A Framework to Visualize Temporal Behavioral Relationships in Streaming Multivariate Data

Shenghui Cheng, Klaus Mueller
Computer Science Department
Stony Brook University
Stony Brook, USA
{shecheng, mueller}@cs.stonybrook.edu

Wei Xu
Computational Science Initiative
Brookhaven National Laboratory
Upton, USA
xuw@bnl.gov

***Abstract*— Big Data analysis for scientific data is extremely challenging due to the following features – high resolution, extreme scale, high acquisition rate, multivariate data format and aggregating in the streaming fashion. Therefore, a visual analysis tool that can process, reduce, manipulate and display extreme-scale data is critical for scientists to make the right decision on-site and adjust their measurement strategies during the experiment. The lack of these tools not only severely reduces the scientific throughput, but also impairs our capability for scientific discoveries. In this paper, we describe StreamVisND – an interactive framework that provides several linked displays designed to reveal multivariate temporal behavior patterns from various perspectives. All of these displays generalize standard visualization paradigms such as line graphs from time samples to time intervals. As such the integral data type of our application is the time interval which we represent as a vector of time samples. Relationships of time intervals are expressed as similarities, possibly warped over time, of pairs of time vectors. These similarities can be among different variables at the same time interval, or different time intervals of the same variable. The former results in a line graph of streaming variables, while the latter results in a new display we called illustrative transform lines of time intervals over the variables. For both displays since the comparative metric is now pairwise similarity, as opposed to absolute value, we require an optimization algorithm, such as multidimensional scaling to perform mapping into display coordinates. Additional displays include a 2D embedding of temporal snapshots of the variables, as well as a 2D embedding of temporal relationships changes among the variables. We demonstrate our system in an environmental pollution diagnostics setting and have obtained encouraging results.**

Keywords— information visualization; streaming data; high dimensional data; time-series; embedding

I. INTRODUCTION

State-of-the-art scientific facilities, such as the NSLS-II, generate high-resolution data streams at an aggregated rate of Giga-bps. The enormous size of the data makes their exploration, analysis, and summarization extremely challenging; yet only with extensive analysis can deep scientific insights be extracted. Currently available tools were not designed with these massive datasets in mind. As a consequence, analysis remains largely manual, burdening users with tedious manipulation tasks that are both inconvenient and error-prone. Therefore, an on-line visual analysis tool that can process, manipulate and display extreme-scale data is critical for them to make the right decision on-site and adjust their measurement strategies during the experiment. The lack of these tools not only severely reduces the scientific throughput, but also impairs our capability for scientific discoveries. The new framework we propose is designed to tackle these challenges. Via data visualization and interactive visual analytics it will provide targeted information to users in real time, enabling timely assessment and decision-making as well as more rapid discovery of subtle trends.

Beside scientific domain, streaming data have become ubiquitous in recent years and arise in many aspects of our daily lives. Examples are social networks, stock tickers, pollution measurements, security feeds, economic trends, credit card transactions, computer networks, science experiments, and abundantly more. Streaming data embrace all of the 5V of big data – volume, velocity, variety, veracity, and value. They are real-time time-series data that afford real-time responses to the measured phenomena. This provides unprecedented opportunities for businesses, government, first responders, and scientists to react to emerging trends. While some of these responses can be automated, it is still desirable to also insert a human into the loop, not only to prevent catastrophic outcomes, but also to add critical human expertise and intuition into the process. Whenever a human is involved in data science applications, visualization and visual analytics can play a critical role. They afford a highly effective gateway to a human's creative faculties and they also have a high propensity of keeping the human expert engaged and alert. We describe such an interface in this paper.

Pressing issues in streaming data are (1) the one-pass constraint – the data need to be processed in-stream and not all can be stored, (2) concept drift – the statistical properties of the derived predictive model keep changing continuously in unpredictable ways, and (3) concept evolution – new features can appear in the predictive model which may either supplement or outdate existing features. Additional complexities arise from the fact that just like other (big) data, streaming data is also often multivariate. There are many economic factors – not just one – and there are also many stocks, security metrics, environmental pollutants, and so on. An effective visual analytics system must be able to deal with all of these issues. The framework we propose, StreamVisND, particularly addresses the multivariate nature of streaming data,

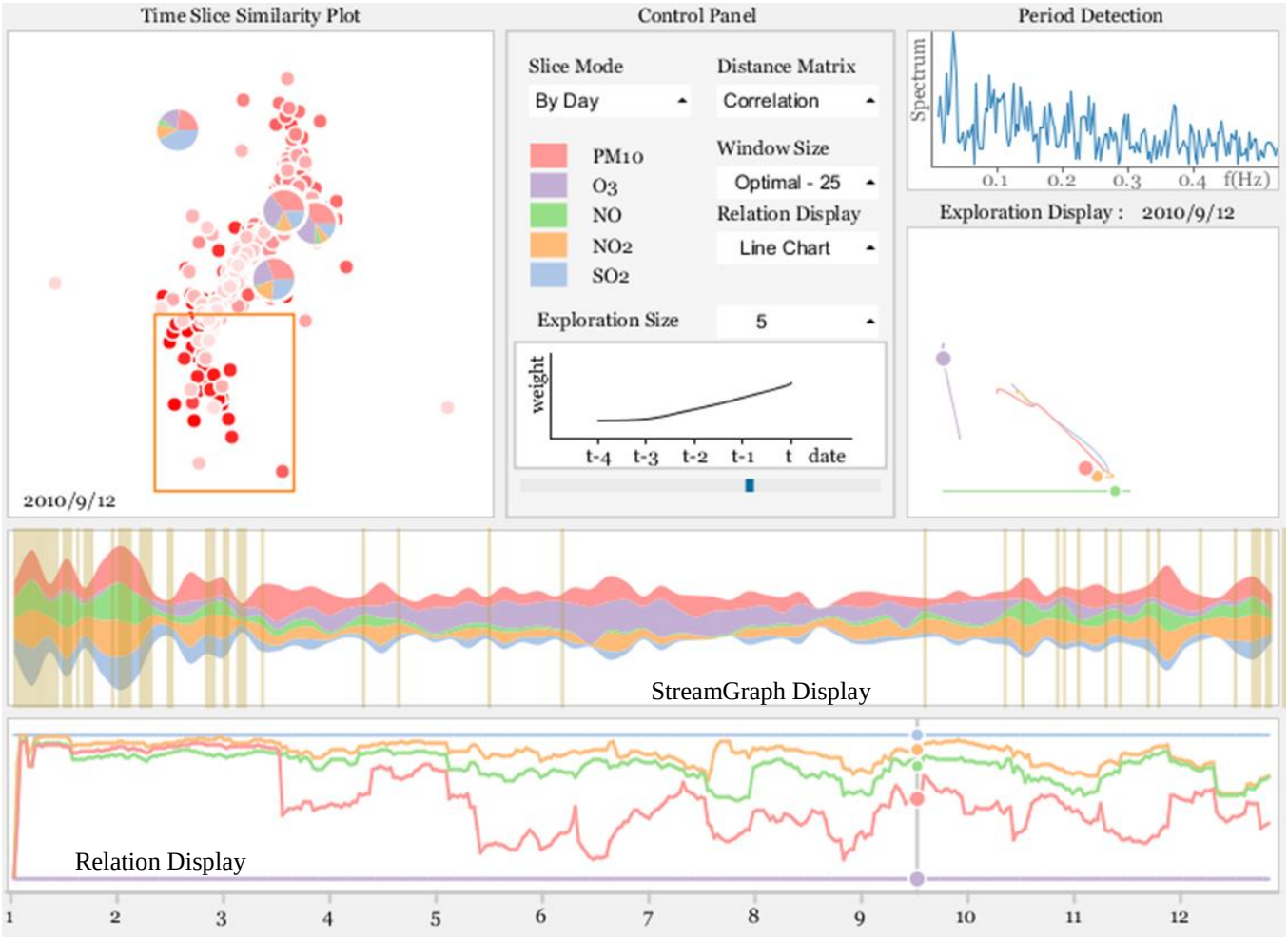

Fig. 1. Interface of our StreamVis[ND] prototype. It consists of five parts – stream graph illustration, slice similarity plot, relation display, window transforms demonstration (with weight function) and configuration control panel. The relation display highlights the relation change on 2010/09/12 in the blue rectangle and exploration display traces the local change of variables at a smaller exploration size. The orange rectangle selects some time slices so that they are highlighted in the stream graph with the slice bands.

but is also responsive to the other complexities mentioned above.

Our system operates on subsequences of the time-series data which we call time intervals. This allows us to compare not just time values but time behaviors, which is aptly more informative and salient in the context of time-variant data. We represent each such time interval as a high-dimensional vector, one for each variable, and then define a set of distance metrics that can be employed to gauge similarity of behavior. Essentially, our framework allows users to recognize temporal co-movement of variables which can give important hints on relationships that may exist among these variables. At the same time, we can also compare the time behaviors of one or more variables exhibited at different points of time, for example, to map events of the past into the present for special analytics tasks.

Our framework essentially generalizes the unit of measure from time point sample (TPS) to time interval sample (TIS) which is a vector quantity. We then define a set of similarity metrics for TIS after which we use standard line plots displays for visualization in the usual mode. It should be noted, however, that these displays now visualize similarity of time behavior as opposed to scalar values. As such, vertical distances are now gauged by pairwise similarity and not value differentials. While the latter requires simple ordering, the former requires optimization which we perform using the stress equation of multidimensional scaling (MDS).

The length of the time interval is obviously a critical element of our framework. We describe several methods by which good intervals for the TIS can be chosen. In addition, we also describe several similarity functions by which pairs of TIS can be compared.

Figure 1 shows an annotated screenshot of our prototype. In the center is a Stream Graph by which users can select a certain range of time slices for display in the Time Slice Similarity Plot. A time slice is just a single multivariate time point. This Time Slice Similarity Plot is an MDS plot in which more similar time slices are laid out more closely, and time is

mapped to color. We see that earlier time slices are more similar since they clump together in the center. The Temporal Attribute Relation Display is the TIS similarity line plot as discussed above. Finally, the Dynamic Local Change Plot is a temporal MDS display which visualizes the change of TIS similarity over time.

Our system can be utilized in post-hoc batch mode and within a streaming scenario. In the latter, when connected to a real data stream, all displays would be constantly updated. Reservoir-type sampling with temporal alignment could be added, but we have not done this at the current time. Our prototype interface can be regarded as a cockpit for streaming data where the user can see different aspects of temporal similarity at the same time.

Our paper is structured as follows. Section 2 presents related work. Section 3 describes the time point sample visualization and Section 4 presents our time interval visualization in details. Section 5 illustrates a case study. Section 6 ends with conclusions.

II. Related Work

Multivariate streaming data possess both multivariate and time series features. In the following we will discuss related multivariate data displays, multivariate data space embeddings, and then the extension to streaming data visualizations.

A. Multivariate Data Displays

Numerous multivariate data displays have been proposed for multivariate data visualization. Parallel coordinates [13], and its radial version, star coordinates [16], create axes either vertically or radially, and then map the data as polylines. They display the multivariate data and let users probe and capture certain patterns. However, clutter is a serious problem when the number of data items becomes large. Scatterplot matrices [7], on the other hand, combine all the pairwise scatterplots as sub-blocks and display the bivariate relations. Their growth however, is polynomial in the number of dimensions. Radviz [10], arranges the attributes as dimension anchors [11] that are equally spaced around the circle and then map the data items inside the circle via linear interpolation. However, the overplotting of the points becomes a significant problem. Generalized barycentric coordinates (GBC) [19] and its improved version [1] can cause less overplotting by reducing the layout error among the data and attributes. However, even for the improved GBC plot, its error is still high [3]. This error will only increase when the data becomes dynamic.

B. Multivariate Data Space Embeddings

Multivariate displays aim to display the multivariate data and allow users to understand data patterns. However, these displays cannot preserve the relations among the data items and attributes well. Multivariate data space embeddings can map the multivariate data space into a 2D layout while preserving the similarities.

PCA [15] takes the plane formed by two eigenvectors and projects the data into this plane. This linear mapping can preserve the similarities among the data, but it causes much distortion. MDS [17][21] takes all pairwise Euclidean distances and optimizes the data layout to preserve these distances. ISOMAP [23] and locally linear embedding [22] take neighbor structures into consideration and visualize them in the 2D plane. t-SNE [18] aims to extract clusters in the data. Our work is inspired by these space embeddings and extends them for streaming data visualization.

C. Streaming Data Visualization

The relations in the streaming data keep changing over time. The stream graph and its cousin, Themeriver, are techniques to visualize changing attributes. However, they cannot convey varying relations.

To adapt multivariate data space embeddings into streaming data visualization, Dwyer et al [1] mapped the data slices onto a 2D layout and used the third dimension to represent the time scale. This allows users to see the time axis while observing multivariate relations. However, 3-dimensional visualizations are difficult to explore. Steiger et al [20] proposed to split the continuous time-series data into fixed length segments and visualized them via self-organizing maps. This can aid users in tracking certain patterns in the similarity layout. However, they only consider the similarity of segments but ignore the attributes. In addition, they can only visualize discrete relations but users cannot assess the continuous relation transformations. Our method [1] also subdivided streams into slices, but it can display the segments and attributes relations, discrete and continuous. Jäckle [14] et al. presented (at the same time) a temporal MDS, which maps each slice into 1D via MDS. However, the un-organized MDS cannot arrange corresponding points well and makes it difficult to track certain patterns. Our technique connects the corresponding points from each slice, and generates a line to see the evolving pattern.

In order to satisfy the goal of assessing relations among data slices and attributes, and visualizing streaming data in a more relationship-centric manner, we design an interface (see Fig 1) called StreamVisND. It consists of five parts – stream graph illustration, similarity plot, relation display, window transforms demonstration (with weight function) and configuration control panel.

III. Time Point Sample (TPS) Based Visualization

The multivariate streaming data typically contain numerous (assume n) variables or attributes during a certain time period with T time stamps. Then the multivariate data can be presented as a set of time-series variables,

$$[V_1, V_2, \dots, V_n] \tag{3.1}$$

where V_i is the ith time-varying variable, attribute or dimension.

At each time stamp, the streaming data generates a record, which we call *data slice*. Each data slice S is an n-dimensional vector,

$$S_i = [x_{i1}, x_{i2}, \dots, x_{in}] \tag{3.2}$$

where x_{ij} (j=1,2,…, n) is the ith (i=1,2,…,T) slice record in the jth dimension.

In this way, the streaming data forms a high dimensional data matrix *DM* with *T* rows and *n* columns – the rows represent the data slices (*S*) at different time stamps, while the columns represent the attributes or variables (*V*):

$$DM = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{T1} & \cdots & x_{Tn} \end{bmatrix} \quad (3.3)$$

Our test dataset in this paper is an air pollution data sample of a city (A) in 2010. It contains the proportions of "SO2", "NO2", "NO", "PM10" and "O3" with 365 daily records.

Our first attempt to visualize these streaming data was to simply visualize these time point slices with a stream graph (Sec. III.A), slice similarity displays (Sec. III.B), variable similarity displays (Sec. III.C) and additional visualization techniques (Sec. III.D).

A. Stream Graph

The stream graph is a common visualization method for streaming data. It represents the values of the attributes (called themes) as vertical bars and joins them horizontally over time. This yields a display of layers – one layer per attribute – with time-varying cross-sections. While the stream graph presents streaming data continuously, it is difficult for analysts to assess, in an explicit way, the similarities of different time slices and the changing relations of the attributes over time. We have implemented this approach and plotted the pollution data in Fig. 1. From Fig.1, we could observe that NO and NO2 have high values at the beginning and ending of the year, but low values in the middle, while O3 behaves complexly opposite. The reason is simple - nitrogen oxide destroys O3. However, with the stream graph, analysts can only observe the continuous value change but it is difficult to tell the similarities or differences among different slices or variables.

B. Slices Similarity Functions

Euclidean Distance

The Euclidean distance typically measures the vector distance in Euclidean space. Suppose we have two slices $X=[x_1, x_2, \ldots, x_n]$ and $Y=[y_1, y_2, \ldots, y_n]$. The Euclidean distance $Dis_{Euclidean}$ between them is

$$Dis_{Euclidean} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (3.4)$$

This distance computes the similarities based on the actual values in the corresponding dimensions -- if two vectors have similar values in each dimension, this distance will be small and vice versa. This metric essentially provides the data distributions based on its values as shown in Fig. 2(a). The brightness is encoded according to the time stamps of the slices (the same for the following sections). In this example, each point represents one day. We observe that later days are similar and mapped to the display center, while earlier days are more dissimilar and are mapped to the periphery.

Correlation Distance

The Euclidean distance that acts on absolute value similarity is not sufficient when two vectors have similar trends but quite different values. To evaluate their similarity, we need another distance metric – correlation distance, which emphasizes the relative component similarity. This distance could be calculated via Pearson correlation. In order to consistently reflect the meaning of distance, we choose the 1-correlation as the metric:

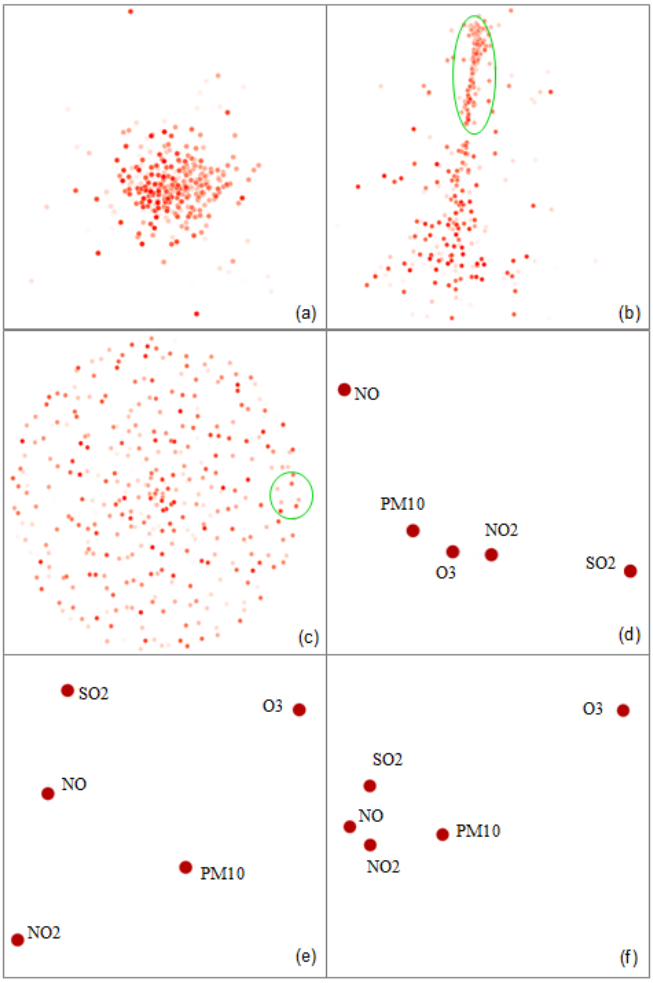

Fig. 2. Data slices and variables visualization with different distance metrics. (a) Euclidean distance (b) Correlation distance (c) SSIM distance (d) Correlation distance (e) DTW distance (f) auto-regression distance.

$$Dis_{Correlation} = 1 - \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \quad (3.5)$$

where E is the expectation, μ_X and μ_Y are the means of X and Y respectively, and σ_Y and σ_Y are the standard deviations of X and Y respectively. As indicated in Fig. 2(b), we can find the later slices on the bottom are quite scattered, which reveals that their components in terms of combination ratios of attributes are different. On the contrary, the slices labeled in the green oval maintain tight relations. This stable time period might imply, for instance, the specific pollution sources during that time.

Structural Similarity Index

The Euclidean distance and correlation distance gauge the similarity from "value" and "component" views, respectively.

However, they neglect the vector perceptual topology that human observers are most sensitive to – the mean, contrast and structure. To identify the vectors with similar topology, we utilize a structure based distance metric - Structural Similarity Index (SSIM) [12].

$$Dis_{SSIM} = [\frac{2\mu_X\mu_Y+c_1}{\mu_X^2+\mu_Y^2+c_1}]^{\alpha} \cdot \left[\frac{2\sigma_X\sigma_Y+c_2}{\sigma_X^2+\sigma_Y^2+c_2}\right]^{\beta} \cdot \left[\frac{\sigma_{XY}+c_3}{\sigma_X\sigma_Y+c_3}\right]^{r} \quad (3.6)$$

where σ_{XY} is the covariance of *XY*, α, β and γ are the parameters for mean, contrast and structure respectively. The constants c_1, c_2, and c_3 are typically small and prevent numerical instabilities when the main terms are close to zero. In this paper, we emphasize the structural aspect and make α and β equal to 0. The result is shown in Fig. 2(c), where we observe the structures of the slices are quite diverse. There is no salient structural similarity depending on the slice timestamp now. For instance, some later slices (circled in green) have similar structures with the earlier days.

C. Variable Similarity Functions

As opposed to slice based comparison, we can process the data "by column" i.e. treat each variable (attribute) as a time-series vector and compare and plot their similarities. Some metrics in the last section are still suitable such as computing the correlation distance among the variables and visualizing them by MDS. However, it no longer makes sense to compare the Euclidean distance between two attributes due to unequal measurements and normalization. New distance metrics are more desirable for gauging the difference. Therefore, we propose two different distance metrics for the variables.

Correlation Distance

The correlation distances among the variables are still valid. It shows the similarities of variable changes across time. As in Fig. 2(d), we could find we could find PM10, O3 and NO2 aggregate as a group while NO and SO2 are quite separated.

Dynamic Time Warping Distance

The correlation distance does not tolerate differences due to misalignment. For time-series variables, their patterns may not match at the exact same time stamps. Instead, some shifts may exist that will reduce the similarity between two variables. The dynamic time warping (DTW) distance [1] computes the optimal match between two sequences:

$$DTW(X,Y) = f(n,n)$$
$$f(j,i) = |x_j - y_i| + min\begin{cases} f(j,i-1) \\ f(j-1,i) \\ f(j-1,i-1) \end{cases} \quad (3.7)$$
$$f(0,0) = 0, f(j,0) = f(0,i) = \infty \quad (i = 1,\dots,n; j = 1,\dots,n)$$

A s shown in Fig. 2(e), different from the correlation distance, with the DTW distance, the distances among different variables are quite even now. They do not have very specific close relation or vice versa.

D. Similarity Visualization (MDS) and Pie Chart

With distance metrics for both slices (Sec III-B) and variables (Sec III-C), it is easy to plot the similarity of the whole dataset using MDS as shown in Fig. 1(a). The scatter plot generated by MDS often brings massive points and causes clutter. This issue usually happens when plotting data similarities. Although the "by day" displays are able to express relations among relatively small window size as a close view of data, this also results in a massive point cloud which becomes confusing when the data volume is extremely high. In order to navigate data from a broader view with various time granularities, we also provide "by week", "by month", and "by season". With this series of displays, users can explore relations hierarchically.

Besides, we provide additional plots and interactions to ease the data exploration. Specifically, the Pie Chart can show the components of all variables. Some interactions can assist users to identify the aggregated points. For this purpose, we implemented three kinds of interactions – selection, filter and pick, shown in Fig.1.

Since the MDS plot facilitates a similarity display but cannot organize data along the time axis, while the stream graph preserves the time sequence but is unable to explicitly show the similarity, connecting these two displays will benefit both aspects. Specifically, selection allows user to choose the time periods in the stream graph (used as a time reference) and the corresponding points in the similarity plot are highlighted simultaneously. Filter reverses this interaction – it offers users to draw a rectangle in the MDS plot that chooses only the points inside (meaning they are well correlated), and then highlights the corresponding slices in the stream graph. For instance, the highlighted slices confirm the fact that the beginning and end of the year are highly correlated. In addition, pick allows users to click on a point in the MDS plot to uncover a pie chart that displays the differences in terms of components among these time slices.

IV. Time Interval Samples (TIS) – Behavior

This similarity analysis is based on the unit slices – either visualizing the similarities among the slices or computing the variables' similarities based on the unit slices. With the static time point slice values, analysts cannot learn much about the evolving behavior in time intervals or windows – just encoding the time stamps as brightness is not enough. We would like to bin the unit data slices as different time-interval windows and then visualize the relation changes based on the windows.

A. Data Window

The streaming data can be divided into a series of intervals which we call data windows. Suppose we slice the original streaming data into *m* windows, thus the data matrix can be rewritten as:

$$DM = [W_1, W_2, \dots., W_m]^T \quad (3.8)$$

where each window W contains $\lfloor n/m \rfloor$ time slices. Then each variable V_i can be represented as:

$$V_i = [W_{1i}, W_{2i}, \dots., W_{mi}] \quad (3.9)$$

To assess the behavior in a time interval, numerous methods have been proposed. Taking the mean of the data

slices in the same window is a common way to measure the behavior of that interval. This is similar to re-creating a sample point with the average value. However, using average to depict a behavior is not sufficient. Essentially, we forgo the higher resolution of the original data.

B. Window Size

In order to gather enough but not redundant information, it is necessary to obtain an appropriate window size, which essentially requires period detection. The multidimensional Fourier transform is a well suited to detect the period. It can transform the time series data into a frequency display. The estimated period can be obtained when it reaches the largest amplitude:

$$\Phi(\omega_1, \dots, \omega_n) = \sum_{d_1=-\infty}^{\infty} \dots \sum_{d_n=-\infty}^{\infty} \varphi(d_1, \dots, d_n) e^{-i\omega_1 d_1 \dots - i\varphi_n d_n} \quad (3.10)$$

where Φ is the Fourier transform function, ω means frequency and φ is the multidimensional discrete-domain function that generates our time series data. The detection result is shown in Fig. 3. We could observe the frequency with largest absolute amplitude is close to 0.04, so we estimate the window size as 25.

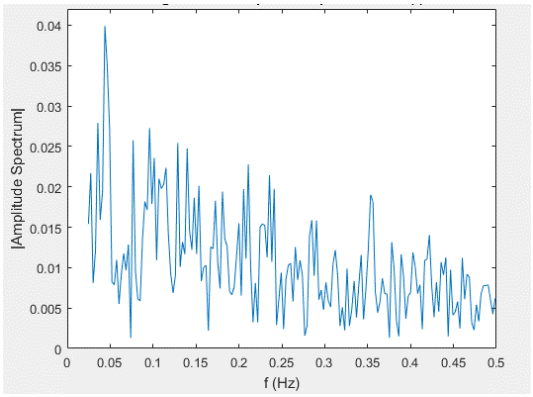

Fig. 3. Time period detection using the multidimensional Fourier transform. We find that the amplitude at 0.04 Hz is the largest Hence. the estimated window size is around 25.

C. Similarity Function

The similarity functions presented in section III.B and III.C are designed to analyze the similarity of time point slices. Beside correlation distance, we still need other metrics to evaluate window similarities.

For the time-series variables, it is significant to capture self-evolving features i.e. elements of the time-series data might be dependent on the values of previous elements. Auto regression distance [8] is one approach to recognize the linear intra-dependency of specific variables. To compare the distance of two multidimensional variables *X* and *Y*, each variable is first divided into intervals with the width of a predefined window size *w*. The last element of each interval can be then represented as the linear regression of previous elements in the same interval:

$$x_t = \sum_{i=t-2+1}^{t-1} CX_i \cdot x_i + AX + EX_t, \quad t = w, 2w, 3w, \dots$$
$$y_t = \sum_{i=t-w+1}^{t-1} CY_i \cdot y_i + AY + EY_t, \quad t = w, 2w, 3w, \dots \quad (3.11)$$

where *CX* and *CY* are the regression coefficients, *AX* and *AY* are the constant, and *EX* and *EY* are the errors.

We then use the auto regression distance to calculate the Euclidean distance between the coefficients *CX* and *CY* to represent the distance between *X* and *Y*:

$$Dis_{AR} = Dis_{Euclidean}(CY, CX) \quad (3.12)$$

Fig. 2(f) shows the visualization of variables with auto regression distance of window size 25 days. This distance produces a layout in which the time-evolving behaviors of the five variables are quite distinctive. They do not have similar evolving patterns. We could find SO2, NO2, NO and PM10 forming a group while O3 is rather far away from this group.

D. TIS Comparison in Line Chart

The line chart is another common display to plot the values of sequences. As in Fig. 4, NO2 and PM10 in the first 100 days – a large period generates a long curve and so a smaller period is picked for demonstration purposes – are displayed. In the final interface, we allow users to restrict the window size (for now still 25), slide in this period and track the behavior between two variables.

To visualize the similarity of these two variables for the first 100 days, the time interval windows are compared via both average values and original window correlations. As shown in Fig. 5, we observe the correlation is more close to the actual relation between NO2 and PM10 that they are close to each other first and then deviate far away when it is around 80 days (the red peak in Fig. 4).

E. Illustrative Transform Lines

As we mentioned before, the similarity analysis can only show the static pairwise relations. The line charts could only show the value changes. To overcome these limitations, we devise a new approach called illustrative transform lines to combine those two together.

For each time interval window, we first generate a 2D MDS map representing the relations of the variables inside that interval window. These 2D maps are treated as 2D planes stacked horizontally in 3D space. This stack is arranged by time and the data items plotted as colored points P_{it} in plane S_i are connected to the corresponding points in the adjacent time planes S_{i-1} and S_{i+1} by straight lines. In other words, we connect the P_{it} (t=1,2,…,n) to form a line. This gives rise to a 3D display (see sketch in Fig. 5) where the changing relations across time planes can be visualized as changes in the pairwise line configurations.

However, since conventional MDS randomizes the initial coordinates of the points and only preserves the relative (but

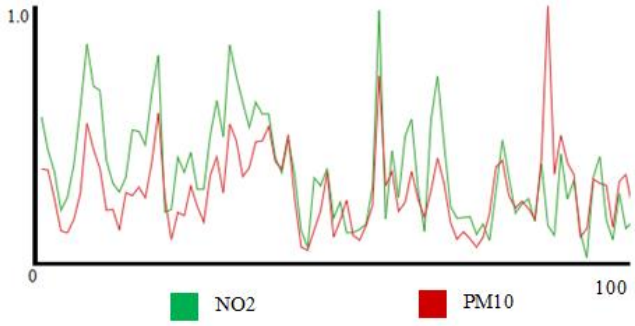

Fig. 4. Value-based comparison of NO2 and PM10 in the first 100 days.

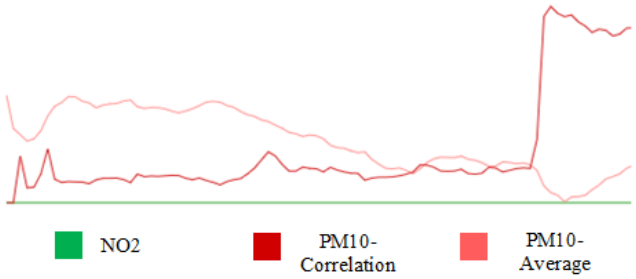

Fig. 5. Behavior comparison of NO2 and PM10 with average distance and correlation distance.

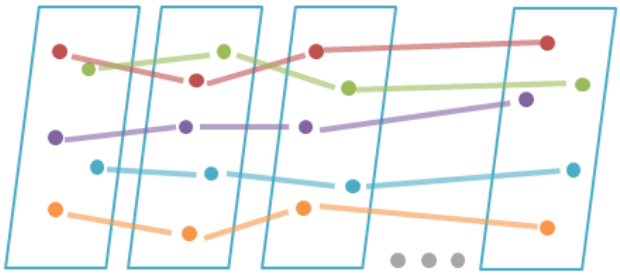

Fig. 6. The sketch of illustrative transform lines formed by connecting the corresponding points in different 2D planes stacked horizontally in 3D space.

not absolute) locations of the points in the final layout that can vary significantly across the layout planes. Hence, the paths created between adjacent planes could be incoherent. We fix this by setting the initial coordinates of the points in a plane to the layout coordinates of the previous plane. In this way, the inter-plane paths can show the relation changes quite well. A remaining problem is that the paths are in 3D which suffers from occlusion problems. So our final step is to map these paths into 2D via another MDS step. In this way the user can recognize any changes easily. As shown in Fig. 7, it is easy to trace the similarity changes among variables even when new data keeps streaming in.

From Fig.7 (b) (same to Fig. 1f in *Relation Display*), we could observe PM10 and NO2 have close relation until Mar, then they divorce sharply, which is consistent with Fig.5. We could also find NO and NO2 maintain good relations during the year, this is because they are both nitrogen oxides.

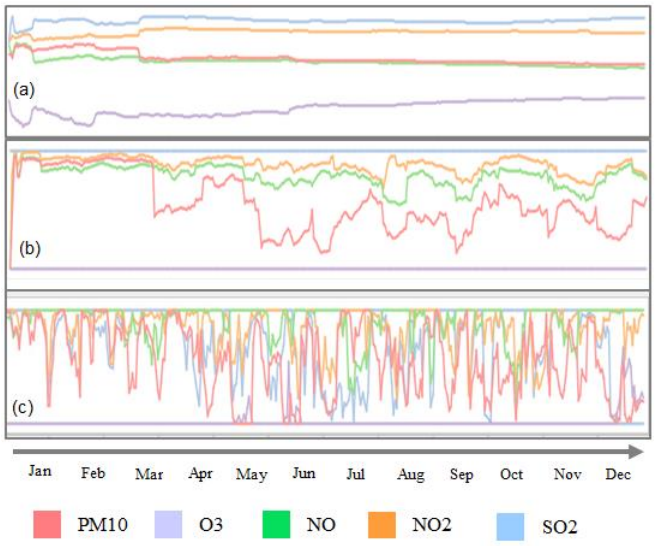

Fig. 7. Comparison between different window size (a) global, (b) estimated and (c) local (window size=5).

F. Window Settings

In the last MDS projection step, we first compute the similarities of the different variables and then project these similarities. However, to gauge the similarities, taking TPS for example, just considering the attributes vector with the estimated window size is not enough. The streaming data is expected to keep changing based on different window size – just an selected one is limited. The different window sizes considering previous time slices will give different attention or weight to the previous and current time slices. We therefore provide a global and a local window for choosing different window sizes.

Global Window

The global window emphasizes the previous slices more and generates the changes over all time periods. This could present the overall changes during the whole period instead of local changes. We achieve this by appending the current time slice to the previous time slices from start time. Based on this window vector, we can lay out these attributes based on similarity. As we defined before, the values of V_i in the slice S_t is x_{it}, thus the window vector W_{it} is:

$$W_{it} = [x_{i1}, x_{i2}, \dots, x_{it}] \tag{3.13}$$

We use this method to our real data as in Fig. 6(a).

Local Window

The global window focuses on the global change over the whole time period. However, it is significant to observe the local change as well. Instead of creating the new attributes vector starting from the very beginning, we create a certain window size to control how many previous time slices to involve. Suppose the window size is l *(l<estimated window size)*, then the new window vector W_{it} is,

$$W_{it} = [x_{i(t-l+1)}, x_{i(t-l+2)}, \dots, x_{it}] \quad (3.14)$$

We can then run MDS based on the distance of the new attribute vectors. The local change (window size=5) of our data is shown in Fig. 7(c).

With the global window size, estimated window size and local window size, analysts can now explore the time-series and track the behavior hierarchically. With the global window size, analysts can gauge the stable relation over local mode. From the local window size, analysts could discover the detail change under the small window. The estimated window size is a good balance among them.

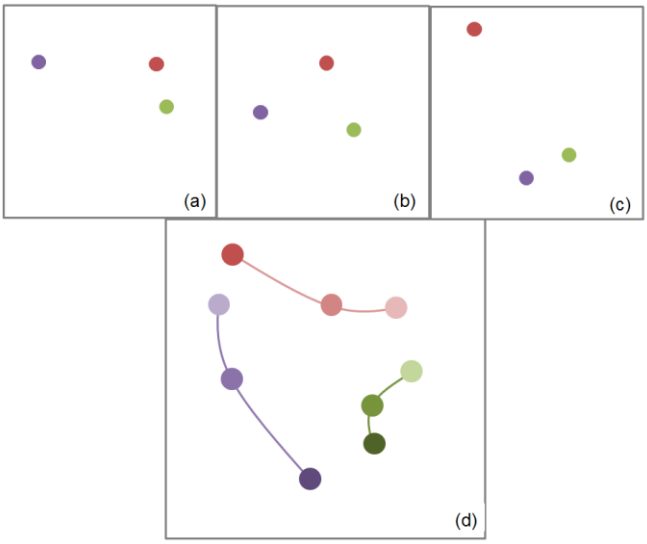

Fig. 8. Dynamic display showing the change using sliding MDS on (a) 2010/09/12 and (b) 2010/10/11 with window size 5.

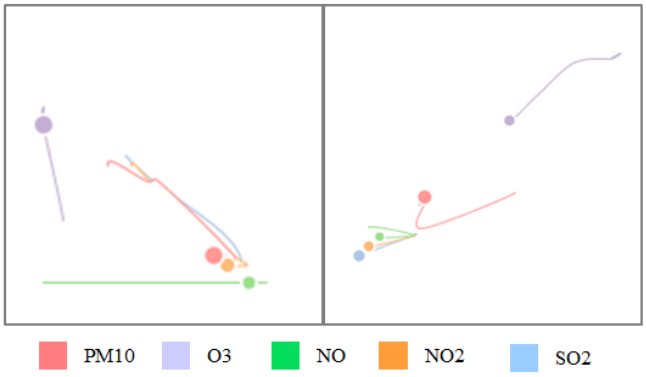

Fig. 9. Dynamic display showing the change using sliding MDS on (a) 2010/09/12 and (b) 2010/10/11 with window size 5.

G. *Local Change Exploration*

The "Relation Display" just presented focuses on the overall relation changes. However, for the streaming analysis the local changes can also be of interest. For this we need to focus on the local changes of different pollutants in a certain time slice. We have developed a special display to show these changes.

Weight Function

The local changes are typically related to adjacent days (but this can be generalized). The exploration size selector (Fig. 1) allows users to set the period they would like to monitor. Additionally, a weight function allows users to set a preference for the days inside the window. We provide three types of weight functions: equal weights, previous focus, and later focus. "Equal weights" gives equal weights of the days in the windows. "Previous focus" gives higher weight for the previous days, and "later focus" emphasises the later days. Fig.1 shows the "later focus" weight function with a window size of 5. In this way, we balance the values of history and current ones.

Sliding MDS

The local (transient) changes can also be visualized via MDS, now by ways of a dynamic layout where the local changes of the points are visualized with streak lines. In order to see more details of local change during the time covered by a exploration window size, we also layout the change via MDS and draw a path to show the change from the past time stamps to the present. This helps to decrease the distortion of 1D MDS and improve the fidelity.

The illustration is shown in Fig.8 (a), (b) and (c) as three time slices that are merged together to generate (d) as the new *sliding MDS*. The brightness indicates the temporal orders as before. In Fig. 9, we compared the results for two specific days – 2010/09/12 and 2010/10/11.

On 2010/09/12, PM10, NO, NO2 and SO2 suddenly have very close relation, while O3 is opposite and moves away from this group (first it moves towards this group a little, then moves completely far way). However, on 2010/10/11, we observe a different pattern – the five pollutants all move close to each other and their relations grow closer. This feature is difficult to see in the relation display. We summarize two reasons – a large window size 'eats' this small feature, and the 2D MDS plot is more accurate than 1D MDS.

Compared with relation display on 2010/09/12 in Fig.1, we could find this 2D display shows more details of the local change. Furthermore, compared with the original values in the stream graph, we could find during the short period close to 2010/09/12, all the pollutants' values increase but just O3 decreases. This observation confirms that our 2D layout is more accurate and helpful to reduce the error in the relation display.

V. Case Study

In this section, we apply our interface to a Futures trading market to evaluate its performance and co-movements. We obtained a data set online with copper, crude oil, platinum,

natural gas and gold prices of 2015. One price per weekday was sampled yielding a total of 245 samples for each variable (attribute) as shown in Fig. 10. From Fig. 10, we could observe the overall value trend of all variables. However, to identify the behavior into a sub time sequence is difficult.

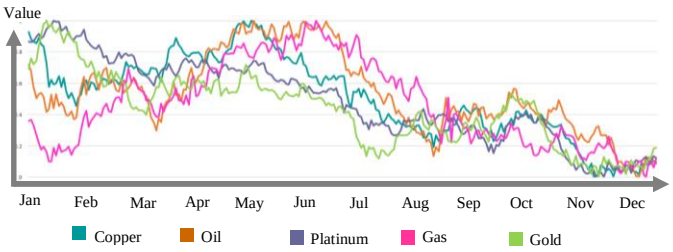

Fig. 10. The line chart of copper, crude oil, platinum, gas and gold price in 2015. The data is normalized for each attribute.

We observe that the behavior relations shown in Fig. 12 among the different variables vary a great deal during 2015. Gold and platinum have relatively stable time relationships. They track each other which is called co-movement in economics. This makes sense since they are both precious metals. Fig. 11(a) shows the actual value curves of gold and platinum, where we can confirm this. Another interesting relationship is that formed by oil and gas (see again Fig. 12). They closely track each other until March, then disconnect, rejoin in April, disconnect again in May, briefly reconnect in August, and then completely disconnect. We can confirm this in the value plot of Fig, 11(b) if we go through the tedious effort and compare the value trends (not absolute values) in the corresponding 2-month time intervals. This again confirms the high utility of our new plots.

VI. Conclusion

We have presented a visual analytics tool that can visualize changing inter-attribute relations within time varying multivariate data. First, with users specifying the desired time slice granularity, the similarities of both the multivariate time samples and the variables can be visualized with different distance metrics in a 2D MDS layout. Second, we propose the notion of illustrative transform lines that can show changes across attributes and adjacent time slices using MDS projection respectively. Third, we offer the period detection to obtain the estimated window size and then build the weight function to balance the emphasis between previous and later time slices. This essentially aids users to detect and explore the local relation changes in more details. Finally, we embed all the displays mentioned above and develop a tool called (StreamVis)ND that can visualize the relations and behaviors in the multivariate streaming data by combining and linking different visualization schemas augmented with interactions. Future work will focus on user studies to refine the framework.

Acknowledgment

This research was supported by NSF grant IIS 1117132, and by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the "ICT Consilience Creative Program" supervised by the IITP (Institute for Information & Communications Tech. Promotion)", and DOE LDRD grant 16-041 from Brookhaven National Laboratory.

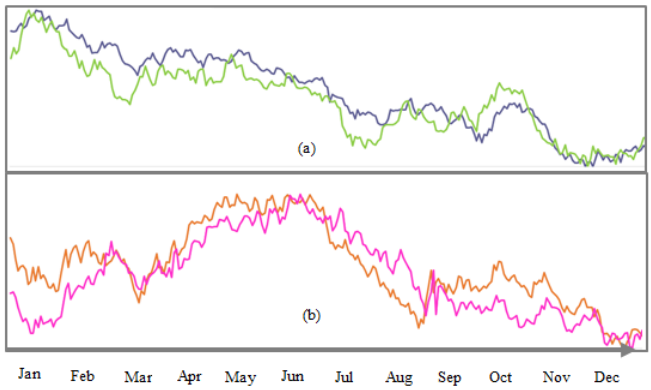

Fig. 11. The values change of Gold and platinum (a) and crude oil and gas (b) in 2015.

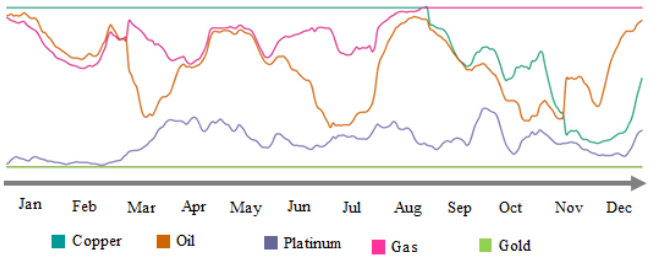

Fig. 12. Dynamic relation between copper, crude oil, platinum, gas and gold price in 2015.

References

[1] D. J. Berndt, J. Clifford, "Using Dynamic Time Warping to Find Patterns in Time Series", *KDD Workshop* 1994: 359-370

[2] L. Byron, M. Wattenberg, "Stacked Graphs - Geometry & Aesthetics", IEEE Trans. Vis. Comput. Graph. 14(6): 1245-1252, 2008.

[3] S. Cheng, K. Mueller, "The Data Context Map: Fusing Data and Attributes into a Unified Display", *IEEE Trans. Vis. Comput. Graph.* 22(1): 121-130, 2016

[4] S. Cheng, K. Mueller, "Improving the Fidelity of Contextual Data Layouts Using a Generalized Barycentric Coordinates Framework," *Proc. Pacific Vis*, pp. 295-302, 2015.

[5] S. Cheng, Y. Wang, D. Zhang, Z. Jiang and K. Mueller, "StreamVisND: Visualizing Relationships in Streaming Multivariate *Data", Proc. IEEE Visualization Conference*, Chicago (USA), October, 2015.

[6] T. Dwyer and D. R. Gallagher, "Visualising changes in fund manager holdings in two and a half-dimensions. *Information Visualization*, 3(4):227–244, 2004.

[7] J. Hartigan, "Printer graphics for clustering," *Journal of Statistical Computation and Simulation*,4(3):187-213, 1975.

[8] A. Hatemi-J, "Multivariate tests for autocorrelation in the stable and unstable VAR models, *Economic Modelling* 21 (4): 661–683

[9] S. Havre, E. Hetzler, L. Nowell: "ThemeRiver: Visualizing Theme Changes over Time," *Proc. IEEE InfoVis*, pp. 115-123, 2000.

[10] P. Hoffman, G. Grinstein, K. Marx, I. Grosse, E. Stanley, "DNA Visual and Analytic Data Mining", *Proc. IEEE Visualization*, pp. 437-441, 1997.

[11] P. Hoffman, G. Grinstein, D. Pinkney, "Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations,"

Proc. Workshop on New Paradigms in Information Visualization and Manipulation, pp. 9-16, 1999.

[12] J. H. Lee, K. T. McDonnell, A. Zelenyuk, D. Imre, K. Mueller, "A Structure-Based Distance Metric for High-Dimensional Space Exploration with Multidimensional Scaling", *IEEE Trans. Vis. Comput. Graph*. 20(3): 351-364 (2014).

[13] A. Inselberg, B. Dimsdale, "Parallel Coordinates: A Tool for Visualizing Multi-Dimensional Geometry," *Proc. IEEE Visualization*, pp. 361-378, 1990.

[14] D. Jäckle, F. Fischer, T. Schreck, D. Keim, "Temporal MDS Plots for Analysis of Multivariate Data". *IEEE Trans. Vis. Comput. Graph*. 22(1): 141-150, 2016

[15] I. Jolliffe, "Principal Component Analysis," *Series: Springer Series in Statistics*, 2nd ed., Springer, NY, 2002, XXIX, 487 pp.28 illus. ISBN 978-0-387-95442-4.

[16] E. Kandogan, "Star Coordinates: A Multi-Dimensional Visualization Technique with Uniform Treatment of Dimensions," *Proc. IEEE Information Visualization, Late Breaking Topics*, pp. 9-12, 2000.

[17] J. Kruskal. M. Wish, *Multidimensional Scaling*. Sage Publications, 1977.

[18] L. Maaten, G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, 9:2579–2605, 2008

[19] M. Meyer, A. Barr, H. Lee, M. Desbrun, "Generalized Barycentric Coordinates on Irregular Polygons," *J. Graphics Tools*, 7(1):13-22, 2002.

[20] M. Steiger, J. Bernard, S. Mittelstaedt, H. Tieke, D. Keim, T. May, J. Kohlhammer, "Visual Analysis of Time-Series Similarities for Anomaly Detection in Sensor Networks," *Comput. Graph. Forum* 33(3): 401-410, 2014.

[21] J. Lee, K. McDonnell, A. Zelenyuk, D. Imre, K. Mueller, "A Structure-Based Distance Metric for High-Dimensional Space Exploration with Multi-Dimensional Scaling," *IEEE Trans. on Visualization and Computer Graphics*, 20(3): 351-364, 2014.

[22] L. Saul , S. Roweis, "An Introduction to Locally Linear Embedding" *IJPRAI* 01/2009; 23:1739-1752. DOI: 10.1142/S0218001409007752.

[23] J. Tenenbaum, V. de Silva, J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science 290*, 2319–2323, 2000.