Global Analysis and Discrete Mathematics
Volume 3, Issue 1, pp. 65–73
ISSN: 2476-5341

Algorithms for Segmenting Time Series

Maliheh Khatami*, Farzaneh Akbarzadeh

Received: 17 February 2018 / Accepted: 21 October 2018

Abstract As with most computer science problems, representation of the data is the key to efficient and effective solutions. Piecewise linear representation has been used for the representation of the data. This representation has been used by various researchers to support clustering, classification, indexing and association rule mining of time series data. A variety of algorithms have been proposed to obtain this representation, with several algorithms having been independently rediscovered several times. In this paper, we examine the techniques and then introduce the best-known algorithm.

Keywords Piecewise Linear Representation, Effective Solutions, Algorithms, Time Series.

Mathematics Subject Classification (2010) 62M10 · 68Qxx

1 Introduction

In recent years, there has been an explosion of interest in mining time series databases. As with most computer science problems, representation of the data is the key to efficient and effective solutions. representations of time series have been used, from Fourier Transforms [1,2], Wavelets [3], Symbolic Mappings [4–6] and Piecewise Linear Representation (PLR) [7,8]. Piecewise

*Corresponding author
Maliheh Khatami
School of Engineering, Damghan University, Damghan, Iran.
E-mail: M_ khatami@du.ac.ir

Farzaneh Akbarzadeh
School of Computer Engineering and Information Technology, Shahrood University, Shahrood, Iran.

Linear Representation makes the storage, transmission and computation of the data more efficient . PLR refers to the approximation of a time series T, of length n, with K straight lines. Figure 1 contains two examples. Specifically, in the context of data mining, the piecewise linear representation has been used to:

- Support fast exact similarly search [2].
- Support novel distance measures for time series, including "fuzzy queries" [9,10], weighted querie[11], multi-resolution queries [12,13], dynamic time warping [14] and relevance feedback [15].
- Support concurrent mining of text and time series [16].
- Support novel clustering and classification algorithms[11].
- Support change point detection [17].

In this work, we will refer to these types of algorithm, which input a time series and return a piecewise linear representation, as segmentation algorithms [9]. The segmentation problem can be framed in several ways.

- Given a time series T, produce the best representation using only K segments.
- Given a time series T, produce the best representation such that the maximum error for any segment does not exceed some user specified threshold, max-error.
- Given a time series T, produce the best representation such that the combined error of all segments is less than some user-specified threshold, total-max-error.

It should be noted that algorithms do not support all of these specifications. Data mining researchers, who needed to produce a piecewise linear approximation, have typically either independently rediscovered an algorithm or used an approach suggested in related literature. For example,from the fields of cartography or computer graphics [18].

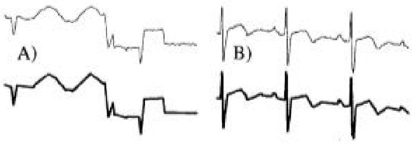


Fig. 1 Two time series and their piecewise linear representation. A) Space Shuttle Telemetry. B) Electrocardiogram (ECG)

In this paper, we review the three major segmentation approaches in the literature and provide an extensive empirical evaluation on a very heterogeneous collection of data-sets from Iran-Khodro, Darusazi-Osveh, Bank-Day and Overall-Index share. We explain the approaches and compare them.

2 Background

In this section, we describe the three major approaches to time series segmentation in detail. Almost all the algorithms have 2 and 3 dimensional analogues, which ironically seem to be better understood. most time series segmentation algorithms can be grouped into one of the following cases with different implementation details:

- Sliding Windows: A segment is grown until it exceeds some error bound. The process repeats with the next data point not included in the newly approximated segment.
- Top-Down: The time series is recursively partitioned until some stopping criteria is met.
- Bottom-Up: Starting from the finest possible approximation, segments are merged until some stopping criteria is met.
  We want to approximate a time series with straight lines, We use two methods for this.
- Linear Interpolation: Here the approximating line for the subsequence $T[a:b]$ is simply the line connecting t_a , and t_b. This can be obtained in constant time.
- Linear Regression: Here the approximating line for the subsequence T[a:b] is taken to be the best fitting line in the least squares sense[9]. This can be obtained in time linear in the length of segment.

The two techniques are illustrated in figure 2. Linear interpolation tends to closely align the endpoint of consecutive segments, giving the piecewise approximation a "smooth" look. In contrast, piecewise linear regression can produce a very disjointed look on some data-sets. The aesthetic superiority of linear interpolation, together with its low computational complexity has made it the technique of choice in computer graphic applications [18]. However, the quality of the approximating line, in terms of Euclidean distance, is generally inferior to the regression approach.

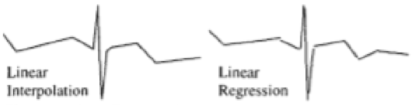
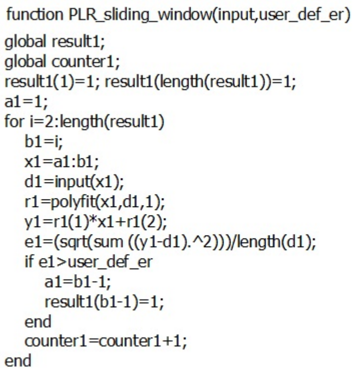


Fig. 2 Two 10-segment approximations of electrocardiogram data

2.1 **The sliding window algorithm**

In the last decade, the theoretical study of the sliding window model was developed to advance applications with very large input and time-sensitive output. In some practical situations, input might be seen as an ordered sequence, and it is useful to restrict computations to recent portions of the input. Examples include the analysis of recent tweets and time series of the stock market [19]. introduced the sliding window model that assumes that the input is a stream (i.e., the ordered sequence) of data elements and divides the data elements into two categories: active elements and expired elements. The Sliding Window algorithm is attractive because of its great simplicity, intuitiveness and particularly the fact that it is an online algorithm [20]. Depending on the error measure used, there may be other optimizations possible. Vullings et al. noted that since the residual error is monotonically non-decreasing with the addition of more data points, one does not have to test every value of i from 2 to the final chosen value [21]. They suggest initially setting i to s, where s is the mean length of the previous segments. If the guess was pessimistic then the algorithm continues to increment i as in this algorithm. Otherwise they begin to decrement i until the measured error is less than max_ error. This optimization can greatly speed up the algorithm if the mean length of the segment is large in relation to the standard deviation of their length. The pseudocode for the algorithm is shown in figure 3.

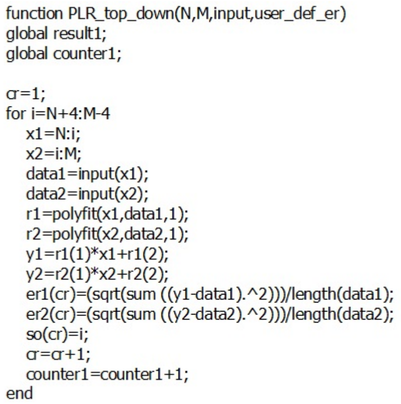


```
function PLR_sliding_window(input,user_def_er)
global result1;
global counter1;
result1(1)=1; result1(length(result1))=1;
a1=1;
for i=2:length(result1)
    b1=i;
    x1=a1:b1;
    d1=input(x1);
    r1=polyfit(x1,d1,1);
    y1=r1(1)*x1+r1(2);
    e1=(sqrt(sum ((y1-d1).^2)))/length(d1);
    if e1>user_def_er
        a1=b1-1;
        result1(b1-1)=1;
    end
    counter1=counter1+1;
end
```

Fig. 3 The generic Sliding Window algorithm

2.2 **The top-down algorithm**

The Top-Down algorithm works by considering every possible partitioning of the times series and splitting it at the best location. Both subsections are then tested to see if their approximation error is below some user-specified threshold. If not, the algorithm recursively continues to split the subsequences until all the segments have approximation errors below the threshold. The pseudocode for the algorithm is shown in figure 4. Variations on the Top-Down algorithm(including the 2− dimensional case)was independently introduced in several fields in the early 1970 s. In cartography, it is known as the Douglas-Peucker algorithm [22]; in image processing, it is known as Ramers algorithm [23]. Most researchers in the machine learning/data mining community are introduced to the algorithm in the classic text book by Duda and Harts, which calls it "Iterative End Points Fits" [23]. In the data mining community, the algorithm has been used by [13] to support a framework for mining sequence databases at multiple abstraction levels. Shatkay and Zdonik use it to support approximate queries in time series databases [10].



```
function PLR_top_down(N,M,input,user_def_er)
global result1;
global counter1;

cr=1;
for i=N+4:M-4
    x1=N:i;
    x2=i:M;
    data1=input(x1);
    data2=input(x2);
    r1=polyfit(x1,data1,1);
    r2=polyfit(x2,data2,1);
    y1=r1(1)*x1+r1(2);
    y2=r2(1)*x2+r2(2);
    er1(cr)=(sqrt(sum ((y1-data1).^2)))/length(data1);
    er2(cr)=(sqrt(sum ((y2-data2).^2)))/length(data2);
    so(cr)=i;
    cr=cr+1;
    counter1=counter1+1;
end
```

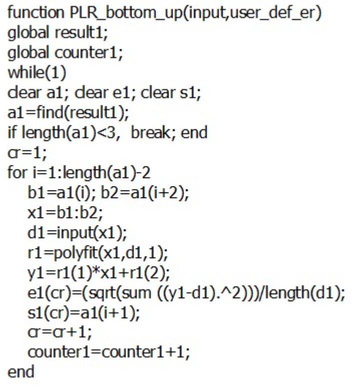
Fig. 4 The top-down algorithm

On real world data-sets with any amount of noise, the approximation is greatly over fragmented. Lavrenko et al. uses the Top-Down algorithm to support the concurrent mining of text and time series [16]. They attempt to discover the influence of news stories on financial markets. Their algorithm contains some interesting modifications including a novel stopping criteria based on the t-test. Finally Smyth and Ge use the algorithm to produce a representation which can support a Hidden Markov Model approach to both change point detection and pattern matching.

2.3 The bottom-up algorithm

The Bottom-Up algorithm is the natural complement to the Top-Down algorithm. The algorithm begins by creating the finest possible approximation of the time series, so that $n/2$ segments are used to approximate the $n-$length time series. Next, the cost of merging each pair of adjacent segments is calculated, and the algorithm begins to iteratively merge the lowest cost pair until a stopping criteria is met. When the pair of adjacent segments i and

$i + l$ are merged, the algorithm needs to perform some bookkeeping. First, the cost of merging the new segment with its right neighbor must be calculated. In addition, the cost of merging the $i - 1$ segments with its new larger neighbor must be recalculated. The pseudocode for the algorithm is shown in figure 5. Two and three-dimensional analogues of this algorithm are common in the field of computer graphics where they are called decimation methods [18]. In data mining, the algorithm has been used extensively by two of the current authors to support a variety of time series data mining tasks [11,15]. In medicine, the algorithm was used by Hunter and McIntosh to provide the high level representation for their medical pattern matching system [16].



```
function PLR_bottom_up(input,user_def_er)
global result1;
global counter1;
while(1)
clear a1; clear e1; clear s1;
a1=find(result1);
if length(a1)<3,  break; end
cr=1;
for i=1:length(a1)-2
    b1=a1(i); b2=a1(i+2);
    x1=b1:b2;
    d1=input(x1);
    r1=polyfit(x1,d1,1);
    y1=r1(1)*x1+r1(2);
    e1(cr)=(sqrt(sum ((y1-d1).^2)))/length(d1);
    s1(cr)=a1(i+1);
    cr=cr+1;
    counter1=counter1+1;
end
```

Fig. 5 The bottom-up algorithm

3 Empirical comparison of the major segmentation algorithms

In this section, we will provide an extensive empirical comparison of the three major algorithms. It is possible to create artificial data-sets that allow one of the algorithms to achieve zero error (by any measure), but forces the other two approaches to produce arbitrarily poor approximations the .

In contrast, testing on purely random data forces the all algorithms to produce essentially the same results. To overcome the potential for biased results, we tested the algorithms on a very diverse collection of data-sets. Figures 6, 7 and 8 illustrate the Iran-Khodro data-sets used in the experiments.

4 Conclusions

We have carried out the first extensive review and empirical comparison of time series segmentation algorithms from a data mining perspective. We have shown the most popular approach, Sliding Windows, generally produces very

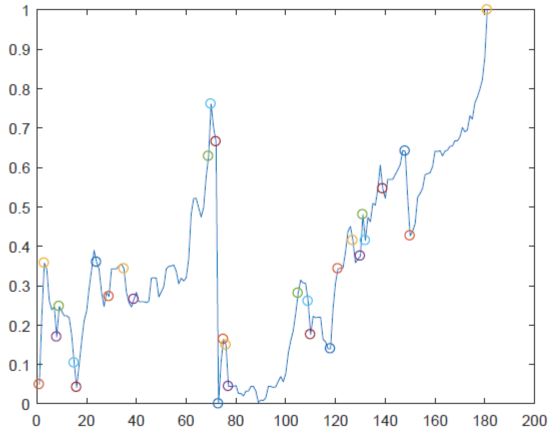


Fig. 6 Iran-Khodro- Sliding window

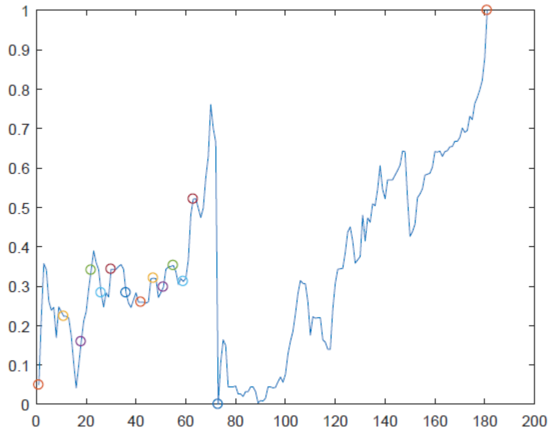


Fig. 7 Iran-Khodro-Top down

poor results, and that while the second most popular approach, Top-Down, can produce reasonable results, it does not scale well. In contrast, the least well known, Bottom-Up, approach produces excellent results and scales linearly with the size of the data-set.

	Iran-Khodro	Darusazi-Osveh	Bank-Dey	Overall-Index
Topdown	0.0056958	0.0026541	0.0033911	0.0031629
Bottomup	0.0056968	0.0041389	0.0036852	0.0029973
Slidingwindow	0.0027282	0.0042957	0.0031934	0.0022434

Table 1. Total error

	Iran-Khodro	Darusazi-Osveh	Bank-Dey	Overall-Index
Topdown	14	29	23	25
Bottomup	4	9	33	39
Slidingwindow	29	15	48	46

Table 2. Number of break points

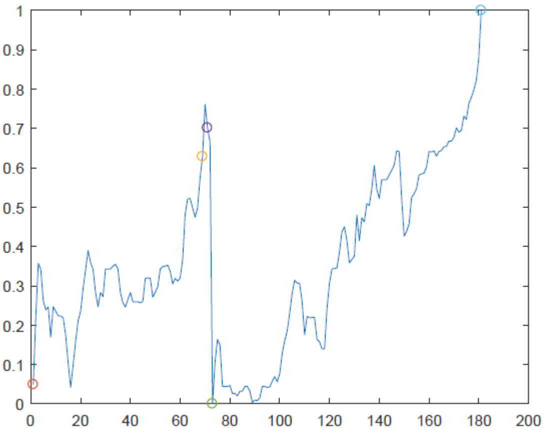


Fig. 8 Iran-khodro- Bottom up

	Iran-Khodro	Darusazi-Osveh	Bank-Dey	Overall-Index
Topdown	436	2179	1023	1426
Bottomup	4002	3977	3509	3302
Slidingwindow	180	180	180	180

Table 3. Total runs (number of approximation)

	Iran-Khodro	Darusazi-Osveh	Bank-Dey	Overall-Index
Topdown	1.763447	8.085701	4.083690	4.202938
Bottomup	7.946990	7.066469	6.116836	3.778965
Slidingwindow	0.403124	0.450311	0.433831	0.234717

Table 4. Total time (seconds)

	0.001	0.005	0.01	0.02	0.04	0.06	0.08	0.1	0.5
Topdown	0.0029084	0.002908	0.0056958	0.019336	0.019336	0.019336	0.019336	0.019336	0.019336
Bottomup	0.0014275	0.0014741	0.0056968	0.019632	0.019632	0.019632	0.019632	0.019632	0.019632
Slidingwindow	0.0000386	0.0004905	0.0027282	0.010543	0.229937	0.019632	0.019632	0.019632	0.019632

Table 5. Effect of user defined error on total error according to Iran-Khodro share

	0.001	0.005	0.01	0.02	0.04	0.06	0.08	0.1	0.5
Topdown	32	31	14	2	2	2	2	2	2
Bottomup	90	74	4	1	1	1	1	1	1
Slidingwindow	159	103	29	12	3	1	1	1	1

Table 6. Effect of user defined error on total break points according to Iran-Khodro share

References

1. R. Agrawal, C. Faloutsos, A. Swami, Efficient similarity search in sequence databases, Proceedings of the 4th Conference on Foundations of Data Organization and Algorithms, 69–84 (1993).
2. E. Keogh, K. Chakrabarti, M. Pazzani, Dimensionality reduction for fast similarity search in large time series databases, Journal of Knowledge and Information Systems, 3(3), 263–286 (2000).
3. K. Chan, W. Fu, Efficient time series matching by wavelets, Proceedings of the 15th IEEE International Conference on Data Engineering, (1999).

4. R. Agrawal, K.I. Lin, H.S. Sawhney, K. Shim, Fast similarity search in the presence of noise, scaling, and translation in times-series databases, Proceedings of 21th International Conference on Very Large Data Bases, 490–500 (1995).
5. X. Ge and P. Smyth, Segmental Semi-Markov Models for Endpoint Detection in Plasma Etching, IEEE Transactions on Semiconductor Engineering, (2001).
6. G. Das, K. Lin, H. Mannila, G. Renganathan, P. Smyth, Rule discovery from time series, Proceedings of the 3th International Conference of Knowledge Discovery and Data Mining, 16–22 (1998).
7. C. Perng, H. Wang, S. Zhang, S. Parker, Landmarks: a new model for similarity-based pattern querying in time series databases, Proceedings of 16th International Conference on Data Engineering, (2000).
8. P.C. Chang, C.Y. Fan, C.H. Liu, Integrating a piecewise linear representation method and a neural network model for stock trading points prediction, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 39 (1), 80–92 (2009).
9. H. Shatkay, Approximate Queries and Representations for Large Data Sequences, Technical Report in Department of Computer Science, Brown University (1995).
10. H. Shatkay and S. Zdonik, Approximate queries and representations for large data sequences, Proceedings of the 12th IEEE International Conference on Data Engineering, 546–553 (1996).
11. E. Keogh and M. Pazzani, An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback, Proceedings of the 41th International Conference of Knowledge Discovery and Data Mining, 239–241(1998).
12. H. Wu, B. Salzberg, D. Zhang, Online event-driven subsequence matching over financial data streams, in: Proceeding of the SIGMOD, Stream Management, 23–34 (2004).
13. C. Li, P. Yu, V. Castelli , A framework for mining sequence database at multiple abstraction levels, Proceedings of the 9th International Conference on Information and Knowledge Management, 267–272 (1998).
14. E. Keogh, S. Chu, D. Hart, M. Pazzani, An online algorithm for segmenting time series, in: Proceedings of the IEEE International Conference on Data Mining, 289–296 (2001).
15. E. Keogh, M. Pazzani, Relevance feedback retrieval of time series data, Proceedings of the 22th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, 183–190 (1999).
16. V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, J. Allan, Mining of concurent text and time series, Proceedings of the 61th International Conference on Knowledge Discovery and Data Mining, 37–44 (2000).
17. N. Sugiura, R.T. Ogden, Testing change- points with linear trend communications , Simulation and Computation, 23, 287–322 (1994).
18. P.S. Heckbert, M. Garland, Survey of polygonal surface simplification algorithms, Multiresolution Surface Modeling Course, Proceedings of the 24th International Conference on Computer Graphics and Interactive Techniques, 1–24 (1997).
19. E. Keogh, S. Chu, D. Hart, M. Pazzani, Segmenting Time Series: A Aurvey and Novel Approach, Data Mining in Time Series Databases, World Scientific Publishing Company, (2004).
20. A. Koski, M. Juhola, M. Meriste, Syntactic recognition of ECG signals by attributed finite automata, Pattern Recognition, 28 (12), 1927–1940 (1995).
21. H.J.L.M. Vullings, M.H.G. Verhaegen, H.B. Verbruggen, ECG segmentation using time-warping. Proceedings of the 2th International Symposium on Intelligent Data Analysis, 275–285 (1997).
22. D.H. Douglas, T.K. Peucker, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, Canadian Cartographer, 10(2), 112–122 (1973).
23. U. Ramer, An iterative procedure for the polygonal approximation of planar curves, Computer Graphics and Image Processing, 244–256 (1972).