

Interactive Visualization of Multivariate Time Series Data

Shawn Martin^(✉) and Tu-Toan Quach

Sandia National Laboratories, Albuquerque, NM 87185, USA
{smartin,ttquach}@sandia.gov

Abstract. Organizing multivariate time series data for presentation to an analyst is a challenging task. Typically, a dataset contains hundreds or thousands of datapoints, and each datapoint consists of dozens of time series measurements. Analysts are interested in how the datapoints are related, which measurements drive trends and/or produce clusters, and how the clusters are related to available metadata. In addition, interest in particular time series measurements will change depending on what the analyst is trying to understand about the dataset.

Rather than providing a monolithic single use machine learning solution, we have developed a system that encourages analyst interaction. This system, *Dial-A-Cluster* (DAC), uses multidimensional scaling to provide a visualization of the datapoints depending on distance measures provided for each time series. The analyst can interactively adjust (dial) the relative influence of each time series to change the visualization (and resulting clusters). Additional computations are provided which optimize the visualization according to metadata of interest and rank time series measurements according to their influence on analyst selected clusters.

The DAC system is a plug-in for Slycat (slycat.readthedocs.org), a framework which provides a web server, database, and Python infrastructure. The DAC web application allows an analyst to keep track of multiple datasets and interact with each as described above. It requires no installation, runs on any platform, and enables analyst collaboration. We anticipate an open source release in the near future.

Keywords: Multivariate time series · Multidimensional scaling · Interactive visualization · Slycat

1 Introduction

There are numerous problems from different fields that produce time series data, including chemical engineering [27], intrusion detection [31], economic forecasting [28], gene expression analysis [21], hydrology [23], social network analysis [32], and fault detection [11]. Fortunately, there are just as many algorithms available for analyzing time series data [9]. These algorithms involve tasks including queries [9, 10], anomaly detection [2], clustering [4, 20], classification [3, 9], motif discovery [9, 24], and segmentation [16]. From a practical point of view, these

algorithms share basic data processing goals starting with pre-processing and normalization [9], representation [17], and similarity computation [9, 15].

In addition to the large body of algorithms available for mining time series data, there is an additional set of techniques available for visualization of time series [1, 22, 26]. These techniques belong to the field of Visual Analytics, or sometimes Interactive Visual Analytics [14, 30], and include methods such as Parallel Coordinates [12], multiple views, brushing, selection, and iteration. Researchers in Visual Analytics have called out the need for greater integration with underlying algorithms [5].

In between these two fields of research there is a smaller body of work which investigates the interactive visualization of multivariate time series data [18, 25, 29]. Most of this work focuses on visualization and interaction with multivariate time series plots. Our work fits within this area, but with an emphasis on the algorithms used in the visualization. We provide a layer of abstraction by providing an interactive visual summary of the data, rather than just looking at the time series themselves.

In this paper, we describe a lightweight system for analyzing multivariate time series data called *Dial-A-Cluster* (DAC). DAC is designed to provide a straightforward set of algorithms focused on allowing an analyst to visualize and interactively explore a multivariate time series dataset. DAC requires pre-computed distance matrices so it can exploit a large number of available algorithms related to time series representation and similarity comparison [9]. The DAC interface uses a multidimensional scaling [6] to provide a visualization of the dataset. The analyst can adjust the visualization by interactively weighting the distance measures for each time series. A modification of Fisher's discriminant [8] can be used to rank the importance of each time series. Finally, an optimized weighting scheme for the visualization can be used to maximally correlate the data with analyst specified metadata.

DAC is implemented as a plugin for Slycat (slycat.readthedocs.org) [7], a system which provides a web server, a database, a Python infrastructure for remote computation (on the web server). The Slycat DAC plugin is a web application which provides the previously described time series analysis algorithms. It requires no installation and is platform independent. In addition, DAC supports (via Slycat) management of multiple users, multiple datasets, and access control, therefore encouraging collaboration while maintaining data privacy. Slycat and DAC are implemented using JavaScript and Python. Slycat is open source (github.com/sandialabs/slycat).

2 Algorithms

The primary goal of DAC is to provide a no-install, interactive user interface which can be used to organize and query multivariate time series data according to the interests of the analyst. There are three algorithms which support this goal: visualization using multidimensional scaling, identifying time series most responsible for differences in analyst selected clusters, and optimizing the visualization according to analyst specified metadata.

2.1 Multidimensional Scaling

DAC uses classical multidimensional scaling (MDS) to compute coordinates for a dataset, where each datapoint is a set of time series measurements. To be precise, suppose we have a dataset $\{x_i\}$, where x_i is a datapoint, for example an experiment or a test. Each datapoint consists of a number of time series measurements, which we write as a vector $x_i = [\mathbf{t}_{ik}]$, where \mathbf{t}_{ik} is the k th time series vector for datapoint x_i . Note that we are abusing notation here, because each vector \mathbf{t}_{ik} may be a different length, but that we require that the \mathbf{t}_{ik} have the same length for the same k . We also assume that we are given distance matrices

$$D_k = \begin{bmatrix} d_k(x_1, x_1) & d_k(x_1, x_2) & \cdots \\ d_k(x_2, x_1) & d_k(x_2, x_2) & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

for each time series measurement, where $d_k(x_i, x_j)$ gives a distance between datapoint x_i and x_j for time series k . For example using Euclidean distance we would have

$$d_k(x_i, x_j) = d(\mathbf{t}_{ik}, \mathbf{t}_{jk}) = \sqrt{\sum_l (t_{ikl} - t_{jkl})^2},$$

where $\mathbf{t}_{ik} = [t_{ikl}]$ is the k th time series vector \mathbf{t}_{ik} for datapoint x_i indexed by l . Other distances can be used, so that each time series distance metric can be tailored to the type of measurement taken.

Now let $\boldsymbol{\alpha} = [\alpha_k]$ be a vector of scalars, with $\alpha_k \in [0, 1]$. The vector $\boldsymbol{\alpha}$ contains weights so that we may compute weighted versions of our datapoints, defined as $\boldsymbol{\Phi}(x_i) = [\alpha_k \mathbf{t}_{ik}]$, where we are again abusing notation since the vectors \mathbf{t}_{ik} are allowed to have different lengths. Now we define a distance matrix D of pairwise weighted distances between every datapoint, where the entries of D are given by

$$d^2(x_i, x_j) = \boldsymbol{\Phi}(x_i)^T \boldsymbol{\Phi}(x_j) = \sum_k \alpha_k^2 d_k^2(x_i, x_j).$$

The matrix D is the matrix of pairwise distances between datapoints used as input to MDS within DAC. The weights α_k are adjustable by the analyst. Note that using these definitions

$$D = \sqrt{\sum_k \alpha_k^2 D_k^2}.$$

For completeness, we describe the MDS algorithm operating on the matrix D . First, we double center the distance matrix, obtaining

$$B = -\frac{1}{2}HD^2H,$$

where D^2 is the componentwise square of D , and $H = I - II^T/n$, n being the size of D . Next, we perform an eigenvalue decomposition of B , keeping only

the two largest positive eigenvalues λ_1, λ_2 and corresponding eigenvectors $\mathbf{e}_1, \mathbf{e}_2$. The MDS coordinates are given by the columns of $E\Lambda^{1/2}$, where E is the matrix containing the two eigenvectors $\mathbf{e}_1, \mathbf{e}_2$ and Λ is the diagonal matrix containing the two eigenvalues λ_1, λ_2 .

Finally, we note that the eigenvectors computed by MDS are unique only up to sign. This fact can manifest itself as disconcerting coordinate flips in the DAC interface given even small changes in α by the analyst. To minimize these flips, we use the Kabsch algorithm [13] to compute an optimal rotation so that the newly computed coordinates are as closely aligned to the existing coordinates as possible. The Kabsch algorithm uses the Singular Value Decomposition (SVD) to compute the optimal rotation matrix. If we assume that matrices P and Q have columns containing the previous and new MDS coordinates, then we form $A = P^T Q$ and use the SVD to obtain $A = U\Sigma V^T$. If we denote $r = \text{sign}(\det(VU^T))$ then the rotation matrix is given by

$$R = V \begin{bmatrix} 1 & 0 \\ 0 & r \end{bmatrix} U^T.$$

2.2 Time Series Differences

In addition to using MDS to visualize the relationships between the datapoints, DAC allows the user to select subsets of the dataset and upon request ranks the time series according to how well each time series separates those subsets. DAC allows two different selections and ranks the time series according to Fisher’s Discriminant [8].

To be precise, for each distance matrix D_k , we compute the values of Fisher’s Discriminant $J_k(u, v)$, where $u, v \subset \{x_i\}$ are two groups that we wish to contrast. By definition,

$$J_k(u, v) = \frac{\|\bar{u} - \bar{v}\|^2}{S_u^2 + S_v^2},$$

where $S_u^2 = \sum_i \|u_i - \bar{u}\|^2$, $S_v^2 = \sum_j \|v_j - \bar{v}\|^2$, and \bar{u}, \bar{v} are averages over the sets $\{u_1, \dots, u_n\}, \{v_1, \dots, v_m\}$. Although we do not provide the algebraic derivation, we claim that

$$\|\bar{u} - \bar{v}\|^2 = \frac{1}{n} \frac{1}{m} \sum_{ij} d^2(u_i, v_j) - \frac{1}{2n^2} \sum_{ik} d^2(u_i, u_k) - \frac{1}{2m^2} \sum_{jk} d^2(v_j, v_k),$$

where k varies over i for \sum_{ik} and k varies over j for \sum_{jk} . We similarly claim that $S_u^2 = \frac{1}{2n} \sum_{ik} d^2(u_i, u_k)$ and $S_v^2 = \frac{1}{2m} \sum_{jk} d^2(v_j, v_k)$. Now we can compute $J_k(u, v)$ using only submatrices of the distance matrices D_k .

DAC ranks the time series in descending order of the values of $J_k(u, v)$. Since a higher value of Fisher’s Discriminant $J_k(u, v)$ indicates a greater separation between the selections, this ranking reveals the time series which exhibit the greatest differences between the subsets.

2.3 Clustering by Metadata

Often an analyst will be interested in metadata describing the datapoints. Questions might include: does the dataset cluster relative to a particular metadata variable?; can we make the dataset cluster relative to the metadata variable by adjusting the α weights of the time series?; and which time series are most affected by a metadata variable? To address these questions, we incorporate a supervised optimization of the visualization which correlates the distances between time series with the distances between metadata values.

Specifically, we compute α such that the distances in $D^2 = \sum_k \alpha_k^2 D_k^2$ are as close as possible to the distances in the matrix D_p^2 , where D_p is a pairwise distance matrix for a given metadata property p . In other words, we want to solve

$$\begin{aligned} \min_{\alpha} \quad & \sum_{ij} (\sum_k \alpha_k^2 d_k^2(x_i, x_j) - d_p^2(x_i, x_j))^2 \\ \text{s.t.} \quad & \alpha_k \in [0, 1], \end{aligned}$$

where $d_p(x_i, x_j)$ is the property distance between x_i and x_j , i.e. $d_p(x_i, x_j) = |p_i - p_j|$, where p_i is the metadata property of x_i and p_j is the metadata property of x_j . Note that for MDS, we can scale α by a positive scalar with no effect, so that the constraint $\alpha \in [0, 1]$ is unnecessary. If we let $\beta_k = \alpha_k^2$ we have

$$\begin{aligned} \min_{\beta} \quad & \sum_{ij} (\sum_k \beta_k d_k^2(x_i, x_j) - d_p^2(x_i, x_j))^2 \\ \text{s.t.} \quad & \beta_k \geq 0. \end{aligned}$$

In the Frobenius matrix norm, we have

$$\begin{aligned} \min_{\beta} \quad & \|\sum_k \beta_k D_k^2 - D_p^2\|_F^2 \\ \text{s.t.} \quad & \beta_k \geq 0. \end{aligned}$$

Now if we let $U = [D_1^2, D_2^2, \dots]$, where each D_k^2 is written as a column vector, and $V = [D_p^2]$, where D_p^2 is written as a column vector, then we have

$$\begin{aligned} \min_{\beta} \quad & \|U\beta - V\|^2 \\ \text{s.t.} \quad & \beta \geq 0. \end{aligned}$$

This is known as a non-negative least squares problem [19]. Once we compute β we can obtain time series weights α corresponding to an MDS visualization optimized to a particular metadata property value.

3 User Interface

The DAC user interface allows access to the algorithms discussed in Sect. 2. DAC assumes that the time series data has been pre-processed, metadata has been collected, and distance matrices have been computed. These assumptions allow flexibility in terms of representing the time series data and computing similarities, two steps in time series analysis served by a wide variety of different algorithms [9]. In addition, pre-computing the distance matrices ensures

that DAC will operate in real-time for reasonable dataset sizes (up to ~ 5000 datapoints).

The DAC user interface consists of sliders to adjust α_k values (labelled using variable names meaningful to analysts), a canvas to display the MDS visualization, traditional time series plots, and a table displaying metadata. The interface is shown in Fig. 1.

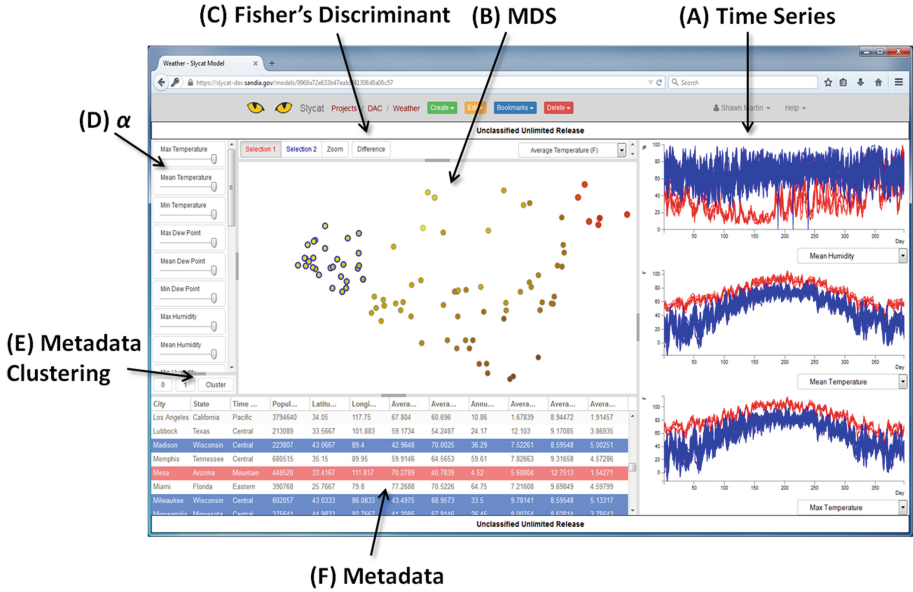


Fig. 1. DAC user interface. Here we show DAC running in Firefox on a Windows PC. Reading the labels counter-clockwise from the upper right: (A) time series data is displayed in the traditional manner; (B) MDS is used to provide a visual representation of the datapoints in the dataset, shown as circles; (C) Fisher's Discriminant can be used to order the time series to maximize the difference between analyst selected red and blue groups; (D) time series measurements can be weighted to adjust the visualization according to analyst preference; (E) the weights can be computed optimally to correlate with an analyst chosen metadata field; and (F) metadata can be examined. (Color figure online)

The DAC interface is a Slycat plugin (slycat.readthedocs.org) [7]. Slycat supports the management of multiple users, multiple datasets, and access controls. Both Slycat and DAC are implemented using JavaScript and Python. DAC is written in JavaScript using jQuery for the controls. The time series and MDS plots are rendered and animated using D3, and the metadata table uses SlickGrid. Calculations are performed on the Slycat webserver using Python and NumPy. Slycat is open source (github.com/sandialabs/slycat) and DAC will be released as open source in the near future.

4 Example

To demonstrate how DAC might be used by an analyst, we provide an example using publicly available weather data. The data consists of weather time



Fig. 2. DAC weather data. On the left (A), we show the DAC MDS visualization of the cities in the weather dataset. In the middle (B), the visualization is colored by average temperature, where yellow is low and brown is high. On the right (C), the visualization is colored by annual precipitation, where yellow is again low and brown is high. (Color figure online)

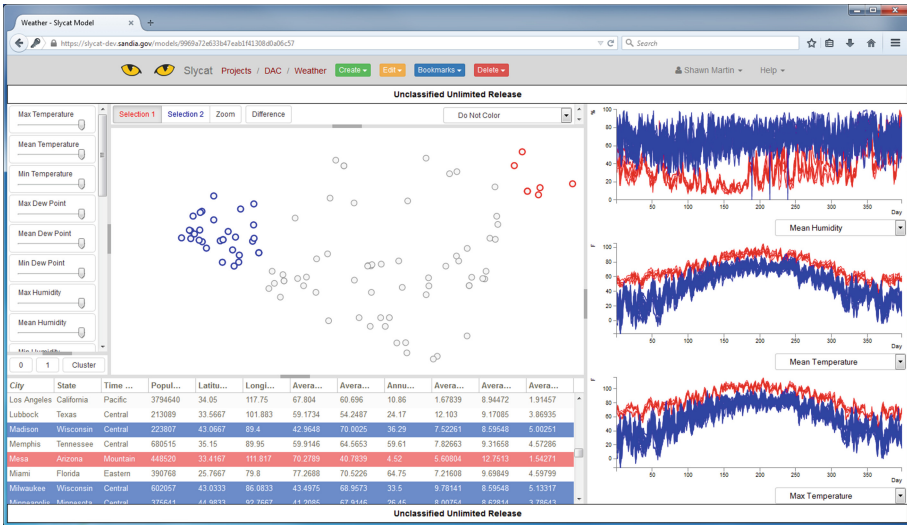


Fig. 3. DAC analyst selections. Here we show two selections made by the analyst, cities on the left hand side of the visualization are selected in blue, and cities in the upper right are selected in red. The same coloring scheme is automatically reflected in the metadata table and colored time series plots are shown on the right. The blue cities include Madison, WI and Milwaukee, WI, and the red cities include Mesa, AZ. By pushing the difference button, DAC ranks and orders the time series plots in the right hand panel of the interface. In this case, humidity gives the greatest difference between the red and blue cities, followed by temperature. (Color figure online)



Fig. 4. DAC optimal MDS. Here we show the optimal MDS coordinates correlated with latitude, computed according to the algorithm in Sect. 2.3. The α values are automatically adjusted to show that temperature, dew point, and sea level pressure are best suited to represent latitude, and the previous city selections show that the cold wet cities tend to be in the north and the hot dry cities tend to be in the south. (Color figure online)

series data from Weather Underground (www.wunderground.com) during the year 2014 for the 100 most populated cities in the the United States. The time series measurements include temperature, dew point, humidity, sea level pressure, visibility, wind speed, precipitation, cloud cover, and wind direction. Meta-data for the cities includes city name, state, time zone, population, latitude, longitude, average temperature, average humidity, annual precipitation, average windspeed, average visibility and average cloud cover.

Upon starting, DAC produces an MDS visualization of the dataset assuming $\alpha = 1$. For the weather data, this visualization is shown in Fig. 2(A). Among the simplest functions provided by DAC is the ability to color the datapoints according to analyst selected metadata. A coloring of the weather data by average temperature is shown in Fig. 2(B) and by annual precipitation in Fig. 2(C).

From the coloring, it appears that cities on the left hand side of the visualization are cold and wet, while cities on the upper right are hot and dry. This can be confirmed by selecting cities in these areas of the visualization and examining their metadata and time series, as shown in Fig. 3. The selections show that cities on the left (blue selections) are indeed cold and wet and are located in the northern and eastern parts of the country, while the cities in the upper right (red selections) are hot and dry and are located in Arizona and Nevada. By pushing the difference button, Fisher’s Discriminant is computed against the red and blue selections to rank the time series plots in the right panel of

the DAC interface, showing that humidity and temperature give the greatest differences between the two selections.

Finally, the analyst might speculate that the latitude has a significant correlation with the MDS coordinate visualization. Coloring by latitude and pushing the cluster button produces the visualization shown in Fig. 4. This visualization is computed according to the optimization in Sect. 2.3 to obtain the MDS coordinates that best correlate with latitude. The analyst's speculation is confirmed in that the red cities are positioned on the upper right and the blue cities are positioned on the lower left. In addition, the α values computed show that temperature, dew point, and sea level pressure are the most significant weights in the optimized MDS coordinates. Unsurprisingly, temperature is the main influence.

5 Conclusion

Interactive visualization of multivariate time series data is a challenging problem. In addition to organizing what can be large quantities of data for display, there are many potential algorithms available for analyzing the data. We have designed a lightweight web application to bridge the gap between these two problems.

Our system, *Dial-A-Cluster* (DAC), allows an expert data mining practitioner to pick and choose among the available algorithms for time series representation and similarity comparison to pre-compute distance matrices for use with DAC. (Alternatively, a novice practitioner can use very simple pre-processing and Euclidean distance to compute the matrices for DAC.)

DAC in turn provides a subject matter expert a lightweight, no-installation, platform independent interface for examining the data. DAC implements a real-time MDS coordinate based abstraction for the dataset, as well as an interactive interface for examining the actual time series data and metadata. DAC uses Fisher's Discriminant to rank and order the time series according to analyst selections. Finally, DAC provides an optimized computation for determining which time series measurements are correlated with metadata of interest to the analyst.

Instead of making the analyst an evaluator of the data mining results, DAC provides an easy to use interface which encourages the analyst to explore the data independently. Further, since DAC is implemented as a Slycat plugin, management of multiple datasets, multiple users, and access controls are also provided, encouraging collaboration between multiple analysts while maintaining data privacy.

Acknowledgements. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. This work was funded by Sandia Laboratory Directed Research and Development (LDRD).

References

1. Aigner, W., Miksch, S., Schumann, H., Tominski, C.: Visualization of Time-Oriented Data. Springer, London (2011)
2. Aggarwal, C.C.: Outlier Analysis. Springer, New York (2013)
3. Bakshi, B., Stephanopoulos, G.: Representation of process Trends-IV. Induction of real-time patterns from operating data for diagnosis and supervisory control. *Comput. Chem. Eng.* **18**(4), 303–332 (1994)
4. Berkhin, P.: A survey of clustering data mining techniques. In: Kogan, J., Nicholas, C., Teboulle, M. (eds.) *Grouping Multidimensional Data*, pp. 25–71. Springer, Heidelberg (2006)
5. Bertini, E., Lalanne, D.: Surveying the complementary role of automatic data analysis and visualization in knowledge discovery. In: *Proceedings of the ACM SIGKDD Workshop on Visual Analytics and Knowledge Discovery (VAKD 2009)*, pp. 12–20 (2009)
6. Borg, I., Goenen, P.J.F.: *Modern Multidimensional Scaling*. Springer, New York (2005)
7. Crossno, P.J., Shead, T.M., Sielicki, M.A., Hunt, W.L., Martin, S., Hsieh, M.-Y.: Slycat ensemble analysis of electrical circuit simulations. In: Bennett, J., Vivodtzev, F., Pascucci, V. (eds.) *Topological and Statistical Methods for Complex Data. Mathematics and Visualization*, pp. 279–294. Springer, Heidelberg (2015)
8. Duda, R.O., Hart, P.E., Stork, D.G.: *Pattern Classification*. John Wiley and Sons, Hoboken (2000)
9. Esling, P., Agon, C.: Time-Series data mining. *ACM Comp. Surv.* **45**(1), 12 (2012)
10. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. *SIGMOD Rec.* **23**, 419–429 (1994)
11. Fontes, C.H., Pereira, O.: Pattern recognition in multivariate time series - a case study applied to fault detection in a gas turbine. *Eng. Appl. Artif. Intell.* **49**, 10–18 (2016)
12. Inselberg, A.: A survey of parallel coordinates. In: Hege, H.-C., Polthier, K. (eds.) *Mathematical Visualization*, pp. 167–179. Springer, Heidelberg (1998)
13. Kabsch, W.: A solution for the best rotation to relate two sets of vectors. *Acta Crystallogr. Sect. A* **32**(5), 922–923 (1976)
14. Keim, D.A., Kohlhammer, J., Ellis, G., Mansmann, F.: *Mastering the information age - solving problems with visual analytics*. Eurographics (2010)
15. Keogh, E., Kasetty, S.: On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Min. Knowl. Discov.* **7**(4), 349–371 (2003)
16. Keogh, E., Chu, S., Hart, D., Pazzani, M.: Segmenting time series: a survey and novel approach. *Data Min. Time Ser. Databases* **57**, 1–22 (2004)
17. Keogh, E., Lonardi, S., Ratanamahatana, C.: Towards parameter-free data mining. In: *Proceedings of the 10th ACM International Conference on Knowledge Discovery and Data Mining*, pp. 206–215 (2004)
18. Konyha, Z., Lez, A., Matkovic, K., Jelovic, M., Hauser, H.: Interactive visualization analysis of families of curves using data aggregation and derivation. In: *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*, pp. 1–24 (2012)
19. Lawson, C.L., Hanson, R.J.: *Solving Least Squares Problems*, vol. 161. Prentice-Hall, Upper Saddle River (1974)
20. Lin, J., Keogh, E.: Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowl. Inf. Syst.* **8**(2), 154–177 (2005)

21. Lin, T., Kaminski, N., Bar-Joseph, Z.: Alignment and classification of time series gene expression in clinical studies. *Bioinf.* **24**(13), 147–155 (2008)
22. Muller, W., Schumann, H.: Visualization methods for time-dependent data - an overview. In: Proceedings of the Winter Simulation Conference, pp. 737–745 (2003)
23. Ouyang, R., Ren, L., Cheng, W., Zhou, C.: Similarity search and pattern discovery in hydrological time series data mining. *Hydrol. Process.* **24**(9), 1198–1210 (2010)
24. Patel, P., Keogh, E., Lin, J., Lonardi, S.: Mining motifs in massive time series databases. In: Proceedings of the IEEE International Conference on Data Mining (ICDM 2002), pp. 370–377 (2002)
25. Peng, R.: A method for visualizing multivariate time series data. *J. Stat. Soft.* **25**(1), 1–17 (2008)
26. Silva, S.F., Catarci, T.: Visualization of linear time-oriented data: a survey. In: Proceedings of the International Conference on Web Information Systems Engineering, p. 310 (2000)
27. Singhal, A., Seborg, D.E.: Clustering multivariate time-series data. *J. Chemometr.* **19**(8), 427–438 (2005)
28. Song, H., Li, G.: Tourism demand modelling and forecasting - a review of recent research. *Tour. Manag.* **29**(2), 203–220 (2008)
29. Thakur, S., Rhyne, T.-M.: Data vases: 2D and 3D plots for visualizing multiple time series. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Kuno, Y., Wang, J., Pajarola, R., Lindstrom, P., Hinkenjann, A., Encarnaç o, M.L., Silva, C.T., Coming, D. (eds.) ISVC 2009, Part II. LNCS, vol. 5876, pp. 929–938. Springer, Heidelberg (2009)
30. Thomas, J. J., Cook, K. A.: Illuminating the path: the research and development-agenda for visual analytics. *Nat. Vis. & Anal. Ctr.* (2005)
31. Zhong, S., Khoshgoftaar, T., Seliya, N.: Clustering-Based network intrusion detection. *Int. J. Reliab. Qual. Saf. Eng.* **14**(2), 169–187 (2007)
32. Zhu, J., Wang, B., Wu, B.: Social network users clustering based on multivariate time series of emotional behavior. *J. China Univ. Posts Telecom.* **21**(2), 21–31 (2014)