An Experimental Evaluation of the Adaptive Sampling Method for Time Series Classification and Clustering

Muhammad Marwan Muhammad Fuad
Forskningsparken 3, Institutt for kjemi, NorStruct
Department of Chemistry, The University of Tromsø - The Arctic University of Norway, NO-9037 Tromsø, Norway

Keywords: Adaptive Sampling, Classification, Clustering, Data Mining, Time Series.

Abstract: Adaptive sampling is a dimensionality reduction technique of time series data inspired by the dynamic programming piecewise linear approximation. This dimensionality reduction technique yields a suboptimal solution of the problem of polygonal curve approximation by limiting the search space. In this paper, we conduct extensive experiments to evaluate the performance of adaptive sampling in 1-NN classification and *k*-means clustering tasks. The experiments we conducted show that adaptive sampling gives satisfactory results in the aforementioned tasks even for relatively high compression ratios.

1 INTRODUCTION AND RELATED WORK

A *time series* S is a sequence of n indexed values

$$S = \langle s(t_1), s(t_2), \ldots, s(t_n) \rangle \quad (1)$$

Time series data mining arises in many domains including economics, medicine, finance, and astronomy. For this reason, time series data mining has received attention over the last years.

The major time series data mining tasks include query-by-content, clustering, classification, anomaly detection, motif discovery, segmentation, and prediction. Executing these tasks usually involves performing another fundamental task in data mining which is the *similarity search*. A similarity search problem consists of a database D, a query or pattern Q, which does not necessarily belong to D, and a tolerance ε that determines the closeness of the data objects to the query in order to be qualified as answers to that query. The principal component of the similarity search problem is the *distance metric* or the *similarity measure* which quantifies how much two data objects are close to each other. The *Euclidean Distance* (ED) (Figure 1) is a widely used time series distance metric. It is defined between two time series $S = \langle s_1, s_2, \ldots, s_n \rangle$ and $R = \langle r_1, r_2, \ldots, r_n \rangle$ as:

$$ED(S,R) = \sqrt[2]{\sum_{i=1}^{n} |s_i - r_i|^2} \quad (2)$$

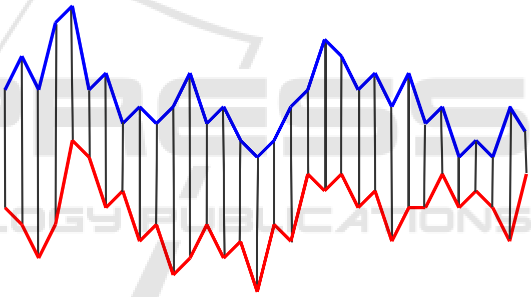

Figure 1: The Euclidean distance.

Another popular similarity measure (not a distance metric) used in time series data mining is the *Dynamic Time Warping* (DTW) (Guo and Siegelmann, 2004) (Figure 2). DTW is defined as:

$$DTW(i,j) = d(i,j) + \min \begin{cases} DTW(i, j-1) \\ DTW(i-1, j) \\ DTW(i-1, j-1) \end{cases} \quad (3)$$

where $1 \le i \le n, 1 \le j \le m$ □

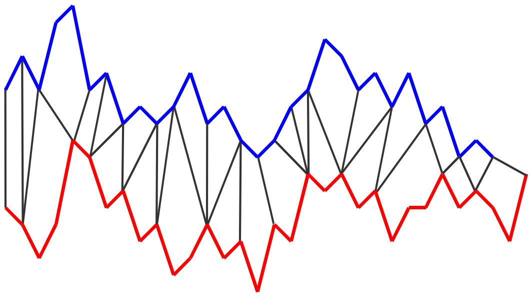

Figure 2: Dynamic time warping.

Fuad, M.
An Experimental Evaluation of the Adaptive Sampling Method for Time Series Classification and Clustering.
DOI: 10.5220/0005694600480054
In *Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2016)*, pages 48-54
ISBN: 978-989-758-173-1

A trivial solution to the similarity search problem is to compare every single time series in D against Q. This is known as *sequential scanning*. Obviously, this solution is not an efficient one given that modern time series databases are usually very large.

Dimensionality Reduction Techniques, also named *Representation Methods*, adopt the GEMINI framework (Faloutsos *et al*, 1994) (Figure 3) to process the similarity search problem of time series more efficiently. In GEMINI, the original time series are mapped onto low-dimension spaces, which reduces their dimensionality, and then the query is processed in those low-dimension spaces.

There are quite a few dimensionality reduction techniques in the literature. The most popular ones are *Piecewise Aggregate Approximation* (PAA) (Keogh *et al*, 2000) and (Yi and Faloutsos, 2000), and *Adaptive Piecewise Constant Approximation* (APCA) (Keogh *et al*, 2001).

One dimensionality reduction technique that is related to our experimental study is *Piecewise Linear Approximation* (PLA) (Morinaka *et al*, 2001) (Figure 4). PLA transforms the time series into Δ-SEALS (Δ-*bounded Sequence of Approximated Liner Segments*). The basic idea of the Δ-SEALS is to approximate the time series by a sequence of l linear segments. Each line segment is the longest possible linear segment whose accumulated error does not exceed a given deviation bound Δ, where the error is defined by the least square method.

```
Algorithm: range_query($Q,\varepsilon$)

 1. Transform the time series in
    database $D$ from the original
    $n$-dimensional space into
    a lower dimensional space of
    $N$ dimensions

 2. Define a lower bounding
    distance on the reduced space:

      $d^N(S_i,S_j) \leq d^n(S_i,S_j) \quad \forall S_i,S_j \in D$

 3. Eliminate all the time series
    for which we have $d^N(Q,S) > \varepsilon$
    to obtain a candidate answer
    set

 4. Apply $d^n$ to the candidate
    answer set and eliminate all
    the time series that are
    farther than $\varepsilon$ from $Q$ to get
    the final answer set.
```

Figure 3: The GEMINI algorithm for range queries.

Another dimensionality reduction technique related to the experimental section of this paper is *Discrete Fourier Transform* (DFT) (Agrawal *et al*, 1993), (Agrawal *et al*, 1995) (Figure 5). The basic idea of DFT is that a time series can be represented using complex numbers called the *Fourier Coefficients*, so a time series of 256 dimensions, for instance, can be represented by 128 complex Fourier coefficients. However, the first coefficients are the most significant and the most representative ones, so the other Fourier coefficients can be truncated without much loss of information.

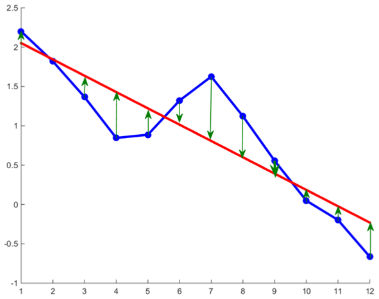

Figure 4: Δ-SEALS.

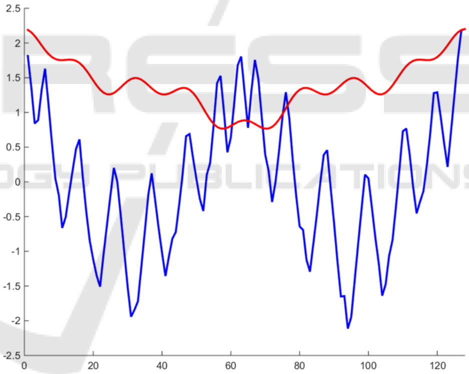

Figure 5: DFT using 8 coefficients.

In this paper we present an extensive experimental evaluation of a certain dimensionality reduction technique which is the adaptive sampling method of time series. We show how this method can give good results in time series classification and clustering tasks compared with other methods.

The rest of the paper is as follows; in Section 2 the adaptive sampling method is introduced. In Section 3 we present the experiments we conducted. In Section 4 we conclude this paper with final remarks and directions for future work.

2 THE ADAPTIVE SAMPLING METHOD

In (Marteau and Ménier, 2006) the authors presented the *Adaptive Multiresolution Simplification* model of times series data which was inspired by the *Dynamic Programming Piecewise Linear Approximation* model presented in (Marteau and Gibet, 2005) and derived from (Perez and Vidal, 1994) and (Kolesnikov and Franti, 2003). This adaptive model yields a suboptimal solution of the problem of polygonal curve approximation by limiting the search space.

We briefly present here an outline of the model: given an *n*-dimensional time series S, the objective is to find an approximation $S_{\hat{\theta}}$ of S that satisfies:

$$\hat{\theta} = \underset{\theta}{ArgMin}\big(E(S, S_\theta)\big) \qquad (4)$$

where E is the root mean square error between S and the model S_θ . The search is limited to the family of piecewise linear and continuous functions $\{S_\theta\,(n)\}$. The successive segments have to be contiguous, so that the end of one segment is the beginning of the next one. The authors apply the dynamic programing algorithm to select the optimal set of parameters $\hat{\theta} = \{\hat{m}_i\}$. This is done as follows: first, we define the compression ratio of the piecewise approximation as:

$$\rho = 1 - \frac{|\{n_i\}|}{|\{S(n)\}|} \times \frac{\rho + 1}{\rho} \qquad (5)$$

where $S(n) \in \mathbb{R}^n, \forall n$

Given the value of ρ and the width of the time window $w = \left|\{S(n)\}_{n \in \{1,2,\dots,w\}}\right|$, the number of piecewise linear segments $N = |\{n_i\}| - 1$ is known in this case.

Let $\theta(k)$, by definition, be the parameters of a piecewise approximation containing k segments, and let $\delta(k,i)$ be the minimal error of the best piecewise linear approximation containing k segments and covering the time window $\{1,2,\dots,w\}$, $\delta(k,i)$ can then be written as:

$$\delta(k,i) = \underset{\theta(k)}{Min}\left\{\sum_{n=1}^{i}\left\|S_{\theta(k)}(n) - S(n)\right\|^2\right\} \qquad (6)$$

According to Bellman's optimality principle (Bellman, 1957), the above term can be decomposed as:

$$\delta(k,i) = \underset{k-1 \le n_k \le i}{Min}\{d(n_k, i) \; + \delta(k-1, n_k)\} \qquad (7)$$

where

$$d(n_k, i) = \sum_{n=n_k}^{i}\left\|R_{k,i}(n) - S(n)\right\|^2$$

and $R_{k,i}(n) = \big(S(i) - S(n_k)\big) \times \frac{n - n_k}{i - n_k} + S(n_k)$ is the linear segment between $S(i)$ and $S(n_k)$.

Recursion is initialized by observing that:

$$\delta(k,i) = 0 \qquad ,\forall k, \forall i < k \qquad (8)$$

At the end of the recursion, we get the optimal piecewise linear approximation; i.e. the set of time stamps of the end points of the linear segments:

$$\hat{\theta}(k) = \underset{\theta(k)}{ArgMin}\left\{\sum_{n=1}^{w}\left\|S_{\theta(k)}(n) - S(n)\right\|^2\right\} \qquad (9)$$

with the minimal error:

$$\delta(k,w) = \sum_{n=1}^{w}\left\|S_{\hat{\theta}(k)}(n) - S(n)\right\|^2 \qquad (10)$$

The complexity of the algorithm is $O(k, w^2)$. In order to reduce this complexity the search window can be limited by using a lower bound $lb = Max\{i - band, 0\}$ for each step i, and where $band$ is a user-defined parameter:

$$\delta(k,i) = \underset{lb \le n_k \le i}{Min}\{d(n_k, i) \; + \delta(k-1, n_k)\} \qquad (11)$$

In practice we choose $band = \frac{2w}{k}$.

3 EXPERIMENTS

Before we present the outcome of our experiments, we briefly introduce the two main data mining tasks on which we tested the adaptive sampling method. The two tasks are classification and clustering.

Classification: The goal of classification (also called *supervised learning*) is to assign an unknown object to one out of a given number of classes or categories (Bunke and Kraetzl, 2003). Classification is based on four fundamental components (Gorunescu , 2006): 1- Class, which is a categorical variable representing the 'label' put on the object after its classification. 2- Predictors, which are represented by the attributes of the data to be classified. 3- Training dataset, which is the set of data containing values for the two previous components, and is used for 'training' the model to recognize the appropriate class based on available predictors. 4- Testing dataset, containing new data that will be classified by the model constructed in the previous steps.

Table 1: 1-NN classification errors for different compression ratios.

Dataset	Method	ρ						
		0%	5%	10%	25%	50%	75%	90%
Synthetic Control	DTW-PLA	0.007	0.003	0.010	0.036	0.083	0.147	0.217
	ED-PLA	0.120	0.113	0.127	0.147	0.210	0.273	0.290
	DFT	0.097	0.127	0.150	0.203	0.273	0.367	0.410
	DFT-PLA	NA	0.097	0.103	0.127	0.203	0.217	0.273
Gun-Point	DTW-PLA	0.093	0.087	0.100	0.133	0.180	0.220	0.287
	ED-PLA	0.087	0.067	0.080	0.113	0.147	0.193	0.220
	DFT	0.087	0.113	0.133	0.160	0.200	0.233	0.307
	DFT-PLA	NA	0.087	0.113	0.140	0.160	0.200	0.273
CBF	DTW-PLA	0.003	0.080	0.021	0.024	0.037	0.071	0.100
	ED-PLA	0.148	0.128	0.148	0.173	0.188	0.209	0.356
	DFT	0.112	0.147	0.184	0.209	0.234	0.382	0.398
	DFT-PLA	NA	0.112	0.136	0.173	0.209	0.234	0.263
Trace	DTW-PLA	0.000	0.000	0.010	0.040	0.090	0.160	0.190
	ED-PLA	0.240	0.190	0.210	0.240	0.360	0.380	0.430
	DFT	0.186	0.210	0.350	0.380	0.410	0.430	0.480
	DFT-PLA	NA	0.170	0.210	0.310	0.370	0.390	0.420
Lightning-2	DTW-PLA	0.131	0.117	0.131	0.164	0.183	0.217	0.262
	ED-PLA	0.246	0.250	0.250	0.283	0.311	0.367	0.383
	DFT	0.213	0.246	0.295	0.333	0.377	0.410	0.426
	DFT-PLA	NA	0.213	0.217	0.233	0.283	0.317	0.367
Lightning-7	DTW-PLA	0.274	0.247	0.274	0.301	0.342	0.397	0.438
	ED-PLA	0.425	0.425	0.429	0.443	0.466	0.486	0.514
	DFT	0.405	0.414	0.443	0.466	0.514	0.629	0.729
	DFT-PLA	NA	0.384	0.425	0.429	0.471	0.571	0.685
ECG	DTW-PLA	0.230	0.210	0.230	0.240	0.250	0.260	0.270
	ED-PLA	0.120	0.090	0.110	0.130	0.160	0.200	0.230
	DFT	0.120	0.130	0.150	0.170	0.210	0.240	0.260
	DFT-PLA	NA	0.100	0.110	0.140	0.160	0.190	0.220
Adiac	DTW-PLA	0.396	0.389	0.399	0.427	0.484	0.574	0.608
	ED-PLA	0.389	0.385	0.393	0.420	0.473	0.567	0.595
	DFT	0.385	0.420	0.470	0.567	0.592	0.687	0.709
	DFT-PLA	NA	0.385	0.413	0.475	0.560	0.575	0.673

One of the most popular classification techniques of time series is k −*Nearest Neighbor Classification* ($k-$NN). In $k-$NN the query is classified according to the majority of its nearest neighbours (Vlachos and Gunopulos, 2003). Usually k is taken to be 1, thus applying a first nearest-neighbor ($1-$NN) rule using leaving-one-out cross validation. This means that every data object is compared to the other data objects in the dataset. If the $1-$NN does not belong to the same class, the error counter is incremented by 1.

Clustering: It is the task of partitioning the data objects into groups of similar objects. Clustering (also called *unsupervised learning*) is different from classification in that in clustering we do not have target variables. Instead, clustering algorithms seek to segment the entire data set into relatively homogeneous subgroups or clusters (Larose, 2005).

There are different categories of clustering algorithms, the one we are interested in in this paper is *Partitioning-based Clustering*. In particular, we are interested in k-means clustering. In k-means clustering we have a set of n data points in d-dimensional space R^d and an integer k and the problem is to determine a set of k points, the centroids, in R^d so as to minimize the mean distance from each data point to its nearest center (Kanungo *et al*, 2002).

Table 2: *k*-means clustering quality for different compression ratios.

Dataset	**Method**	ρ						
		0%	5%	10%	25%	50%	75%	90%
Synthetic Control	DTW-PLA	0.990	0.995	0.962	0.858	0.726	0.618	0.528
	ED-PLA	0.649	0.656	0.617	0.573	0.479	0.392	0.289
	DFT	0.723	0.719	0.685	0.623	0.543	0.441	0.378
	DFT-PLA	NA	0.839	0.718	0.653	0.587	0.456	0.389
Gun-Point	DTW-PLA	0.879	0.896	0.814	0.684	0.527	0.438	0.373
	ED-PLA	0.473	0.484	0.447	0.402	0.348	0.289	0.226
	DFT	0.489	0.491	0.469	0.424	0.351	0.312	0.261
	DFT-PLA	NA	0.890	0.806	0.657	0.511	0.423	0.351
CBF	DTW-PLA	0.983	0.985	0.916	0.847	0.683	0.513	0.426
	ED-PLA	0.602	0.610	0.595	0.548	0.463	0.372	0.253
	DFT	0.643	0.656	0.617	0.579	0.512	0.436	0.324
	DFT-PLA	NA	0.787	0.746	0.675	0.610	0.476	0.358
Trace	DTW-PLA	0.843	0.835	0.802	0.736	0.619	0.494	0.327
	ED-PLA	0.453	0.436	0.405	0.387	0.322	0.254	0.211
	DFT	0.510	0.476	0.447	0.406	0.359	0.287	0.244
	DFT-PLA	NA	0.675	0.634	0.576	0.468	0.329	0.259
Lightning-2	DTW-PLA	0.958	0.969	0.879	0.808	0.612	0.465	0.389
	ED-PLA	0.589	0.602	0.566	0.526	0.445	0.361	0.222
	DFT	0.618	0.624	0.598	0.554	0.481	0.384	0.254
	DFT-PLA	NA	0.917	0.858	0.765	0.548	0.411	0.354
Lightning-7	DTW-PLA	0.817	0.786	0.739	0.675	0.628	0.463	0.301
	ED-PLA	0.437	0.415	0.388	0.354	0.311	0.232	0.204
	DFT	0.458	0.435	0.403	0.389	0.355	0.285	0.254
	DFT-PLA	NA	0.617	0.565	0.532	0.441	0.332	0.264
ECG	DTW-PLA	0.985	0.976	0.878	0.739	0.623	0.476	0.390
	ED-PLA	0.674	0.680	0.652	0.611	0.543	0.425	0.354
	DFT	0.662	0.653	0.635	0.578	0.521	0.398	0.322
	DFT-PLA	NA	0.670	0.647	0.597	0.537	0.402	0.338
Adiac	DTW-PLA	0.672	0.684	0.646	0.585	0.449	0.386	0.269
	ED-PLA	0.362	0.380	0.343	0.321	0.287	0.224	0.189
	DFT	0.380	0.395	0.364	0.334	0.305	0.245	0.195
	DFT-PLA	NA	0.395	0.351	0.344	0.311	0.264	0.211

More formally, the k-means clustering error can be measured by:

$$E = \sum_{i=1}^{k} \sum_{j=1}^{n_j} d(u_{ij}, c_i) \qquad (12)$$

Where u_{ij} is the j^{th} point in the i^{th} cluster, and n_j is the number of points in that cluster. The quality of the k-means clustering increases as the error given in (Eq. 12) decreases.

The number of clusters is determined by the user, application-dependent, or given by a certain clustering validity measure. □

We conducted experiments on classification and clustering tasks of time series data available at (Chen *et al*, 2015). This archive makes up between 90% and 100% of all publicly available, labeled time series data sets in the world, and it represents the interest of the data mining/database community, and not just one group (Ding *et al*, 2008).

The length of the time series varies between 60 (Synthetic_control) and 637 (Lightning-2). The size of datasets varies between 61 (Lightning-2) and 900 (CBF), so as we can see, we tested our method on a wide range of datasets of different lengths and sizes to avoid getting biased results.

Table 1 shows the classification error (the smaller the better) in a $1-\mathrm{NN}$ classification task of the adaptive sampling method applied to DFT (c.f. Section 1), and also applied using DTW and the Euclidean distance. The experiments are conducted for different compression ratios ρ, where the ρ = 0% indicates no adaptive sampling is performed (the method is turned off)

The results show that DTW is adapted to the classification task in question. Adaptive sampling gave acceptable results even for compression ratios between 25% and 50%. For dataset ECG the results were quite acceptable even for a very high compression ratio (ρ = 90%).

ED was also adapted to adaptive sampling as the classification error was in general acceptable for compression ratio of 50%.

When applying adaptive sampling to DFT, the results were always better than the original method for all datasets and for all compression ratios.

An interesting phenomenon that we noticed is that in many cases, applying adaptive sampling for a compression ratio of 5% gave better results than the raw data themselves. We believe the reason for this is that compassion has a positive effect of smoothing the data.

We also conducted *k*-means clustering experiments on the same datasets and for the same compression ratios. Table 2 shows the *k*-means clustering quality (the larger the better) of the datasets we tested. As we can see from Table 2, the results of the *k*-means clustering are similar to those of $1-\mathrm{NN}$ classification. They show that DTW is the most adapted method for the *k*-means clustering task and again the adaptive sampling yielded acceptable results even for compression ratios between 25% and 50% for almost all the datasets tested. The results, however, degraded in most cases for high compression ratios.

ED was also adapted to adaptive sampling as the quality of *k*-means clustering was still acceptable even for a compression ratio of 50%.

As was the case with classification, adaptive sampling improved the performance of DFT for all datasets and for all compression ratios. When applying adaptive sampling to DFT, the results were always better than the original method for all compression ratios and for all compression ratios.

The smoothing effect that appeared in the classification task experiments for a compression ratio of 5% also appeared in the *k*-means clustering experiments.

4 CONCLUSIONS

In this paper, we conducted extensive experiments on the adaptive sampling method of time series in $1-\mathrm{NN}$ classification and *k*-means clustering tasks. These experiments were conducted on a variety of time series datasets, using the Euclidean distance, the dynamic time warping, and the discrete Fourier transform (DFT). The output of our experiments shows that even when using high compression ratios, the performance of the adaptive sampling method is still acceptable in the two aforementioned time series data mining tasks. In some cases, the adaptive sampling method yielded acceptable results even for a high compression ratio.

In the future, we intend to study the impact of adaptive sampling on other time series data mining tasks and also to compare it with other time series dimensionality reduction techniques.

REFERENCES

Agrawal, R., Faloutsos, C., & Swami, A. (1993): Efficient similarity search in sequence databases. *Proceedings of the 4th Conf. on Foundations of Data Organization and Algorithms.*

Agrawal, R., Lin, K. I., Sawhney, H. S. and Shim, K. (1995): Fast similarity search in the presence of noise, scaling, and translation in time-series databases. *In Proceedings of the 21st Int'l Conference on Very Large Databases*. Zurich, Switzerland.

Bellman, R., (1957): *Dynamic programming*. Princeton University Press, Princeton, NJ.

Bunke, H., Kraetzl, M. (2003): Classification and detection of abnormal events in time series of graphs. In: Last, M., Kandel, A., Bunke, H. (eds*.): Data mining in time series databases.* World Scientific.

Chen,Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., and Batista, G. (2015). The UCR Time Series Classification Archive. URL. www.cs.ucr.edu/~eamonn/time_series_data.

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. (2008): Querying and mining of time series data: experimental comparison of representations and distance measures. *In Proc of the 34th VLDB*.

Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. (1994): Fast subsequence matching in time-series databases. *In Proc. ACM SIGMOD Conf.*, Minneapolis.

Gorunescu, F. (2006): Data mining: concepts, models and techniques, *Blue Publishing House*, Cluj-Napoca.

Guo, A.Y., and Siegelmann, H. (2004): Time-warped longest common subsequence algorithm for music retrieval, *in Proc. ISMIR*.

Keogh, E., Chakrabarti, K., Pazzani, M. & Mehrotra,S. (2000): Dimensionality reduction for fast similarity search in large time series databases. *J. of Know. and Inform. Sys*.

Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2001): Locally adaptive dimensionality reduction for similarity search in large time series databases. *SIGMOD pp 151-162.*

Kolesnikov, A., and Franti, P. (2003): Reduced-search dynamic programming for approximation of polygonal curves. *Pattern Recognition Letters*.

Kanungo, T., Netanyahu, N.S., Wu, A.Y. (2002): An efficient *k*-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern analysis and machine intelligence* 24(7).

Larose, D.T. (2005): *Discovering knowledge in data: an introduction to data mining*. New York, Wiley.

Marteau, P.F., and Gibet, S. (2005): Adaptive sampling of motion trajectories for discrete task-based analysis and synthesis of gesture. *In Proc. of Int. Gesture Workshop*, Vannes, France.

Marteau, P.F., Ménier, G. (2006): Adaptive multiresolution and dedicated elastic matching in linear time complexity for time series data mining, *Sixth International conference on Intelligent Systems Design and Applications* (*ISDA 2006*), Jinan Shandong, China, 16-18 October.

Morinaka, Y., Yoshikawa, M., Amagasa, T., and Uemura, S. (2001): The L-index: an indexing structure for efficient subsequence matching in time sequence databases. *In Proc. 5th PacificAsia Conf. on Knowledge Discovery and Data Mining*, pages 51-60.

Perez, J. C., and Vidal, E. (1994): Optimum polygonal approximation of digitized curves. *Pattern Recognition Letters*.

Vlachos, M., and Gunopulos, D. (2003): Indexing time-series under conditions of noise. In: Last, M., Kandel, A., Bunke, H. (eds.): *Data mining in time series databases*. World Scientific.

Yi, B, K., & Faloutsos, C. (2000): Fast time sequence indexing for arbitrary Lp norms. *Proceedings of the 26st International Conference on Very Large Databases*, Cairo, Egypt.