A Framework for Time-Series Analysis*

Vladimir Kurbalija[1], Miloš Radovanović[1], Zoltan Geler[2], and Mirjana Ivanović[1]

[1] Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad, Trg D. Obradovica 4, 21000 Novi Sad, Serbia
[2] Faculty of Philosophy, University of Novi Sad, Dr Zorana Đinđića 2, 21000 Novi Sad, Serbia
{kurba,radacha,zoltan.geler,mira}@dmi.uns.ac.rs

Abstract. The popularity of time-series databases in many applications has created an increasing demand for performing data-mining tasks (classification, clustering, outlier detection, etc.) on time-series data. Currently, however, no single system or library exists that specializes on providing efficient implementations of data-mining techniques for time-series data, supports the necessary concepts of representations, similarity measures and preprocessing tasks, and is at the same time freely available. For these reasons we have designed a multi-purpose, multifunctional, extendable system FAP – Framework for Analysis and Prediction, which supports the aforementioned concepts and techniques for mining time-series data. This paper describes the architecture of FAP and the current version of its Java implementation which focuses on time-series similarity measures and nearest-neighbor classification. The correctness of the implementation is verified through a battery of experiments which involve diverse time-series data sets from the UCR repository.

Keywords: time-series analysis, time-series mining, similarity measures.

1 Introduction

In different scientific fields, a time-series consists of sequence of values or events obtained over repeated measurements of time [2]. Time-series analysis comprises methods that attempt to understand such time series, often either to understand the underlying context of the data points, or to make forecasts.

Time-series databases are popular in many applications, such as stock market analysis, economic and sales forecasting, budgetary analysis, process control, observation of natural phenomena, scientific and engineering experiments, medical treatments etc. As a consequence, in the last decade there has occurred an increasing amount of interest in querying and mining such data which resulted in a large amount of work introducing new methodologies for different task types including: indexing, classification, clustering, prediction, segmentation, anomaly detection, etc. [1, 2, 3, 4].

There are several important concepts which have to be considered when dealing with time series: pre-processing transformation, time-series representation and similarity/distance measure.

* This work was supported by project *Abstract Methods and Applications in Computer Science* (no. 144017A), of the Serbian Ministry of Science and Environmental Protection.

Pre-processing transformation: "Raw" time series usually contain some distortions, which could be consequences of bad measurements or just a property of underlying process which generated time series. The presence of distortions can seriously deteriorate the indexing problem because the distance between two "raw" time series could be very large although their overall shape is very similar. The task of the pre-processing transformations is to remove different kinds of distortions. Some of the most common pre-processing tasks are: offset translation, amplitude scaling, removing linear trend, removing noise etc. [22] Pre-processing transformations can greatly improve the performance of time-series applications by removing different kinds of distortions.

Time-series representation: Time series are generally high-dimensional data [2] and a direct dealing with such data in its raw format is very time and memory consuming. Therefore, it is highly desirable to develop representation techniques that can reduce the dimensionality of time series. Many techniques have been proposed for representing time series with reduced dimensionality: *Discrete Fourier Transformation* (DFT) [1], *Singular Value Decomposition* (SVD) [1], *Discrete Wavelet Transf.* (DWT) [5], *Piecewise Aggregate Approximation* (PAA) [6], *Adaptive Piecewise Constant Approx.* (APCA) [7], *Symbolic Aggregate approX.* (SAX) [8], *Indexable Piecewise Linear Approx.* (IPLA) [9], *Spline Representation* [10], etc.

Similarity/distance measure: Similarity-based retrieval is used in all a fore mentioned task types of time-series analysis. However, the distance between time series needs to be carefully defined in order to reflect the underlying (dis)similarity of these specific data which is usually based on shapes and patterns. There is a number of distance measures for similarity of time-series data: *L_p distance* (L_p) [1], *Dynamic Time Warping* (DTW) [11], distance based on *Longest Common Subsequence* (LCS) [12], *Edit Distance with Real Penalty* (ERP) [13], *Edit Distance on Real sequence* (EDR) [14], *Sequence Weighted Alignment model* (Swale) [31], etc.

All these concepts, when introduced, are usually separately implemented and described in different publications. Every newly-introduced representation method or a distance measure has claimed a particular superiority [4]. However, this was usually based on comparison with only a few counterparts of the proposed concept. On the other hand, by the best of our knowledge there is no freely available system for time-series analysis and mining which supports all mentioned concepts, with the exception of the work proposed in [4].

Being motivated by these observations, we have designed a multipurpose, multifunctional system FAP - Framework for Analysis and Prediction. FAP supports all mentioned concepts: representations, similarity measures and pre-processing tasks; with the possibility to easily change some existing or to add new concrete implementation of any concept. We have implemented all major similarity measures [4], and conducted a set of experiments to validate their correctness.

The rest of the paper is organized as follows. Section 2 gives an overview and comparison of existing systems for time-series analysis. The architecture of FAP system and implemented similarity measures are described in Section 3. Section 4 presents the evaluation methodology based on [4], and gives the results of experiments. Future work is discussed in Section 5. Section 6 concludes the paper.

2 Related Work

As a consequence of importance and usefulness of time series, there is a large number of applications which deal with time series, based on different approaches.

The most popular collection of data mining algorithms is implemented within *WEKA* (Waikato Environment for Knowledge Analysis) tool [15]. *WEKA* supports a great number of data-mining and machine-learning techniques, including data pre-processing, classification, regression and visualization. However, *WEKA* is a general-purpose data-mining library, and it is not specialised for time series. As a result, the time-series support within WEKA is based on external implementations contributed by some users.

A similar system, *RapidMiner*, is a product of company *Rapid-I* [16]. It is also an open source (Community Edition) collection of data-mining and machine-learning techniques. *RapidMiner* has a very sophisticated graphical user interface, and it is also extensible with the user's implementations. It supports some aspects of statistical time-series analysis, prediction and visualisation.

Also, there are several tools specialised for summarisation and visualisation of time series: *TimeSearcher* [23], *Calendar-Based Visualisation* [24], *Spiral* [25] and *Viz-Tree* [26], but they are not specialised for real-world time-series analysis.

The above systems, which partially support time-series analysis, are mainly based on data mining methods. On the other hand, there is a large number of systems which are based on statistical and econometric modelling. Probably the most famous business system is *SAS* [17]. Among many business solutions including: Business Analytics, Business Intelligence, Data Integration, Fraud Prevention & Detection, Risk Management etc., SAS has an integrated subsystem for time series. This subsystem provides modelling of trend, cycles and seasonality of time series as well as time series forecasting, financial and econometric analysis. However, *SAS* is a complex commercial system which is not freely available.

GRETL (GNU Regression, Econometrics and Time-series Library) is open source, platform-independent library for econometric analysis [18]. It supports several least-square based statistical estimators, time-series models and several maximum-likelihood methods. *GRETL* also encloses a graphical user interface for the *X-12-ARIMA* environment. *X-12-ARIMA* is the Census Bureau's new seasonal adjustment program [27]. It supports several interesting concepts such as: alternative seasonal, trading-day and holiday effect adjustment; an alternative seasonal-trend-irregular decomposition; extensive time-series modeling and model-selection capabilities for linear regression models with ARIMA errors.

In addition, some of the most common time-series analysis systems based on statistical modelling are: *TRAMO* [28], *SEATS* [28], *SSA-MTM* [29], *JMulTi* [30], etc.

Clearly two kinds of applications can be distinguished: general purpose data-mining applications which in some extent support time series, and applications specialised for time series based on statistical and econometric models. So, it is evident that there is a huge gap between these two types of applications. There is no available system specialised for time-series mining (time-series analysis based on data-mining techniques) with the exception of the library presented in [4] which can be obtained on demand. This was our main motive for designing a FAP system. FAP will contain all main features and functionalities needed for time-series analysis (pre-processing tasks, similarity measures, time-series representations) and necessary for different data-mining tasks (indexing, classification, prediction, etc). We believe that such a system will significantly help researchers in comparing newly introduced and proposed concepts with the existing ones.

3 Architecture of the Framework for Analysis and Prediction

FAP – Framework for Analysis and Prediction is designed to incorporate all main aspects of time-series mining in one place and to combine easily some or all of them. In the moment when the system will be completed, it will contain all important realisations of proposed concepts up to then, with the possibility to add easily newly proposed realisations. Currently, all important similarity measures, mentioned in introduction are implemented and the results of their evaluation are in more details explained in the next section.

The system is implemented in Java which will make FAP to be easier for maintenance and for future upgrades.

The package diagram of the system with the main dependencies between packages is given in Fig. 1. All packages are connected with package fap.core, but the dependencies are not shown because they would burden the image.

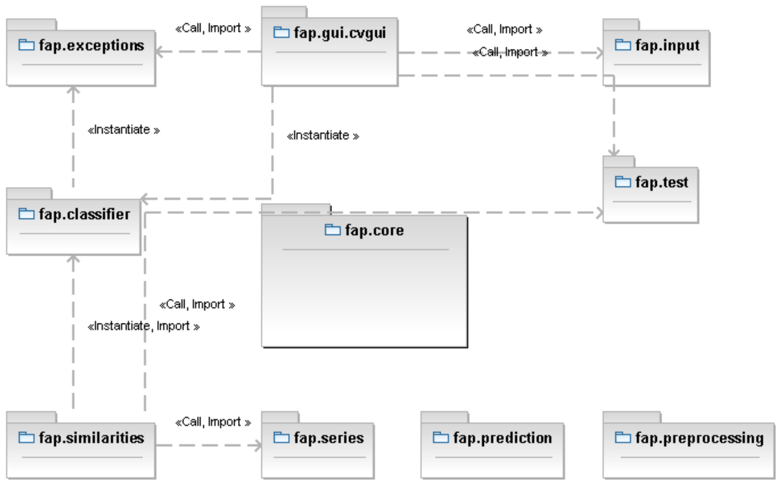

Fig. 1. Package diagram of the FAP system

The main part of the FAP system is package fap.core. It encloses the following subpackages: fap.core.math, fap.core.exceptions, fap.core.data, fap.core.series, fap.core.input, fap.core.similarities, fap.core.test, fap.core.prediction, fap.core.classifier and fap.core.preprocessing. These sub-packages contain basic classes and interfaces, which describe the functionality of the system. Almost every package from the fap.core package has its outside counterpart, where the implementations of desired concepts are stored. In the rest of this section, the contents and functionalities of the presented packages will be described.

The auxiliary packages are fap.core.math, fap.core.exceptions, fap.exceptions and fap.gui.cvgui. The first package contains implementations of additional mathematical concepts needed in the FAP (for example the implementation of polynomials, needed for the Spline representation). The fap.core.exceptions package contains the interface which describes the exceptions thrown by the FAP, while the concrete exceptions are implemented in fap.exceptions. Package fap.gui was intended to contain the implementations of various GUIs needed for different applications, while package fap.gui.cvgui contains our implementation of a GUI needed for cross-validation experiments concerning similarity measures (see the next section).

Package fap.core.data contains the basic implementations of a data point, serie and serie list. Package fap.core.series contains the raw implementation of a time series as a list of its points. It also contains an interface which describes what properties other representations should satisfy. The implementations of other representations are stored in package fap.series. Currently, the spline representation is realised as an additional one [10]. It is possible that one time serie has several representations.

The fap.core.input package is responsible for different kinds of inputs for time series. It contains interfaces which describe what classes responsible for the input should satisfy. The input can be from a file, network or any other stream. Classes which implement the basic functionality of CSV (Comma Separated Values) parser for an arbitrary separator are also stored here. The particular implementation of the CSV file reader is implemented in package fap.input.

Package fap.core.similarities contains an interface which describes the functionality of a class that computes one particular similarity measure. Some similarity measures have one or more parameters which need to be tuned based on the training part of a data set. The interface for tuning classes is also stored in this package. The actual implementations of similarity measures and their corresponding tuners are stored in package fap.similarities. The major similarity measures are implemented: L_p ($L_{1/2}$, L_1, L_2 and L_∞ distances are implemented separately for efficiency reasons), *DTW*, *CDTW* (*Constraint DTW*), *LCS*, *CLCS* (*Constraint LCS*), *ERP*, *EDR* and *Swale*. The results of experiments with these similarity measures are shown in the next section.

Similarly, packages fap.core.prediction, fap.core.classifier and fap.core.preprocessing contain the interfaces which describe prediction algorithms, classification algorithms and pre-processing tasks, respectively. The actual implementations of particular prediction algorithms, classification algorithms (e.g. kNN or 1NN classifier) and pre-processing tasks are stored in packages fap.prediction, fap.classifier and fap.preprocessing, respectively.

Package fap.core.test contains one interface which needs to be implemented when creating any statistical test in FAP. The particular implementations of these statistical tests are stored in package fap.test. We have implemented the following statistical tests: cross-validation with random splitting, cross-validation with stratified splitting and leave-one-out cross-validation.

4 Evaluation of the System

In this stage of FAP development we focused on time-series similarity measures and their complete implementation. Up to now we have implemented the following currently most important similarity measures: L_p, *DTW*, *CDTW*, *LCS*, *CLCS*, *ERP*, *EDR* and *Swale*.

All measures are very carefully implemented concerning efficiency and memory consumption. Furthermore, the measures based on dynamic programming are implemented using the Sakoe-Chiba Band [20] which limits the warping path to a band enclosed by two straight lines that are parallel to the diagonal of the warping matrix. This implementation has shown very good results especially concerning memory consumption.

Table 1. Error rates of similarity measures

Data Set	Num. of Crosses	L_1	L_2	L_∞	$L_{1/2}$	DTW	CDTW	ERP	EDR	LCS	CLCS	Swale
50words	5	0,387	0,421	0,558	0,377	0,367	0,302	0,399	0,260	0,270	0,286	0,270
Adiac	5	0,486	0,462	0,429	0,510	0,457	0,438	0,439	0,437	0,435	0,431	0,435
Beef	2	0,550	0,517	0,517	0,533	0,550	0,517	0,550	0,600	0,500	0,600	0,500
CBF	16	0,046	0,070	0,535	0,046	0,002	0,005	0,003	0,032	0,030	0,034	0,030
Coffee	2	0,161	0,161	0,107	0,250	0,125	0,143	0,161	0,196	0,304	0,286	0,304
ECG200	5	0,154	0,151	0,184	0,151	0,221	0,175	0,214	0,173	0,206	0,166	0,206
FaceAll	11	0,185	0,221	0,399	0,177	0,090	0,071	0,078	0,032	0,037	0,040	0,037
FaceFour	5	0,116	0,190	0,448	0,083	0,145	0,109	0,060	0,027	0,054	0,056	0,054
fish	5	0,302	0,283	0,304	0,328	0,309	0,270	0,190	0,155	0,169	0,184	0,169
Gun_Point	5	0,089	0,116	0,173	0,095	0,129	0,041	0,057	0,049	0,056	0,067	0,056
Lighting2	5	0,217	0,300	0,388	0,219	0,180	0,199	0,360	0,260	0,294	0,267	0,294
Lighting7	2	0,412	0,406	0,595	0,413	0,287	0,308	0,776	0,447	0,405	0,441	0,405
OliveOil	2	0,150	0,117	0,183	0,200	0,133	0,117	0,150	0,167	0,183	0,200	0,183
OSULeaf	5	0,452	0,464	0,518	0,456	0,427	0,422	0,391	0,260	0,242	0,252	0,242
SwedishLeaf	5	0,289	0,296	0,358	0,282	0,249	0,205	0,164	0,136	0,151	0,146	0,151
synthet-	5	0,145	0,132	0,223	0,166	0,012	0,015	0,035	0,058	0,044	0,048	0,044
Trace	5	0,286	0,341	0,459	0,249	0,014	0,018	0,160	0,133	0,049	0,090	0,049
Two_Patterns	5	0,038	0,102	0,796	0,049	0,000	0,000	0,000	0,001	0,001	0,001	0,001
wafer	7	0,004	0,005	0,020	0,005	0,016	0,005	0,008	0,004	0,005	0,005	0,005
yoga	11	0,162	0,159	0,181	0,167	0,151	0,140	0,129	0,113	0,124	0,119	0,124

In order to validate the correctness of our implementation, we have compared our results with the results presented in [4]. The objective evaluation method, proposed in [4], is used for comparison of different time series similarity measures: one nearest neighbour (1NN) classifier is used on labeled data to evaluate the efficiency of the similarity/distance measures. Each time series in the database has a correct class label. The classifier tries to predict the label of a time series as a label of its first nearest neighbor in the training set. There are several advantages with this approach:

- The underlying distance metric is crucial to the performance of the 1NN classifier; therefore, the accuracy of the 1NN classifier directly reflects the effectiveness of the similarity measure.
- The 1NN classifier is straightforward to be implemented and is parameter free, which reduces the possibility of appearance of errors in implementation.
- Among many other classification techniques, such as decision trees, neural networks, Bayesian networks, support vector machines, etc., some of the best results in time-series classification come from simple nearest neighbor methods [20].

In order to evaluate the effectiveness of each similarity measure, the following cross-validation algorithm has been applied. Firstly, a stratified random split has been used to divide the input data set into k subsets. However, we presented the results without randomization in order for them to be reproducible. The number of crosses in a cross-validation algorithm, k, is shown in Tab. 1. Secondly, the cross validation algorithm is applied, by using one subset at a time for the training set of the 1NN classifier, and the rest $k-1$ subsets as the test set. If the similarity measure requires parameter tuning, the training set is divided into two equal-sized stratified subsets where one is used for parameter tuning. Finally, the average error rate of 1NN classification over the k cross-validation fold is reported in Tab. 1.

For the purpose of testing similarity measures the FAP Graphical User Interface (GUI) has been implemented. The input for this FAP GUI application is the FAP experiment specification. On the basis of this specification the FAP will perform the experiments. The structure of the FAP experiment specification is very simple: the name of the specification is given in the first line while every subsequent line is reserved for one dataset. Every line consists of 9 attributes with the following meaning:

- **Name of the dataset.** FAP will look for the data sets in the subfolder with this name.
- **Number of splits (*k*) for cross-validation.** The default values for some or all datasets can be given in file “Datasets.csv” and can be omitted here. The values given in Tab. 1 are default values taken from [4].
- **Start fold.** The default value is 0.
- **End fold.** The default value is *k-1*.
- **Seed for the stratified random split.** Can be useful for reproduction of previous results. If it is not specified, the stratified split is performed without randomization.
- **Name of the similarity measure class.** The mappings between full class name and short names can be given in file “Distances.csv”.
- **Name of the tuner class** if the corresponding similarity measure requires tuning.
- **Frequency of tuning serialization.** Number which tells how frequently the main FAP object will be serialized during the tuning process. Default value is 20.
- **Frequency of testing serialization.** Number which tells how frequently the main FAP object will be serialized during the testing process. Default value is 20. This and the previous attribute introduce the possibility of serializing objects at runtime. This can be very useful when long computation must be terminated for some reason. In this case the computation can be continued later.

As an output, FAP application generates four files: “*SpecName*.txt” where the code of the FAP specification is stored, “*SpecName*.csv” where only the average error rates of data sets are stored, “*SpecName*.log” where the full logging information is stored and “*SpecName.scvp*” where the object serialized during runtime is stored. The last file also contains already computed results and can be migrated to another computer where the computation will be continued.

The experiments were conducted using the described methodology, on 20 diverse time series data sets. The data is provided by the UCR Time Series Repository [21], which includes the majority of all publicly available, labeled time series data sets in the world. The average error rates of the similarity measures on each data set are shown in Tab. 1. We have compared our results with the results presented in [4] in order to verify the correctness of our implementation. The only differences appear at the second or third decimal place which is the consequence of randomization in stratified random split of the cross-validation algorithm. These facts strongly support the correctness of our implementation, which has been our main goal.

5 Usefulness of FAP and Future Work

Time series are very useful and widely applicable in different domains. In the same time, working with time series is relatively complicated, mainly because of their high dimensions. So, it is very important to correctly choose appropriate methods and algorithms when working with specified time series. Motivated by these observations, we decided to create system FAP which will incorporate all important concepts of time-series analysis and mining. FAP is mainly intended for time-series researchers for testing and comparison of existing and newly introduced concepts. However, FAP system can also be useful for non-professionals from different domains as an assistance tool for choosing appropriate methods for their own data sets.

At this stage, FAP can be successfully used for testing and comparing existing similarity measures as well as for comparison of newly proposed measures with the existing ones. However, in order to be fully functional for time series mining, several algorithms and concepts need to be implemented.

The first step in the future development of FAP system is the implementation of wide range of existing techniques for time series representation: *DFT*, *SVD*, *DWT*, *PAA*, *APCA*, *SAX*, *IPLA*, while the spline representation is already implemented.

The next step of development will be the implementation of some pre-processing tasks. These tasks (*offset translation*, *amplitude scaling*, *removing linear trend*, *removing noise* etc.) can remove some inappropriate distortions like noise, trend, or seasonality. These distortions can greatly influence indexing of time series and spoil the results of similarity measures.

Although, it is reported that the 1NN technique gives among the best results for time series [20] it would be valuable to test other classification techniques like: decision trees, neural networks, Bayesian networks, support vector machines, etc.

In addition, the functionalities of FAP are not limited to classification only. We intend to extend it with other time-series mining task types like: clustering, forecasting, summarization, segmentation, etc. Also, it would be very useful to have some component for visualization.

6 Conclusion

Time-series analysis and mining has been a very popular research area in the past decade. This resulted in a huge amount of proposed techniques and algorithms. A great majority of techniques and algorithms were sporadically introduced and sometimes not correctly compared with their counterparts. This is the consequence of a lack of a quality open-source system which supports different aspects of time-series mining. For all these reasons, we created a universal framework (FAP) where all main concepts, like similarity measures, representation and pre-processing, will be incorporated. Such a framework would greatly help researchers in testing and comparing newly introduced concepts with the existing ones.

In this stage all main similarity measures are implemented. Currently, the modeling and implementation of representation techniques is in progress. The current version of FAP is available online as open-source software on the address: http://perun.pmf.uns.ac.rs/fap/.

We believe that the FAP system could be intensively used in research due to its numerous advantages. First, all important up to date concepts needed for time-series mining are integrated in one place. Second, modifications of existing concepts, as well as additions of newly proposed concepts could be obtained very easily (FAP is written in Java). Finally, FAP will be open source, and everyone will be invited to contribute with newly proposed concepts. This will insure that the system is always up to date and that all major techniques in time-series mining are supported.

References

1. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast Subsequence Matching in Time-Series Databases. In: SIGMOD Conference, pp. 419–429 (1994)
2. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, San Francisco (2005)
3. Keogh, E.J.: A Decade of Progress in Indexing and Mining Large Time Series Databases. In: VLDB, p. 1268 (2006)
4. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures. In: VLDB 2008, Auckland, New Zealand, pp. 1542–1552 (2008)
5. pong Chan, K., Fu, A.W.-C.: Efficient Time Series Matching by Wavelets. In: ICDE, pp. 126–133 (1999)
6. Keogh, E.J., Chakrabarti, K., Pazzani, M.J., Mehrotra, S.: Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. Knowl. Inf. Syst. 3(3), 263–286 (2001)
7. Keogh, E.J., Chakrabarti, K., Mehrotra, S., Pazzani, M.J.: Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. In: SIGMOD Conference, pp. 151–162 (2001)
8. Lin, J., Keogh, E.J., Wei, L., Lonardi, S.: Experiencing SAX: a novel symbolic representation of time series. Data Min. Knowl. Discov. 15(2), 107–144 (2007)
9. Chen, Q., Chen, L., Lian, X., Liu, Y., Yu, J.X.: Indexable PLA for Efficient Similarity Search. In: VLDB, pp. 435–446 (2007)
10. Kurbalija, V., Ivanović, M., Budimac, Z.: Case-Based Curve Behaviour Prediction. Software: Practice and Experience 39(1), 81–103 (2009)
11. Keogh, E.J., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowl. Inf. Syst. 7(3), 358–386 (2005)
12. Vlachos, M., Gunopulos, D., Kollios, G.: Discovering similar multidimensional trajectories. In: ICDE, pp. 673–684 (2002)
13. Chen, L., Ng, R.T.: On the marriage of lp-norms and edit distance. In: VLDB, pp. 792–803 (2004)

14. Chen, L., Özsu, M.T., Oria, V.: Robust and fast similarity search for moving object trajectories. In: SIGMOD Conference, pp. 491–502 (2005)
15. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations 11(1), 10–18 (2009)
16. `http://rapid-i.com/` (January 2010)
17. `http://www.sas.com` (January 2010)
18. Baiocchi, G., Distaso, W.: GRETL: Econometric software for the GNU generation. Journal of Applied Econometrics 18(1), 105–110
19. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing 26(1), 43–49 (1978)
20. Xi, X., Keogh, E.J., Shelton, C.R., Wei, L., Ratanamahatana, C.A.: Fast time series classification using numerosity reduction. In: ICML, pp. 1033–1040 (2006)
21. Keogh, E., Xi, X., Wei, L., Ratanamahatana, C.: The UCR Time Series dataset (2006), `http://www.cs.ucr.edu/~eamonn/time_series_data/`
22. Keogh, E., Pazzani, M.: Relevance Feedback Retrieval of Time Series Data. In: The Twenty-Second Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pp. 183–190 (1999)
23. Hochheiser, H., Shneiderman, B.: Interactive Exploration of Time-Series Data. In: Proc. of the 4th Int'l Conference on Discovery Science, Washington D.C., pp. 441–446 (2001)
24. van Wijk, J.J., van Selow, E.R.: Cluster and Calendar based Visualization of Time Series Data. In: Proceedings of IEEE Symposium on Information Visualization, pp. 4–9 (1999)
25. Weber, M., Alexa, M., Müller, W.: Visualizing Time-Series on Spirals. In: Proceedings of the IEEE Symposium on Information Visualization, pp. 7–14 (2001)
26. Lin, J., Keogh, E., Lonardi, S., Lankford, J.P., Nystrom, D.M.: Visually Mining and Monitoring Massive Time Series. In: Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Seattle, WA, pp. 460–469 (2004)
27. Findley, D.F., et al.: New Capabilities and Methods of the X-12-ARIMA Seasonal-Adjustment Program. Journal of Business & Economic Statistics, American Statistical Association 16(2), 127–152 (1998)
28. Gómez, V., Maravall, A.: Guide for using the program TRAMO and SEATS. In: Working Paper 9805, Research Department, Banco de España (1998)
29. Ghil, M., Allen, R.M., Dettinger, M.D., Ide, K., Kondrashov, D., Mann, M.E., Robertson, A., Saunders, A., Tian, Y., Varadi, F., Yiou, P.: Advanced spectral methods for climatic time series. Rev. Geophys. 40(1), 3.1–3.41 (2002)
30. Lütkepohl, H., Krätzig, M.: Applied Time Series Econometrics. Cambridge University Press, Cambridge (2004)
31. Morse, M.D., Patel, J.M.: An efficient and accurate method for evaluating time series similarity. In: SIGMOD Conference, pp. 569–580 (2007)