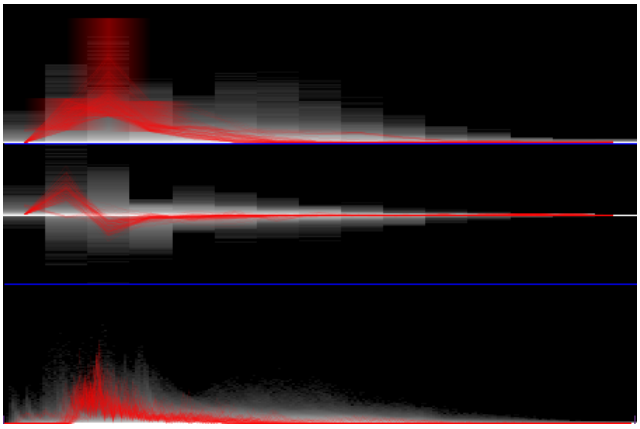

Fig. 6. Selected time histograms from a 227 time step earthquake data set, featuring brushing and linking. In the top time histogram, which is 15.13 time step low coefficient trends, the user has brushed a trend of interest, and the time curves that pass through the brushed area are highlighted. The same time curves or data points are linked in the other time histograms. The middle image is the same data at the same frequency band, but is showing high wavelet coefficients. The bottom image is the original data time histogram. The user is able to brush and alter the selection in all of the linked time histogram cells.

user can make selection through value ranges via dynamic transfer function painting [1], [2]. The spatial combination of selected data points is accomplished through boolean operations or operators over time [29]–[31]. To make trends more visible in animation, we allow the user to create visualizations that dynamically rescale the animation speed based upon the data trend.

IV. Multi-scale Temporal Extraction

In order to study the time-varying data at multiple temporal resolutions, we need to transform the data. Wavelets provide an elegant implementation for filter banks [24], which are the foundation for multi-resolution analysis. Wavelets have been used often in visualization for compression and multi-resolution rendering [19] in the past. Wavelets are defined by basis functions that filter a signal into two parts: low-frequency or approximation coefficients, and high-frequency or detail coefficients. The process can be repeated on the low-frequency coefficients to create a hierarchy of resolutions [24]. The hierarchy formed by repeated applications of the wavelet transform forms a filter bank, or a set of band limited signals based upon the original signal. Among the different types of discrete wavelet transforms, the Haar wavelet and the Daubechies wavelet are commonly used. The Haar wavelet is very fast to calculate and has a simple support, and from the user point of view, the calculation of the wavelet and its meaning is simple to understand. We have a series of increasingly low-pass signals based upon the original signal, representing the data at different time scales. Also, we have the derivatives of each of the low-pass signals, showing the rate of change of the data over different time scales.

A. Haar Wavelet

Assuming that the time sampling rate for the data is regular or can be converted to a uniform sampling rate, we use the Haar wavelet to transform the data into a multi-scale temporal format. The Haar wavelet has several different notational forms. The version that we use is the low-frequency coefficients being the

mean of two adjacent samples, and the high-frequency coefficients being the difference of two adjacent samples. The reason for this notation and is for the ease of understanding from the user point of view, rather than reconstruction support. In this work, the wavelet transform is used to represent the time series at different temporal scales for user exploration. By reasoning on the wavelet transform in this way, to the user, the low-frequency wavelet coefficients are the average or value signal, and the high-frequency wavelet coefficients are the rate-of-change or derivative signal. The Haar transform is given in Equations 1–2, where O is the original signal, L are the low-frequency coefficients, and H are the high-frequency coefficients.

$$L_i = \quad (O_{2i+1} + O_{2i})/2 \tag{1}$$

$$H_i = \quad O_{2i+1} - O_{2i} \tag{2}$$

The recursive process of applying the Haar transform on the L coefficients results in a cascading filter bank of $\lceil log_2(t) \rceil$ levels of low-frequency coefficients and high-frequency coefficients, where t is the number of samples. For reconstruction, strictly speaking, a level l filter bank has $\lceil t/2^l \rceil$ low-frequency coefficients and $\lceil t/2^l \rceil$ high-frequency coefficients.

Though in our visualization, we use $\lceil t/2^{l-1} - 1 \rceil$ high-frequency coefficients, to display the difference between every adjacent time step. Strictly speaking, if we were using the wavelet as a compression or backend level-of-detail method, the additional coefficients are extraneous for reconstruction. The additional coefficients are for visualization and user analysis to show the difference between every time point. For example, given 8 data points, 12233445, the strict Haar high frequency coefficients would be 1111. Instead of visualizing those coefficients, we increase the number and show the user 1010101, which is much more informative in terms of showing the rate of change of the signal. This is on the order of a $2(nlogn)$ transformation, because of the extra set of coefficients.

For our data transformation, for a data point x in a time varying data set, it has t samples over time, where t is the number of steps in the time series. The t samples of x form a time series vector v, where the elements of v are ordered by time. The Haar transform is applied to every time vector in the data set, such that for every v, $Haar(v)$ is a Haar wavelet hierarchy representing data point x at different frequency bands and wavelet coefficient types. Alternatively, it can be thought of as filtering a data point x across time to extract its characteristic signal across frequency bands and signal types. This wavelet transformation is used to display a data point's temporal trend at various temporal scales to the user. Assuming that the data is scalar, the multi-scale temporal trend of a data point can be displayed by drawing the 2D curve of each wavelet vector.

Figure 1 is an example of applying the Haar wavelet transform to one data point. The top image is the original temperature over time. The bottom images are the wavelet transformed data plots.

B. Temporal Activity Clustering

It is not possible to be able to draw every 2D wavelet graph for every data point in a data set of realistic size. There would be simply be too many line graphs to plot and explore. In order to reduce the data set size and to create summary information, we employ clustering on the wavelet data. Clustering is a method for grouping a set of high-dimensional vectors into semantic sets. In the most general terms, a similarity or distance metric is repeatedly applied to the input set of vectors to separate the data into semantic groups or clusters. Common methods for clustering include energy minimization solutions like k-means, and machine learning algorithms like the self-organizing map [7], [15].

Our use of clustering is employed on the time vectors that are generated through the wavelet transform to form sets of data points that exhibit similar time activities. For each data point x in a time series data set, it will have $2 * \lceil log_2(t) \rceil + 1$ vectors representing the original data, and the low-frequency and high-frequency wavelets from the Haar transform. For each of the vector types, we cluster the data points in the time-varying data set. Since we have $2 * \lceil log_2(t) \rceil + 1$ groups of frequency band and signal type, a data point x will belong to one cluster in each of the groups as a result of this clustering process.

By performing clustering on each signal type separately, a data point is grouped with other data points that have similar temporal activity in a particular frequency band and wavelet type. By clustering by low coefficients, temporal activity grouping is dominated by value over time. When grouping by high coefficients, data points are clustered by the rate-of-change over time. This allows the user to explore the range of possibilities of temporal trends over each frequency band and signal type. We assume that there will be population separation in the clustering, such that clusters across frequency bands and wavelet coefficient types will have different data point populations. This is where interesting temporal trends should appear in exploration.

Figure 2 displays several clusters in a low-frequency band. The second row shows data points which were grouped together because they had similar low wavelet coefficient activity over time in that frequency band. If we cluster by high wavelet coefficients, we obtain clusters, seen in the bottom row, that contain some of the same data points in the first two clusters. Therefore, the user can make a more refined selection by taking the intersection, through boolean operations, between two clusters. For example, if the user intersects the upper left cluster with the lower right cluster, she is able to visualize a region that has high temperature value, but also raised and then dipped in temperature.

The clustering method that we use is a hybrid SOM and k-means with kd-tree acceleration. While k-means and kd-tree are well known, the SOM, or self organizing map, is an AI learning network that projects high dimensional vectors to a lower dimensional space, while trying to preserve the topology of the higher dimension space. We use the SOM to quickly arrive at an initial, hopefully globally optimal, centroid set for clustering, and then use k-means to refine the set to a convergent answer. The drawback is that k-means and SOM require the number of clusters as input, therefore it may under- or over-cluster because the number of clusters picked may not be the number of natural clusters in the data. The initial use of SOM attempts to shake k-means into a global minimum, rather than local minimum. There may be utility in using an alternative clustering method for creating temporal summaries, compared to the clustering we have used.

One drawback or limitation that we have in our method is that we are able to classify data points into temporal hotspots, such as regions in space that share the same activity, but we are not able to do spatial tracking of values or temporal phase shift. For example, in in the case studies found later in the paper,

our method is able to classify the weather data into geographical regions by their temperature patterns over time. Though, for data such as the combustion data set, we classify volumetric regions that have spatially moving data values are moving through a region, rather than tracking the value over space. This is because a value moving through space appears as an impulse time curve as it passes through different regions of space.

V. User Interface

The goal of the user interface is to display the time activity of a data set and to allow the user to select data points based on the visualized temporal activity. A cluster spreadsheet is formed from data acquired through wavelet transform and clustering on the input. From the clusters, the user is able to explore and find data that have interesting temporal activity. To get further detail, the user can select the clusters of interest and place them on a time histogram spreadsheet. From the cluster selection and refinement, the visualization is formed, highlighting data of user selected temporal activity, which in turn can lead to further refinement and exploration.

A. Cluster Spreadsheet

The cluster spreadsheet is the primary interface that forms the temporal trend exploration. Each cell of the spreadsheet represents a cluster formed from the clustering of wavelet data, explained in the previous sections. Cells are sorted by spreadsheet columns, such that each column contains clusters of one wavelet level (frequency band), and sorted left to right by lowest detail to highest detail.

Low wavelet coefficients are on the left half of the spreadsheet, while high wavelet coefficients are on the right half of the spreadsheet, so that comparisons can be made between wavelets of the same type. Given there are $\lceil log_2(t) \rceil$ wavelet levels (frequency bands) from the Haar wavelet transformation, two wavelet coefficient types from low and high coefficients, and a fixed number of clusters k per wavelet category type, then there will be $2 * \lceil log_2(t) \rceil * k$ cells in the spreadsheet, where there are k rows and $2 * \lceil log_2(t) \rceil$ columns. An example full cluster spreadsheet can be seen in Figure 7.

To show the summary information, each cell graphs the centroid time curve of the cluster, the average variance of the cluster members from the centroid, and a thumbnail rendering of the data points in the cluster, giving a temporal and spatial summary of the data that is contained in each cluster, as seen in Figure 8. On the right edge of the cell is a bar indicating the ratio of population of data points in the cluster to the total data point population, so the user can see the size of the cluster. Next to that, there is an additional bar indicating either intersecting population count or centroid distance from a reference cluster that is selected by the user. The quantity that this bar shows is used in relevance reorganization, as is explained below.

One point of mention is that this spreadsheet interface can have a problem with data explosion or overwhelming the user with too much information. Since we extract several different time scales and multiple clusters of data, it can be a daunting task for the user to be able to search and explore the data. The clustering was a first pass at data reduction, and we have several different user interface controls sorting and simplify the amount of data shown to the user. Our user interface makes a good attempt for visualizing multi-scale temporal data, but there is room for future improvement, beyond what we have done here.

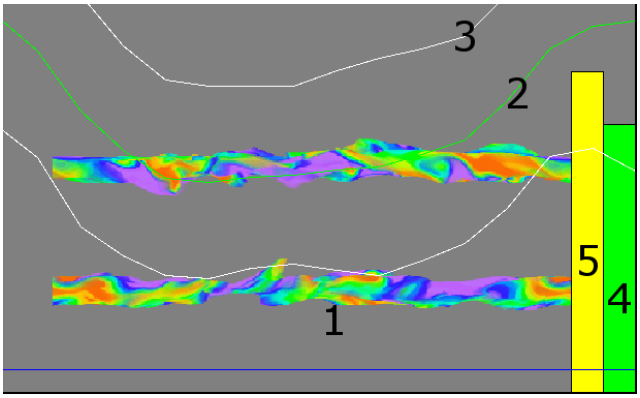

Fig. 8. A close up of a cluster cell from Figure 7. The background of a cell (1) contains a thumbnail rendering of the data points in the cluster. In the foreground, the green curve (2) represents the centroid temporal trend of the data contained in the cluster. The white curves (3) indicate the average temporal variance around the centroid time curve. The green bar in the lower right (4) indicates the population size of the cluster. The yellow bar in the lower right (5) is for showing data point overlap with another cluster, or the centroid distance from another cluster.

B. Relevance Reorganization

When the user selects a particular cluster cell, the entire spreadsheet is reorganized to display the relative relevance of other clusters to the selected cluster. The basic rules for reorganization is that a cell will stay in its own column, as column is an indication of frequency band and wavelet coefficient type. A cell can move up or down within its column. The goal of sorting in the column is to move cells vertically closer to the picked cell so that closer cells will have higher relevance. So, when the user is browsing the spreadsheet, after choosing a cell, she is presented with a spatial reorganization of the spreadsheet to display clusters with similar temporal or cluster population characteristics. We note that all the following reorganization methods can be inverted to reorganize the cluster spreadsheet to highlight dissimilar cells when necessary.

1) Same Column Reorganization: Within the column that a picked cell resides, other cells in that column are reorganized based on the cluster centroid distance from the picked cell. The effect is that when a user picks a cell, clusters in the same column are moved closer if they have a smaller distance, using the clustering distance metric, between their temporal centroids, and moved farther away if there is a larger distance. The first columns of the images in Figure 9 show an example of resorting cells in the column of the picked cell. Cells that are temporally similar have moved closer, in this case similar cold temperatures, to the picked cell (please see the figure caption for detail). By moving cells that have similar centroids closer to a target cluster, we emphasize the clusters that have the same temporal trend as the picked cluster. In addition to resorting the cells, we always show the relative normalized temporal centroid distance in a cell with a red vertical bar on the right of the cell.

2) Other Column Reorganization: The other columns, which are not the column of a picked cell, are sets of clusters in different frequency bands and wavelet coefficient types. In order to show their relevance to a picked cell, we can resort cells vertically

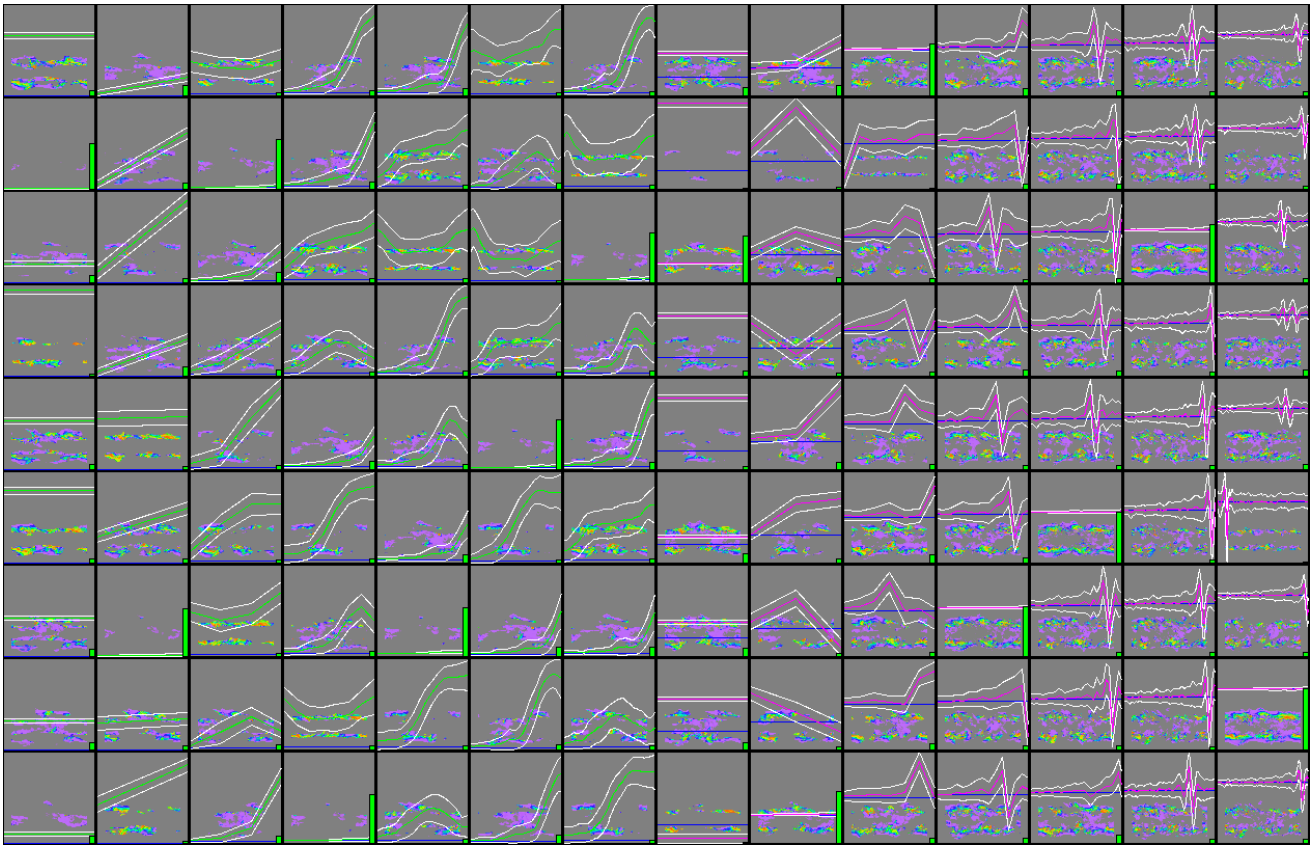

Fig. 7. A cluster spreadsheet of the OH variable of 122 time step combustion data set. After the data has been wavelet transformed and clustered, it is organized into a spreadsheet. The left half of the spreadsheet shows low wavelet coefficients, while the right half shows high wavelet coefficients. The columns are organized from lowest frequency band (long term trends) to highest frequency band (short term trends), reading left to right. Each cell is one cluster of a particular frequency band and wavelet coefficient type. The spreadsheet gives a global view of the different trends that are present in the time varying data.

such that their relative vertical distance from a picked cell is equal to the percentage of shared population from the picked cell. This is equal to $COC(A,B) = |A \bigcap B|/|A|$ where A is the picked cell and B is the cell to be sorted, which we call *cross-over-count*. The second and third columns of the top image in Figure 9 shows an example of how other column reorganization works. The user has picked cell (1) with cold temperatures, and cells with the highest overlapping data point population in other columns are moved vertically closer. Cluster cells with smaller cross-over count will be moved relatively farther away from the center row. By moving cells with the highest shared percentage population vertically closest to a picked cell, we emphasize how data point populations recluster across frequency bands and where temporal trends diverge across frequency bands. In this example, we can see that there is a split in the cluster population as we increase in detail of temporal scales. For this particular data set, we can reason that this is due to the northern and southern hemisphere monthly temperature cycle. In short term temporal trends, data points are temporally similar to regions in the same hemisphere. In a longer term trends, data points are similar to regions in the same latitude, ignoring monthly trends. We always show the cross-over count between a picked cell and every other cell by a yellow vertical bar on the right of a cell.

3) Row Sensitive Reorganization: To emphasize similarity across all rows, we can sort the cells in each column based on a greedy cross-over-count selection compared with the cells in the column that the picked cell resides. To do this, The picked cell first moves the cell with the highest cross-over-count in each column vertically into its row. Then, the cell with the smallest centroid distance to the picked cell in the same column does the same, i.e., moves the cell with the highest cross-over-count with itself in each column vertically into its row. This is repeated for the next smallest centroid distance cluster, until there are no more cells to reorganize. The bottom image in Figure 9 shows an example of using this reorganization method on the spreadsheet. Rows (A) and (B) are populationally similar to the clusters (2) and (3), respectively, in the column where the picked cell resides. This tries to ensure that there is relevance across columns as well, such that the user reads across rows, all of the clusters are from similar data point population, although this is not always guaranteed due to the greedy selection method we use. We can see a shift in the other cluster populations over temporal scales as well as the picked cell. The user can still see the cross-over-count with the picked cell by the yellow bar on each other cell. As an alternative for using cross-over-count as a reorganization metric, the centroid distance between clusters can be used as well in the previous two methods for sorting columns, to emphasize temporal trend similarity between clusters rather than population similarity.

C. Spreadsheet Simplification

The user interface visualization can be complex due to the fact that many cells are displayed at once, like in Figure 7. Further simplification can be done by only showing a few columns of

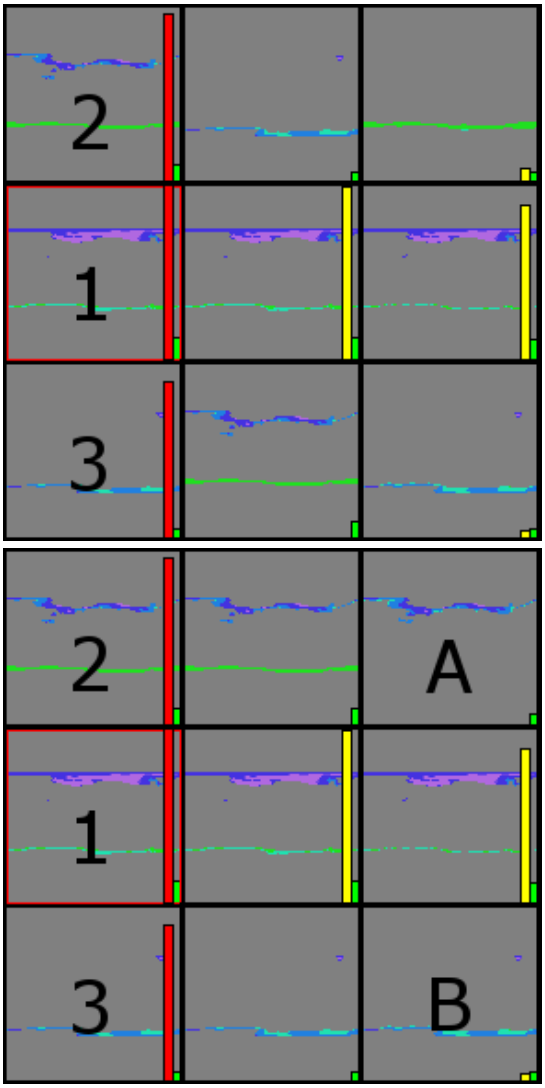

Fig. 9. Two sorted views of a portion of a cluster spreadsheet for 2 meter monthly temperature data of 6000 time steps, showing clusters of data points that have similar (left to right), 32 month, 16 month, and 8 month trends. In the top image, the user has picked the cell (1). Cells in the same column are sorted by the centroid time curve distance to the selected cell, so that clusters (2) (3) of similar temperatures over time have moved closer to the picked cell. This is also indicated by the length of the red bar in the cells. Cells in the other columns are sorted, such that the greater population overlap it has with cell (1), are moved vertically closer, but stays in the same column. This can also be quantitatively seen by the length of the yellow bar in the cells. In the second sorted image, the difference is that cells in row (A) have the highest overlap with cell (2) and cells in row (B) have the highest overlap with cell (3).

interest, like we have done in Figure 9. Alternatively, we can use similarity metrics to automatically cull cells or reduce the number of cells. Clustering across different frequency bands results in clusters with different populations, but there will still be overlap in the population of data points. Like in Figure 9, there is a shift in cluster population indicating a shift or change in temporal trends, but many of the clusters have similar populations across temporal scales. By culling clusters that have similar populations, but retaining ones with different populations, we reduce the complexity of the spreadsheet but preserve the information.

1) Cell Culling: After the user selects an initial set of clusters, instead of showing clusters that might be considered redundant because they have similar data point populations, cells can be culled if the cluster population does not exceed a percentage population difference threshold from the closest cluster population in the selection set. This percentage population cross-over count is maximum of the size of the intersection set between two clusters A and B divided by A for all A in the user selection set U, $MCOC(B) = max(\forall A \in U : |A \bigcap B|/|A|)$. This is performed incrementally in a greedy manner, adding new cells to the user selection set U as they exceed the population threshold. An example can be seen in Figure 10. Cell culling tends to be able to discard low wavelet coefficient clusters, because clusters tend to be similar over temporal scales. On the other hand, high wavelet coefficients, by their very nature, retain the detail information of every temporal scale, and thus are unlikely to carry the same cluster population over temporal scales.

2) Cell Merging: Additionally, we can use the distance metric used in the clustering algorithm to merge cells in the same column based on the centroid difference. Cell merging is a type of user based clustering to reduce the screen area occupied by the spreadsheet. If two clusters have similar centroids, such that the distance between two cluster centroids is under a threshold, we merge the cells to an overlay cell. The user can manually merge cells together to form a cell union if she decides that the data is similar, or belongs together. All of the centroids merged together are rendered in the overlay cell, and the rendered thumbnail of the clusters is a spatial union of the combined cluster data. The user is allowed to re-split the clusters into individual cells of their own, if she only wishes to pick one cluster or to see the data of each cluster individually.

D. Time Histogram Spreadsheet

When the user selects a cluster, it is added to a secondary spreadsheet, a *time histogram* spreadsheet, as in Figure 11. The column layout is the same, such that there are two halves corresponding to low and high wavelet coefficients, and each column corresponds to one frequency band. Initially, the time histogram spreadsheet is empty. When a cluster is selected, it is added as a new row to the time histogram spreadsheet, and the time histogram of the cluster across frequency bands is displayed in each column for a row. The time histogram in each frequency band provides a summary of the time curves that are contained within a cluster, so that the user can see the details of distribution of values over time. If there is more than one row, the spreadsheet is resorted to show the relevance between clusters, as was mentioned in the previous section. Essentially, the time histogram spreadsheet is to display details for the selected clusters. It also provides a good interface for the user to be able to make fine tuning adjustments of the data points within a cluster, as described below.

1) Time Curve Brushing and Linking: Our manipulation of the data contained within a cluster is a brush widget. There are two modes of operation with the brush widget. In the first mode, the user paints an area in time histogram she wants to explore, and time curves that pass through the selected bucket on the time histogram are selected. Any time curves that fall within the user painted area are selected, drawn as poly lines, and linked [4], [25] across the time histogram columns, seen in Figure 6. By drawing the curves as continuous poly line rather than plotting the data points, the change in value over time becomes more apparent. The user can make refined selections by using intersection, union, and

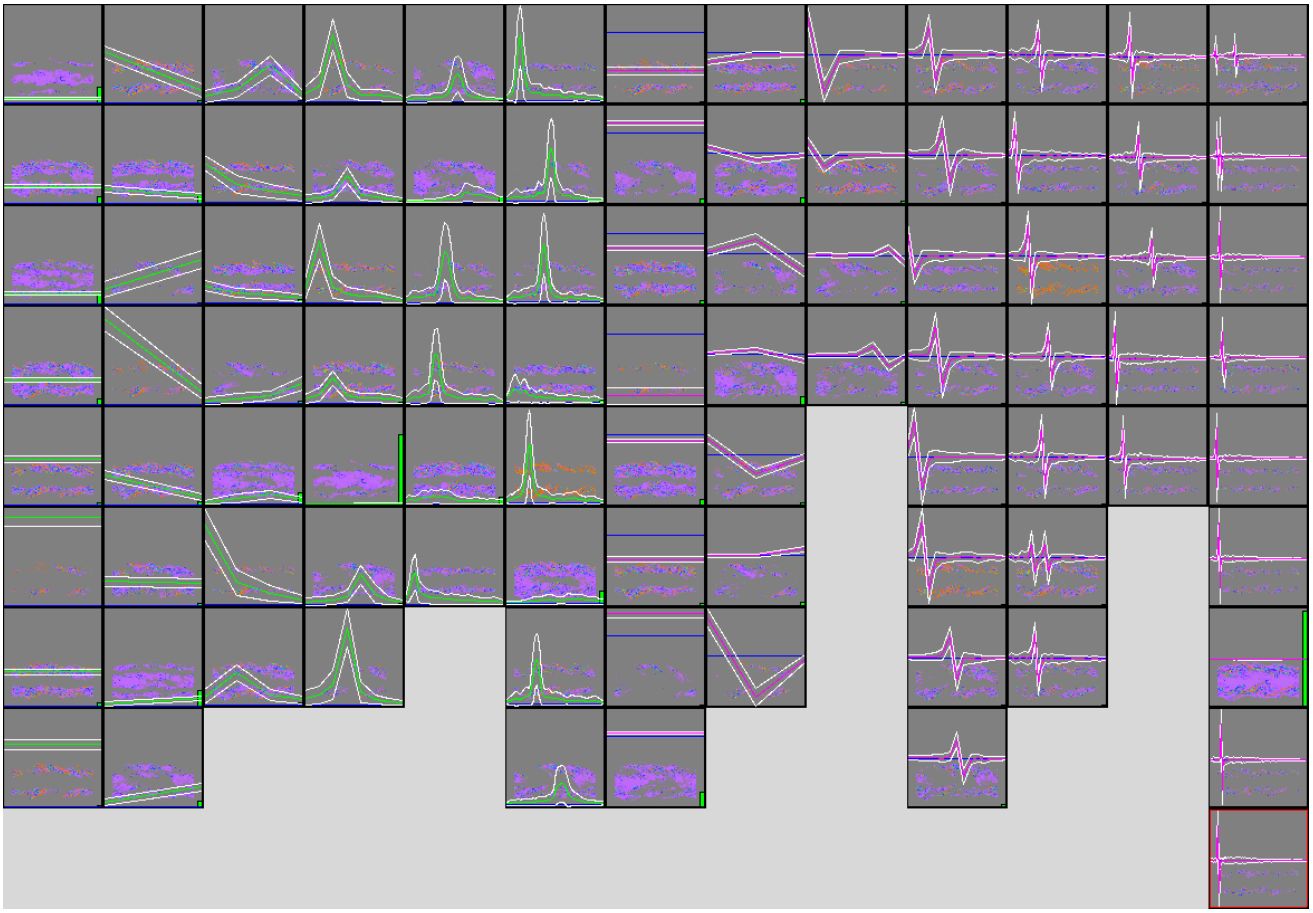

Fig. 10. An example of culled spreadsheet of χ variable from 122 time step combustion data set. The culling order was performed from highest detail to lowest detail at a 75% population threshold. 35 cells have been culled out of 126, resulting in a 27% space savings. By culling cluster cells that do not change populations over temporal scales, the essence of the temporal trends in are retained, and even highlighted since they are the only cells that remain.

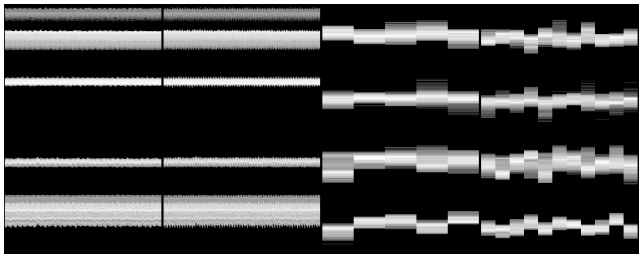

Fig. 11. An example of time histogram spreadsheet from 2 meter atmospheric temperature data of 6000 time steps. Four clusters have been selected, ranging from top to bottom. Two time histograms showing the low wavelet coefficients are on the left, and two time histograms showing the high wavelet coefficients are on the right. The multiscale aspect is very useful in hard to detect trends, such as long term trends, as can be seen in this example. The low wavelet short term coefficients look to be flat over time, but high wavelet long term coefficients reveal that there are changes in the time curve.

difference brushes, such that the different brushes remove or add data points from the selection set based on the operation of a brush. By linking time curves across frequency bands, the user can see the temporal profile of the data, and potentially make fine tuning adjustments in another frequency space. Brushing and linking is restricted to the data within a cluster or a merged cluster. To make clear the relationships between clusters, the user can use spatial boolean operators to combine clusters into one visualization.

2) Dynamic Transfer Function Brushing and Linking: The second mode of operation uses value space selection [1], [2], rather than time curve selection. By painting on the time histogram with color and opacity brushes, the user can create a temporally dynamic transfer function. She paints the value of the transfer function over time, by using the time histogram as a guide for value ranges. The paint is linked across all time histograms within a row, such that if the user paints in another frequency band, the transfer function is updated. The paint can also be linked across rows, such that the transfer function can be shared across several clusters.

While the former time curve selection method is more useful for temporal trend selection, the latter method is more useful for value range tracking. Both modes can be combined together to make temporal trend selections and value range tracking. The system can also generate dynamic transfer functions through a semi-automatic method. The user specifies a static transfer function with a center value, and the system applies the transfer function centered around the cluster centroid over time. This allows the user to track the centroid value and the variance from the centroid over time, within a cluster, like in Figure 12.

VI. Additional Exploration Schemes

By selecting clusters and refining the clusters through the time histogram spreadsheet, the user narrows the set of data points to

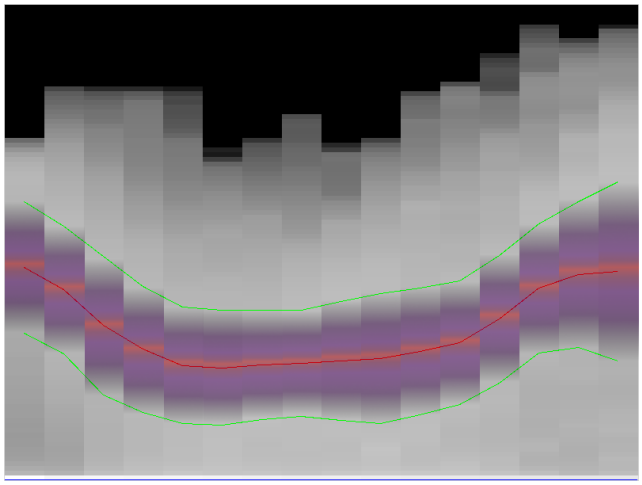

Fig. 12. A cluster of low wavelet coefficients from the OH variable of a 122 time step combustion data set. An automatic dynamic transfer function is generated that is centered around the cluster centroid. A transfer function of this will track the centroid value over time, and measure the variance of the data around the centroid.

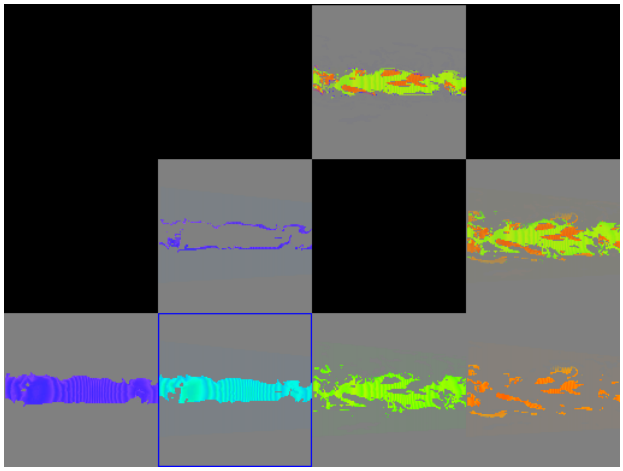

Fig. 13. An example of performing a visual query with selected clusters from 2 meter atmospheric temperature data of 6000 time steps. The composition spreadsheet with volume tree operators allows us to combine clusters to drill down into the data. The two bottom left clusters are from different frequency bands, and an XOR operation is used to find the outliers that are not in common with either. The bottom right clusters are from the same frequency band, so they do not intersect in space, but we can join them together with an ATOP operation [30]. Then, the temporal outliers between the two operations can again be found by taking the XOR of both results.

a small set that exhibits some temporal behavior. Time curves map to data point positions, and there is spatial overlap between selected clusters across temporal scales. This is because each data point is clustered independently on each of its wavelet vectors, and a data point may be selected multiple times across cluster selections. This behavior is desirable, because by having cluster sets that have intersecting members, the user can create visual relationships between cluster sets based on their temporal activity.

Each user refined cluster is a set of data points that share temporal behavior, the user can create a visualization composing relationships between temporal activity clusters. Assuming we have a pre-defined static transfer function or the user can define a dynamic transfer function per cluster in the previous section, we can operate on selected clusters to create a visual query [30].

By using the spreadsheet interface to select clusters and the time histogram to refine those clusters, the user forms an operator tree to compose clusters into a temporal query. Intersection operations are used to find data points that share trends in two clusters. Union operations are used to join data points together that potentially have different trends. Difference operations are used to find key differences between clusters, such as finding outliers to two trends. An example of this is found in Figure 13 using the 2 meter atmospheric temperature data.

Additionally, within a time segment, the user can create value and trend highlighting through temporal operators [29], [31]. In the time histogram spreadsheet view, the user can select individual time steps or a run of time steps to be operated on over time. By providing temporal operators, the user can compose several time steps into one time step that has derived data to highlight the data value trends that are present in that time segment, and to extract value differences or similarities.

A. Multivariate Interaction

Additional variables can be easily added to the system by adding spreadsheet panes, such that one variable takes up one pane in the spreadsheet. To avoid cluttering the screen in the cluster spreadsheet, clusters from different variables are not able to cross panes into other variable's spreadsheet space. In addition, cluster selection and spreadsheet reorganization is limited to the space that one variable takes in the spreadsheet since clustering and spreadsheet reorganization is for temporal relationships within the variable. However, after clusters are added to the time histogram spreadsheet, the user is allowed to move rows around to compare trends between clusters from different variables. Even though brushing and linking is limited within one variable, the way interactions are created are through boolean operators to intersect, union and difference the data between variables. By saving the operations until the end, the user can make adjustments in the temporal trends per cluster, per variable and see the final results after operation, rather than co-mingling the operations and selection together, resulting in a convoluted process.

B. Animation Rescaling

In addition to being able to operate on clusters to create temporal relationships, we can also make the temporal trends more apparent in animation. Our goal is that if there is a slow trend, we want to speed up the animation so that the trend is more apparent. Conversely, if there is a temporal trend that occurs relatively too fast, we want to be able to slow down the animation. In either case, we are trying to normalize visual activity, such that it is more apparent to the user [23].

In our time histogram spreadsheet interface, the time runs horizontally left to right in each cell. There is a time base mapping such that the animation runs at r time steps per second and f frames per second, where the animation runs at r/f time steps per frame. Additionally, the spatial layout of the spreadsheet maps x horizontal pixels per time step. Given these conditions, the user can dynamically rescale the animation speed so that it changes how fast or how slow changes appear to the user during animation, by the multiplication of a speedup factor.

From observing the slope of selected time curves or the centroid curve, the user can infer the rate of change that will take place in the animation. If it is too steep, she may wish to slow down

the animation, likewise if a slope is too shallow, she may wish to speed up the animation. By marking two points on the time axis, the user can stretch the time axis, by pulling the points away from each other. Time curves will be stretched so that slopes are flattened. This will dilate the animation time between the two points, or slow down the animation between these two points. Additional frames will be inserted in the animation. Conversely, by dragging one point towards the other on the time line, the animation time base will be compressed. Frames will be dropped in between two points during animation, and the time curve slopes will be steeper in the interface.

The speed up factor is determined by measuring the distance between the old distance on the time line and the new distance. If the old distance between time a and b on the time line is o pixels and the new distance is n pixels, the speedup factor is o/n between the two points on the time line. To remove abrupt instantaneous changes in animation speed, we give the user the ability to add automatic ease-in and ease out. A simple sine wave interpolation, between the base rate r/f and $(r/f)*(o/n)$, is used in the ease-in and ease-out ranges to speed up or slow down the animation in a smooth manner. In order to indicate the time base mapping during animation, a time line and time stamp are embedded in the animation. In this way, the user has context of when events are taking place, and how much the animation is being sped up or slowed down.

Given our previous assumption, we can apply an automatic time rescaling scheme, to reschedule the animation frames to have dynamic speed-up and slow-down to highlight all of the temporal changes equally. The user provides an absolute value rate of change optimization parameter p, and a centroid vector or the average of multiple centroids to optimize. We dynamically speed up and slow down the animation so that apparent rate of change, of the given cluster centroid, matches the user given value rate of change over real time. This is assuming that the rate of change is data value based, but the rate of change can also normalize in color or opacity space. The animation speedup factor between time step t and $t+1$ is $p/H(t)$, where $H(t)$ is the high-frequency wavelet coefficient at time step t of the centroid H, and p is the optimal rate of change. The user can specify a maximum acceleration or deceleration parameter to clamp the rate of change, so that animation speed will not increase or decrease too abruptly. Included with the supplemental material are two earthquake movies that showcase automatic frame rate change based on the centroid curve. The visible rate of change increases over time, so the animation slows down to compensate for the visual change, and highlight rapid change, while it speeds up when there isn't much change happening.

VII. Case Study: Combustion

In this section we present a sample usage of our system to a turbulent combustion data set at a 480 x 720 x 120 grid simulation, 122 time steps, with multiple variables, provided by Dr. Jacqueline Chen of Sandia National Laboratory through the SciDAC Ultravis Institute. When exploring the OH variable in the cluster spreadsheet, there is a cell in Figure 8, that is different from the other clusters in value over time. Most data points at a glance have an upward sloping trend over time, while this particular cluster starts high value, decreases, and then increases at the end of the time series. This trend can be seen as well in

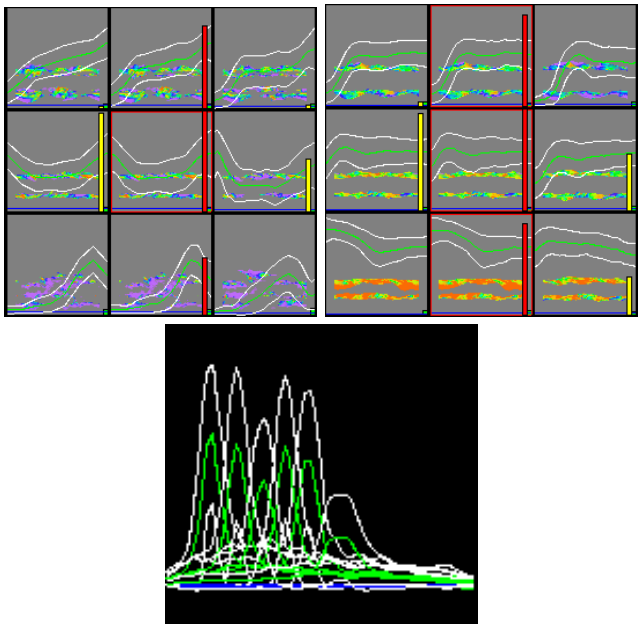

Fig. 14. The left image is a portion of spreadsheet of the OH variable. The right is a portion of spreadsheet of the mix fraction variable over time. The bottom image is an overlay cell from the χ variable, showing a value moving through space over time.

the left image of Figure 14. All of the clusters can be seen in Figure 3, and how they spatially relate to one another.

The center cell is the cell of interest the left image of Figure 14, and we resort the spreadsheet to show relevance to the cell we are interested in. When looking in the same column, we see that there are no other cells that have the similar temporal trend in that frequency band. Additionally, the thumbnail rendering shows that the data points of the cluster form a well defined structure. We can also see this through the cross-over-count information, by the yellow bar, is steady across temporal scales.

If we inspect another variable, we can possibly correlate the spatial area to another temporal trend. A portion of the mix fraction spreadsheet can be seen in the right image of Figure 14, we look for areas that have the same spatial area or correlated curves. There appear to be several clusters that appear to have similar spatial volumetric occupancy, from the visualization thumbnails, where OH and mixfrac have temporal clusters in the same area of the data. We can also see this from Figure 3.

We also look at the χ variable to see if there are any trends that might be related to OH and mix fraction, but there aren't any clusters that appear to be related. Though, one thing that is of interest is the pulse train that is in many of the frequency bands in the short term trends, seen in Figure 10. Even though we are not able to perform spatial value tracking, we are still able to detect visually from the spreadsheet some value movement through space as a temporal trend of a pulse. Our method excels at the detection of event start and ends, so we can precisely define the area and time that the value movement started by the one cluster at the head of the pulse train. By combining all of the cells together that have the pulse into an overlay cell, as in the bottom image of Figure 14, we are able to see that the pulse forms a moving value in space over time. Each cluster is an area that the value moves through. Potentially, the area of the overlay cell coincides with the data in OH and mix fraction.

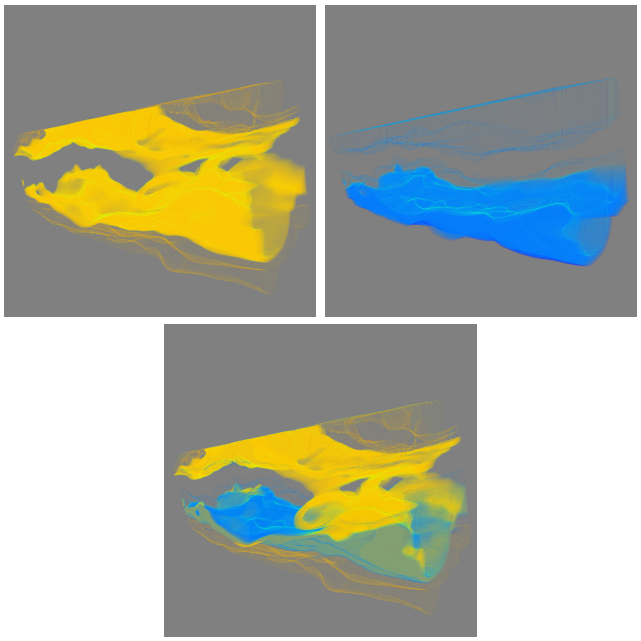

Fig. 15. The left image shows a temporal cluster from the OH variable of the combustion data. The right image shows a temporal cluster from the mix fraction variable. The bottom image is a projection or combination of the two variables in the same space to show coincidence and overlap. The bottom volumetric region the two clusters from the two variables overlap with each other and share similar structural features.

When we intersect the OH cluster with the one of the mix fraction clusters, seen in the left and right images of Figure 15, using projection operators, there is a spatial overlap and structural coincidence, in the bottom image. They mostly overlap with each other, and this is a strong indication that those data points have a correlation in the two variables for those data points. The χ clusters that are extracted do not seem to have value correlation or spatial correlation with the OH and mixfrac. We can not say that χ is temporally correlated from the information that we have, and may be indepedent of those two variables.

VIII. Case Study: Climate Modeling

The Community Climate System Model 3.0 from the National Center for Atmospheric Research is a climate model for predicting past, present and future climates. The particular data set we use is a 6000 time step series of world wide 2 meter monthly atmospheric temperature on a 256 x 128 2D grid. This multi-scale temporal methodology works quite well with climate model data and in particular for the large number of time steps in the CCSM. By using the multi-resolution wavelets to filter time, we can see long term trends that might otherwise be obscured. The high wavelet coefficients are well suited to show the activity that are present in long term trends.

For example, with CCSM, the time series is 6000 time steps and local temporal feature has little effect on the long term picture. While looking at the time series at the original resolution, it is just not possible to see large temporal features. The top image of Figure 16 shows clusters of large scale temporal features in the low wavelet coefficients. There does not seem to be any activity in the data. If we look at the high wavelet coefficients, found in the bottom image, a different picture emerges. In this example, we can see that all areas in the climate model have a change in temperature over the first 3000 months and over the next 3000 months. Due to the scale of the data, the high wavelet coefficients make it easy to detect changes in value.

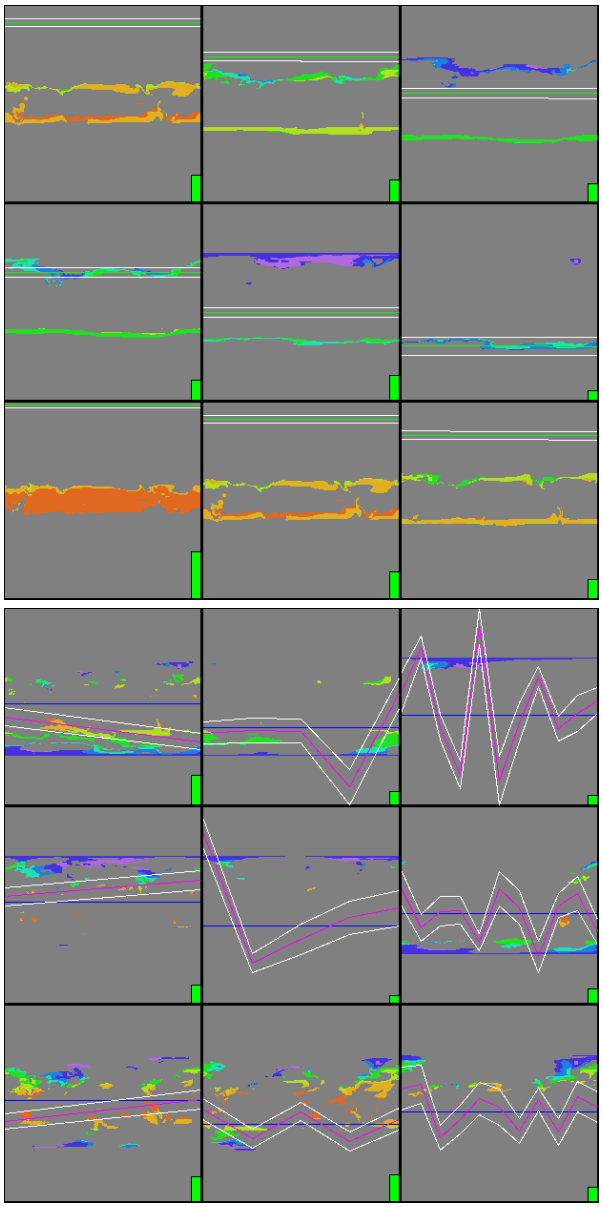

Fig. 16. Two spreadsheet portions from the climate model data set. The left image displays several clusters from the low wavelet coefficients. The time series is 6000 time steps long, and even after wavelet analysis, the trends are not visible because of the value scale and length of data. The right image shows clusters from the high coefficient clustering of the same data set. Here we can see that there are long term trends in the data, as the cluster data tells us there is a rate of change over the long term in time.

For large time data such as this, it would not be feasible to use traditional animatation to see long term trends, unless we were to use time rescaling. Even then, if the data was not smoothed, the high frequency noise of speeding up the animation may make it difficult to see the long term trend. Our filtering and clustering method is able to remove the short term trends, and display the long term summaries that are present in the data.

We also have previous examples, Figure 2, which show the ability to classify geographical regions based on seasonal temperature activity. Additionally through multi-scale temporal filterting,

we can classify geographical regions into hemispheres or ignore hemispheres differences, depending on the seasonal phase shift. A example of this can be seen in Figure 9 with the row that contains cell A. Cell A is a cluster from 8 month trends, so there will be differences in the temperature between the northern and southern hemispheres due to seasonal phase shift between the hemispheres. Cell A contains cluster of data points that share similar temperatures in the northern hemisphere. The cells to the left of A are at 16 and 32 month trends, the hemispherical differences are not seen at this time scale, and they have data points in both hemisphere that share the same temperature at that time scale.

IX. Case Study: Earthquake

The earthquake data was created by the TeraShake 2.1 simulation Southern California Earthquake Center, provided by the San Diego Supercomputer Center. It is a 226 time step series on a 100 x 375 x 750 grid, where we used a computed velocity magnitude scalar as the data. Clustering is able to extract the regions that have different wave characteristics due to different geology. This is because the basins that amplify earthquake waves have different temporal behavior compared to other data points. By clustering in time, we are able to find the basin because it behaves differently compared to the surrounding geographical region, and therefore we are able to isolate it.

The top image in Figure 17 shows the entire data in the time sequence. The second image is the cluster that was extracted from the data through temporal activity clustering. As we can see from comparing the two images, we are able to isolate just the area that corresponds to the basin, because those data points share the same temporal activity, while the other surrounding data points have different activity, and therefore, they are not clustered with the basin. Given traditional methods, a user would have to watch an animation to deduce this different activity, and even then, she may not be able to isolate the data points quite as precisely as this. The accuracy depends on the transfer function and animation and the ability of the user to notice the visual activity difference, while our method was able to automatically find the different time activity that comprises the basin.

In visualizing the earthquake, the phenomenon is a burst of wave activity in the basin, and therefore we can use our animation time scaling to slow down and emphasize the activity in that time period. Animations in the supplemental material show an earthquake shockwave coming into the basin. Initially, there isn't much activity happening in the basin, so the animation runs faster. As soon as the earthquake starts to happen, the animation begins to slow down, so the user can see the temporal details. Once the shockwave and activity burst has passed, the animation speeds up once more.

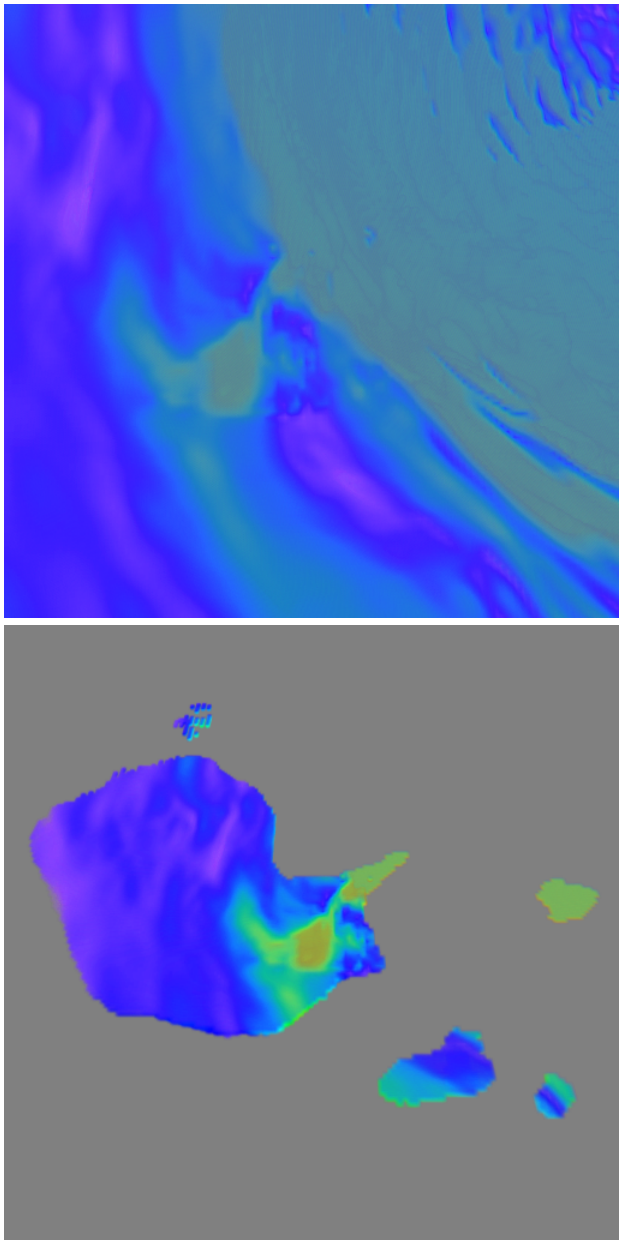

Fig. 17. Two images from exploring the earthquake data. The top image is the entire data, before wavelet transformation and clustering. The second image shows the basin extracted from the surrounding data. The basin has a different temporal behavior from the rest of the data, and thus is able to be separated from the temporal activity background.

X. Conclusion

We have presented a methodology for exploring time series data by focusing on temporal trends. Our goal was to locate data points of similar temporal trends across multiple time scales. To achieve this, we apply the wavelet transform to data along time, to create a multi-resolution temporal representation. Then, we cluster the data in the different temporal scales and wavelet coefficient types to derive groups of similar trends. These trends are then shown to the user, who can browse the trends present in their data, select and interact with the data, and eventually visualize the explored phenomena.

The temporal clusters are shown in a visualization spreadsheet which summarizes the temporal and cluster content of the data. From there, the user can choose clusters to explore, which will be placed on a secondary time histogram spreadsheet. The time histogram allows the user to see the value distribution over time, and also make adjustments to the data in the cluster through brushing and linking. Selected clusters are then used in the final visualization, where the user can perform boolean operations on the data and animate the temporal trends.

We believe that the proposed method and system is useful for exploring data in a time centric manner, rather than focusing

on space and value. There is room for improvement, though. In particular, the clustering method is more appropriate for spatially static data, such as weather climate data or temporal hotspots such as the earthquake basin. In the future, we would like to extend the method to be able to deal with spatially moving time activity or temporal phase shift, such as is seen in the combustion or earthquake data. Secondly, the user interface needs to be able to adapt to the data explosion from extracting the multiple temporal scales and clusters. As can be seen with the cluster spreadsheet, even with cell culling, there can be many cells that can overwhelm the user. Additional metrics, controls, or cues to highlight potential interesting time activity would be useful to reduce the amount of data that is shown to the user.

REFERENCES

[1] H. Akiba, N. Fout, and K.-L. Ma. Simultaneous classification of time-varying volume data based on the time histogram. In *Eurographics Visualization Symposium*, pages 1–8, 2006.

[2] H. Akiba and K.-L. Ma. A tri-space visualization interface for analyzing time-varying multivariate volume data. In *Eurographics/IEEE VGTC Symposium on Visualization*, 2007.

[3] E. H.-H. Chi, J. Riedl, P. Barry, and J. Konstan. Principles for information visualization spreadsheets. *Computer Graphics and Applications*, 18(4):30–38, 1998.

[4] H. Doleisch, M. Mayer, M. Gasser, R. Wanker, and H. Hauser. Case study: Visual analysis of complex, time-dependent simulation results of a diesel exhaust system. In *VisSym*, pages 91–96, 343, 2004.

[5] Z. Fang, T. Möller, G. Harmarneh, and A. Celler. Visualization and exploration of spatio-temporal medical image data sets. 2007. Proceedings of Graphics Interface 2007.

[6] E. Hadjidemetriou, M. D. Grossberg, and S. K. Nayar. Multiresolution histograms and their use for recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(7):831–847, July 2004.

[7] J. A. Hartigan and M. A. Wong. Algorithm as 136: A *k*-means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979.

[8] J. Heer and M. Agrawala. Multi-scale banking to 45 degrees. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):701–708, 2006.

[9] H. Hochheiser and B. Shneiderman. Dynamic query tools for time series data sets: Timebox widgets for interactive exploration. *Information Visualization*, 3(1):1–8, 2004.

[10] P. Jain and S. N. Merchant. Wavelet based multiresolution histogram for fast image retrieval. In *TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region*, volume 2, pages 581–585, 2003.

[11] T. Jankun-Kelly and K.-L. Ma. Study of transfer function generation for time-varying volume data. In *Proceedings of Volume Graphics 2001 Workshop*, pages 51–65, 2001.

[12] T. Jankun-Kelly and K.-L. Ma. Visualization exploration and encapsulation via a spreadsheet-like interface. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):275–287, July 2001.

[13] T. Jankun-Kelly and K.-L. Ma. A spreadsheet interface for visualization exploration. In *Proceedings of the conference on Visualization '00*, pages 69–76, 2007.

[14] J. Kniss, G. Kindlmann, and C. Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Visualization, 2001. VIS '01. Proceedings*, pages 255– 562, 2001.

[15] T. Kohonen. *Self Organizing Maps*. Springer, 2001.

[16] R. Kosara, F. Bendix, and H. Hauser. Timehistograms for large, time-dependent data. In *Proceedings of the 2004 Eurographics/IEEE TVCG Symposium on Visualization*, pages 45–54, 340, 2004.

[17] M. Levoy. Spreadsheets for images. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 139–146, 1994.

[18] J. Lin, E. Koegh, S. Lonardi, J. Lankford, and D. Nystrom. Visually mining and monitoring massive time series. In *Tenth ACM SIGKDD*, pages 460–469, 2004.

[19] L. Linsen, V. Pascucci, M. A. Duchaineau, B. Hamann, and K. I. Joy. Hierarchical representation of time-varying volume data with "4th-root-of-2" subdivision and quadrilinear b-spline wavelets. In *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, page 346, 2002.

[20] E. Lum, J. Shearer, and K.-L. Ma. Interactive multi-scale exploration for volume classification. In *Pacific Graphics 2006 Conference, also as a special issue of Visual Computer*, pages 622–630, 2006.

[21] K.-L. Ma. Visualizing time-varying volume data. *Computing in Science and Engineering*, 5(2), 2003.

[22] B. Petersch, M. Hadwiger, H. Hauser, and D. Hönigmann. Real time computation and temporal coherence of opacity transfer functions for direct volume rendering of ultrasound data. *Computerized Medical Imaging and Graphics*, 29(1):53–63, 2005.

[23] D. J. Simons and M. S. Ambinder. Change blindness. theory and consequences. *Current Directions in Psychological Science*, 14(1):44–48, 2005.

[24] G. Strang and T. Nguyen. *Wavelets And Filter Banks*. Wellesley-Cambridge Press, 1st edition, 1996.

[25] D. Swayne, D. T. Lang, A. Buja, and D. Cook. Ggobi: evolving from xgobi into an extensible framework for interactive data visualization. *Computational Statistics and Data Analysis*, 43(4):423–444, 2003.

[26] F.-Y. Tzeng and K.-L. Ma. Intelligent feature extraction and tracking for visualizing large-scale 4d flow simulations. In *SC '05: Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, page 6, 2005.

[27] J. J. van Wijk and E. R. van Selow. Cluster and calendar based visualization of time series data. In *INFOVIS*, pages 4–9, 1999.

[28] M. Weber, M. Alexa, and W. Muller. Visualizing time-series on spirals. In *INFOVIS*, pages 7–14, 2001.

[29] J. Woodring and H.-W. Shen. Chronovolumes: A direct rendering technique for visualizing time-varying data. In *Proceedings of the 2003 Eurographics/IEEE TVCG Workshop on Volume Graphics*, pages 27–34, 2003.

[30] J. Woodring and H.-W. Shen. Multi-variate, time varying, and comparative visualization with contextual cues. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):909–916, 2006.

[31] J. Woodring, C. Wang, and H.-W. Shen. High dimensional direct rendering of time-varying volumetric data. In *Visualization, 2003. VIS 2003.*, pages 417–424, 2003.