An improved algorithm for segmenting online time series with error bound guarantee

Article *in* International Journal of Machine Learning and Cybernetics · June 2016

DOI: 10.1007/s13042-014-0310-9

CITATIONS
11

READS
188

4 authors, including:

Huanyu Zhao
Hebei Academy of Sciences
31 PUBLICATIONS **236** CITATIONS
SEE PROFILE

Hao Lan Zhang
NIT, Zhejiang University
122 PUBLICATIONS **469** CITATIONS
SEE PROFILE

Yun Xue
Hubei University
62 PUBLICATIONS **523** CITATIONS
SEE PROFILE

Int. J. Mach. Learn. & Cyber. (2016) 7:365–374
DOI 10.1007/s13042-014-0310-9

ORIGINAL ARTICLE

An improved algorithm for segmenting online time series with error bound guarantee

Huan-yu Zhao · Guang-xia Li · Hao-lan Zhang · Yun Xue

Received: 7 June 2014 / Accepted: 27 October 2014 / Published online: 7 November 2014

Abstract In many real application, the volume of time series data increases seriously. How to store and process data becomes more interesting and challenge things. Effective representations can make storage less, processing more easily. In this paper, we contribute to construct a new piecewise linear approximation algorithm for segmenting online time series with error bound guarantee. To beat our targets, we combine a disconnected segment strategy into Feasible Space Window method, and to test our algorithm, we compare with algorithms that adopts the above strategies on both real and synthetic data sets. The time complexity of our algorithm is O(n) and the number of segments is smaller than FSW algorithm on all tested data sets.

Keywords Error bound · Piecewise linear approximation · Time series · Data mining

List of symbols

δ	A given error bound on each data point
P	A time series
F	A set of linear line
p_k	The k-th data point in time series
$\widetilde{p_k}$	The k-th intersection by represent line
$\underline{p_i}$	Data point with deleted tolerant error
$\overline{p_i}$	Data point with added tolerant error
p_{start}	The starting point
p_{next}	The next coming data point
p_{t_e}	The data point at which the FSW becomes empty
p'_{t_e}	The joint point at t_e
u	An upper boundary line
l	A lower boundary line
n_s	The number of segments
$\underline{p_{t_e}}$	The intersection point at t_e by l
$\overline{p_{t_e}}$	The intersection point at t_e by u

1 Introduction

A time series is a sequence of data points with a time stamp. Time series exist in many application domains such as medicine [1], economic [2, 11] and weather [3]. The volume of time series stream data grows rapidly in some fields [12]. This leads that we can not store the whole time series and may loss some important information. So the huge data not only poses a challenge to store and represent, but also makes the processing methodologies [4, 17] more complicated. In addition, some time series may be generated continually. This need to analyze data in almost real time. Therefore, for time series streams, the algorithm must have the ability to deal with dynamic data.

Piecewise linear approximation (PLA) is a widely used method because of its simplicity. PLA uses line segments to represent a time series data stream. This strategy is a

H. Zhao
Institute of Applied Mathematics, Hebei Academy of Sciences, Shijiazhuang 050081, China

G. Li
Department of Information Engineering, Shijiazhuang University of Economics, Shijiazhuang 050031, China

H. Zhang (✉)
NIT, Zhejiang University, Ningbo 315100, China
e-mail: haolan.zhang@nit.zju.edu.cn

Y. Xue
Guangzhou Higher Education Mega Centre, South China Normal University, Guangzhou 500006, China

good preprocessing of mass data set and can make the processing methods relatively easy.

There are many algorithms in PLA. According the current literatures, we category PLA into two classes: offline segmentation and online segmentation. The offline segmentation methods need to collect the whole time series data before conducting. Some heuristic strategies, such as top-down/bottom-up algorithm and evolutionary computation [14], are introduced to solve the problem. Since most time series are attained in large amounts dynamically. The online segmentation methods are more fit real-time working environment. The online ones deal with each point incrementally, and the algorithms are more effective and efficient.

If we consider the criteria of error, the online segmentation method can be categorized into two aspects: the holistic approximation error and the guaranteed error bound. There are many algorithms with the holistic approximation error in literatures. Bellman [15] proposes that the original question can be formulated into dynamic programming under giving a time series and the number of segment. Due to the time complexity, other methods [16] are presented. The sliding window (SW) algorithm [5] is widely used online algorithm. It processes by using the first data point of a time series as the starting data point of a segment, using coming point and the first point to construct the approximated line, and computing the holistic approximation error. When the error exceeds the predefined one, new segment is determined. Do the steps until the end point of the time series is reached. To reduce the time complexity of SW, some researchers improve the algorithms. Keogh et al. [6, 13] fusions the SW and the Bottom/Up strategies and proposes a hybrid algorithm. Palpanas et al. [16] gives a methodology to achieve the time complexity.

About the guaranteed error bound, there exits three papers. Liu et al. [7] introduces a new segmentation criterion called feasible space (FS) and proposes a novel method called feasible space window (FSW). This algorithm (denoted ConnAlg) can find the farthest endpoint of a segment along the incoming data stream. Comparing the SW and Bottom/Up methods, it can produce far fewer segments, achieve higher representation quality and reduce the time complexity to linear function. Qi et al. [9] proposes an adaptive mechanism to determine the most compact candidate function for each part of a time series. Based on FS, another segmentation criterion called feasible coefficient space (FCS) is given to adapt polynomial functions instead of linear functions to approximate segments. Although the new algorithm called adaptive approximation (AA) can achieve the most compact functions, it increases the time complexity. Moreover, it may make data mining more complexity. In addition, Xie et al. [8] give an interesting research to generate disconnect one. They proof that their algorithms (denoted DisconnAlg) generate optimal disconnect segments with error bound guarantee. In this paper, We introduce this strategy to segment connect one.

The contributions of this paper are that giving an improved algorithms with the same time complexity O(n) to reduce the number of segments of ConnAlg for segmenting online time stream. Our main idea is as follows. Firstly, Constructing the first and second segment by DisconnAlg and obtaining two represent line. Secondly, Initialing the point which is intersection of the above-second line with the end tolerant range and segmenting by ConnAlg. With the end of the current segment, Initialing the feasible range which is constructed by ConnAlg at the end of point and segmenting by DisconnAlg. Thirdly, doing the ConnAlg and DisconnAlg iteratively with all data. The oblivious advantage of the idea is that the initialing feasible space is bigger than the ConnAlg.

The rest of the paper is organized as follows: Sect. 2, gives the question formulation, the symbol representations and preliminary, Sect. 3, advocates our hybrid algorithm; Sect. 4, reports the results of our experimental study; Sect. 5, concludes the paper and gives future works.

2 Symbol representations, question formulation and related works

2.1 Symbol representations

Before proposing our work, the symbols used in this paper are listed in Abbreviation.

2.2 Question formulation

Given a time series $P = (p_1, p_2, \ldots, p_n)$, an error bound δ, our problem is to divide P into n_s continuous segments $(p'_{t_1}, p'_{t_2}, \ldots, p'_{t_{n_s}})$, where p'_{t_i} is the i-th joint point at t_i, $i \in [1, \ldots, n_s]$.

Here, the segment $(p'_{t_i}, \ldots, p'_{t_j})$ is the i-th one and satisfies that:

1. Each segment is represented by a line with the error bound on each data point, formally, $dis\tan ce(\tilde{p}_k - p_k) \leq \delta$;
2. n_s is minimized.

2.3 Preliminary

To solve the above-problem, Liu et al. [7] propose the feasible space window (FSW) algorithm. This algorithm

achieves the longest segment with an error bound guarantee on each data by a concept named feasible space (FS). FS is an area in the data value space of a time series where any straight line in this area can approximate each data point of the corresponding segment within a given error bound.

Figure 1 shows an example of the FS. Suppose p_1 is the starting point of a time series. With new data point p_2 coming, u_1 is the upper line that passes p_1 and $\overline{p_2}$, l_1 is the lower line that passes p_1 and $\underline{p_2}$. The area between u_1 and l_1 is the FS. When new data points are read, the FS is incrementally updated. For example, the point p_3 is read and u_2, l_2 are constructed. Currently, the FS is shrieked to the area between u_2 and l_1. In other words, the FS is more smaller with data points coming. When the FS becomes empty at new data point p_{t_e} with time stamp t_e, the new segment is generated. At this time, the point before p_{t_e} is the end point of the segment and the intersection at time stamp before t_e on the represent line is the starting point of next segment. The segments are attained by doing the above-mentioned steps with all data points.

In this process, the FS is constructed by intersecting area between u and l. Note that this intersection is not always done with all points. Figure 2 shows the situation. The last FS is equips to the new one.

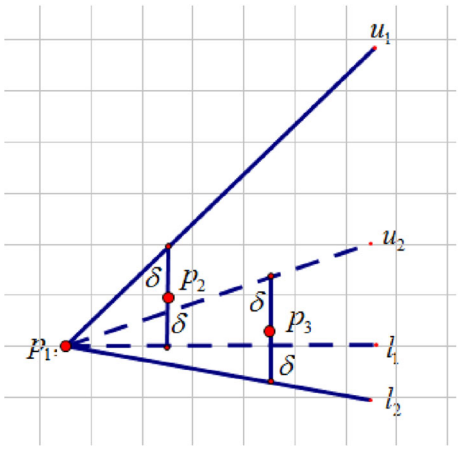

Fig. 1 The feasible space

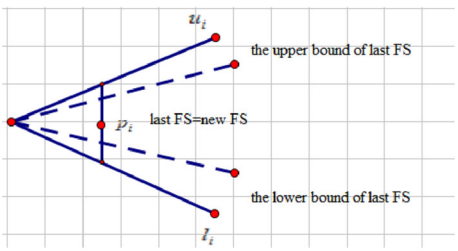

Fig. 2 The situation of not checking

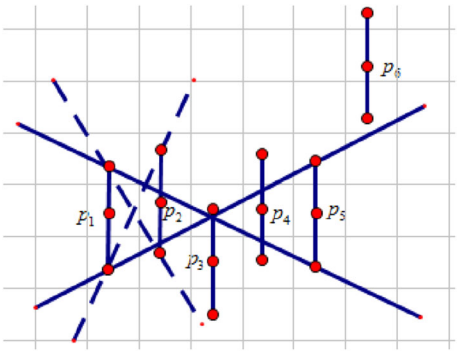

Fig. 3 The process of [8]

The FSW algorithm is used to attain continue line. The paper [8] gives a strategy to construct disconnect line optimally. The main idea is that determining the area of feasible line segments, which is incrementally update with coming data until the new point can not be approximated with error bound.

Figure 3 explains the process. Suppose p_1 is the starting point of a time series. With new data point p_2 coming, u_1 is the upper line that passes $\underline{p_1}$ and $\overline{p_2}$, l_1 is the lower line that passes $\overline{p_1}$ and $\underline{p_2}$ (as depicting dotted line). When new data are read, the extreme lines will be updated (as depicting solid line). Actually, the extreme lines are the smallest and largest slope lines which are constructed by deleted and added tolerant error points. If the new point can not generate two extreme lines which guarantees all points with error bound, the new segment is coming and the current point is the starting of next segment. The segments are attained by doing the above-mentioned steps with all data points.

Note that update the extreme lines do not check all lines which are constructed by the current tolerant points and before ores. Some details can be found in [8].

Now we give the relationships between the two strategies.

1. The former one is used to generate the connected segments, while the later one generates the disconnected ones.
2. The former one is limited the represented line must pass the first point, while the later one is not.
3. The later one is optimal for disconnected segments, while the former is not.
4. At the end of each segment, the represented lines constructed by the two strategies are not only. In other words, there are feasible ranges at the end.
5. The time complexity of two strategies are $O(n)$.

3 The proposed algorithm

In this section, we give our hybrid algorithm (call connAlgI) which is the fusion of the strategies mentioned above.

In Sect. 2, some relationships are discussed between two strategies. The fourth one is a key idea for our algorithm. As Fig. 4 shows, both construct a feasible range line whose length is less 2δ at the end of segments. If we use the later strategy to segment (that is the feasible range is regarded as initial error bound), the initial area of feasible line segment is more larger. It is convenient to find the farthest segmenting point and attain fewer segments for all points.

Now, how to make the represent line to connect is another key problem. For solving this, our idea is as follows: In our process, two represent lines are constructed at the same time. In other words, when the current segment is achieved (the FS or the area of feasible line is empty), we do not attain the represent line (both strategies can construct the lines by the extreme lines and their intersected points). The line for current segment is got by the extreme lines intersection and the point which is intersected by represent line of next segment and the end feasible range of current segment. This step is done iteratively for all points.

Note that the above idea can be implemented by the later algorithm, but we do not adopt. We use the two strategies to segment alternately. This is because that although the time complexity of two algorithms are the same, the consume time is different. For amount of calculation, the former one is less than the later. So our strategy can consume less time. Our method are showed in Figs. 5 and 6. Figure 5 explains the process of the first and second segments. Figure 6 explains the iterative process.

Based above discussion, the improved hybrid algorithm is described as follows:

Step 1: Construct the first and second segment by the DisconnAlg strategy. When the second segment is determined, the corresponding represent line is achieved by extreme lines (determine the slope by $\frac{\rho_1+\rho_2}{2}$, where ρ_1 and ρ_2 are the slops of extreme lines) and their intersection.

Step 2: At this time, the point which is intersected by the line and feasible range (for example, A in Fig. 5 or B in Fig. 6) is regarded as the starting point of the FWS algorithm. After FWS process, the feasible range at the end is regarded as the error bound of the later strategy. If the later process is done, the corresponding represent lines can be determined as step 1.

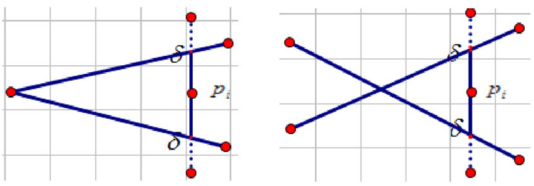

Fig. 4 The feasible range of ConnAlg and DisConnAlg at the end

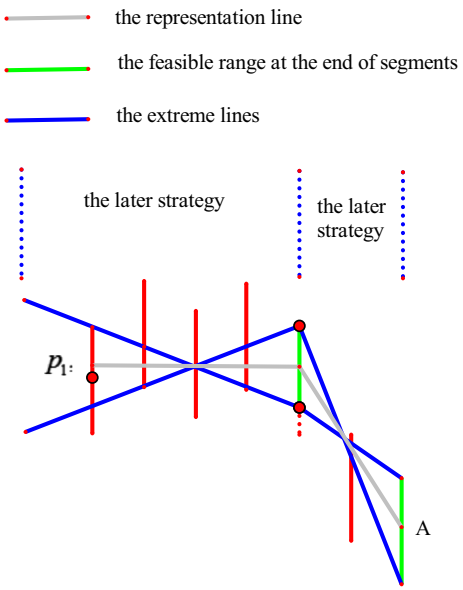

Fig. 5 The processing of constructing the first and second segments

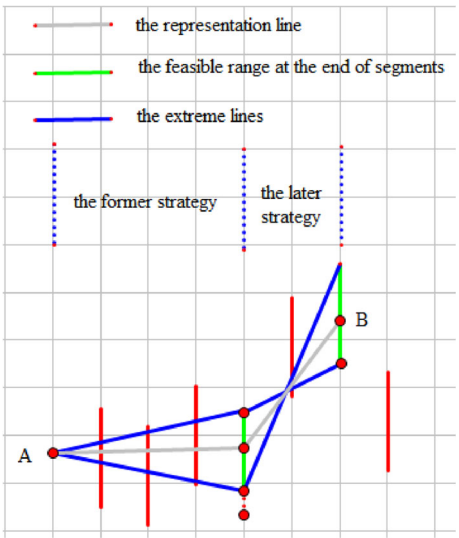

Fig. 6 The processing of constructing other segments

Step 3: If all points is not deal with, go step 2; otherwise, go to step 4.

Step 4: Output the number of segments, represent lines and others.

4 The experiment

In this section, we do a lot of experiments base on both real and synthetic data sets. The real ones are from the UCR TIME Series Archive [10]. We choose 43 data sets and the max length of them is 8441901.

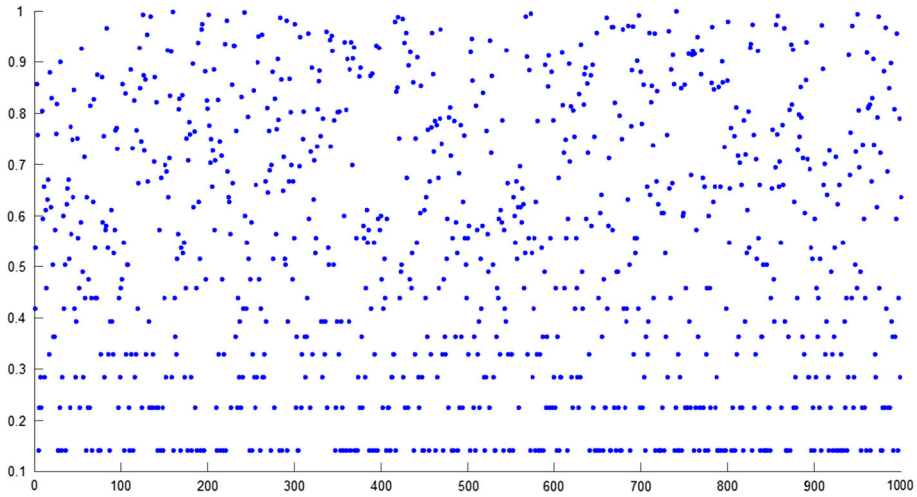

Fig. 7 The ZipF data

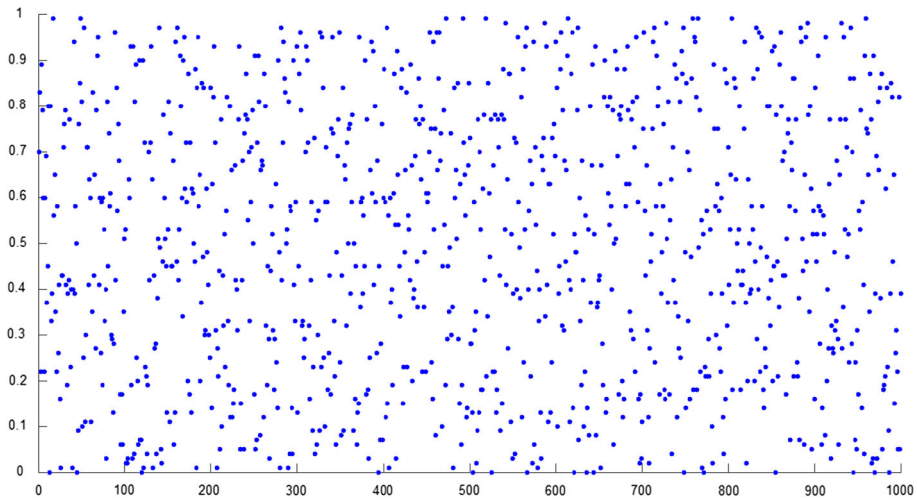

Fig. 8 The random data

The synthetic data set contains two kinds whose are Zipf and Random distribution as depicted in Figs. 7 and 8. The parameter of Zipf is 1.25 and the random is in [0, 1].

In this paper, all algorithms are implemented in Eclipse with C++. The CPU of computer is Intel Core I5 and the memory is 8 GB.

4.1 Number of segments and processing time

Our algorithm is comparing with the FSW algorithm on all datasets and record the number of segments generated and processing time. For the fairness of all datasets, we set the error bound as the 2.5, 5 % of the value range of the series. The results are listed in Tables 1 and 2.

Regarding the number of segments, we can conclude from tables that ConnAlgI is better than A definitely. The reason is that the strategy in [8] can find the local optimal segmenting point with the error bound guarantee. Another reason is, in most cases, we adopt the feasible range at the end point of current segment to increase the feasible area of represent lines for the next segment, which can attain less segments that the FSW algorithm.

Regarding the processing time, we can see from tables that our algorithm is more than the FSW one and the relationship is about three times. Through analyzing our algorithm, we conclude that the time complexity of ConnAlgI is O(n). As our discussions in session II.C, it is the same as ConnAlg. Due to more amount of computation, the constructing time is more. Note that the unit of consuming time is ms, we think it do not influence on application.

4.2 The effect of error bound

In this subsection, we fix the data length 10^6 and carry out the algorithms with varied error bounds. The error bound is choose 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5 and 0.55 respectively. The number of segments and the construction time based on different error bounds are showed in Figs. 9 and 10.

Table 1 The 2.5 % situation

Data	Length	Number of segment		Processing time (ms)	
		ConnAlg	ConnAlgI	ConnAlg	ConnAlgI
50words	123,305	3,541	3,350	28	86
Adiac	69,207	2,293	2,279	15	61
Beef	14,131	95	92	3	12
Coffee	8,037	739	709	2	5
OliveOil	17,131	617	610	4	14
CBF	116,100	76,893	72,666	44	79
ECG200	9,700	2,761	2,472	3	6
FaceAll	223,081	49,636	46,954	62	158
FaceFour	30,888	7,230	6,350	9	20
FISH	81,201	1,189	1,139	19	57
Gun_Point	22,650	1,370	1,320	6	17
Lighting2	38,918	2,219	1,908	9	21
Lighting7	23,360	2,615	2,283	6	16
OSULeaf	103,576	5,598	5,417	25	71
SwedishLeaf	80,626	6,941	6,674	20	59
synthetic_c-ontrol	18,301	12,792	12,143	7	12
Trace	27,600	1,290	1,146	7	16
Two_Patter-ns	516,000	246,252	234,601	174	405
yoga	1,281,000	52,887	51,037	301	858
wafer	943,092	66,900	65,397	226	884
ChlorineCo-ncentration	641,281	142,499	128,440	179	520
Cricket_X	117,391	8,049	7,278	28	75
Cricket_Y	117,391	7,659	6,955	28	75
Cricket_Z	117,391	8,065	7,276	28	74
DiatomSize-Reduction	105,877	2,675	2,614	24	77
ECGFiveDay-s	117,958	11,542	11,068	30	84
Haptics	336,645	6,150	5,840	77	211
InlineSkate	1,035,650	10,230	8,805	232	557
MedicalIma-ges	76,000	8,224	7,760	19	54
MoteStrain	106,420	13,347	12,743	27	80
SonyAIBOR-obotSurfac-e	42,671	18,148	16,949	14	31
SonyAIBOR-obotSurfac-eII	62,899	30,477	28,593	21	45
Symbols	397,006	11,105	10,833	91	272
TwoLeadEC-G	94,537	15,576	14,836	25	66
WordsSyno-nyms	172,898	6,361	6,249	40	129
CinC_ECG_t-orso	2,263,201	24,343	22,484	507	1,324
FacesUCR	270,601	60,427	57,179	75	191
ItalyPower-Demand	25,726	12,769	12,118	9	19
MALLAT	2,403,626	81,401	79,754	552	1,615
StarLightCu-rves	8,441,901	98,145	95,538	1,873	5,690
uWaveGest-ureLibrary-_X	1,131,913	41,651	40,407	262	787
uWaveGest-ureLibrary-_Y	1,131,913	42,859	41,631	261	793
uWaveGest-ureLibrary-_Z	1,131,913	43,914	42,767	262	789
ZipF	100,000	93,192	92,076	43	62
Random	100,000	92,860	91,959	42	62

Table 2 The 5 % situation

Data	Length	Number of segment		Processing time (ms)	
		ConnAlg	ConnAlgI	ConnAlg	ConnAlgI
50words	123,305	1,752	1,566	27	75
Adiac	69,207	2,012	1,868	16	36
Beef	14,131	89	89	3	12
Coffee	8,037	484	463	1	6
OliveOil	17,131	400	398	4	13
CBF	116,100	52,948	47,278	39	82
ECG200	9,700	1,433	1,259	3	6
FaceAll	223,081	34,464	32,192	58	149
FaceFour	30,888	3,500	3,188	8	20
FISH	81,201	880	877	18	59
Gun_Point	22,650	1,091	1,085	6	17
Lighting2	38,918	966	787	9	23
Lighting7	23,360	1,209	1,058	5	13
OSULeaf	103,576	4,156	4,051	24	72
SwedishLeaf	80,626	5,002	4,810	19	58
synthetic_c-ontrol	18,301	9,121	8,306	6	13
Trace	27,600	825	787	7	17
Two_Patter-ns	516,000	179,144	162,608	159	429
yoga	1,281,000	34,756	33,603	295	864
wafer	943,092	55,925	53,695	223	779
ChlorineCo-ncentration	641,281	73,531	56,866	163	443
Cricket_X	117,391	4,570	4,196	27	72
Cricket_Y	117,391	4,360	4,012	27	73
Cricket_Z	117,391	4,545	4,109	27	70
DiatomSize-Reduction	105,877	2,034	2,014	24	74
ECGFiveDay-s	117,958	8,379	7,963	28	84
Haptics	336,645	4,235	3,821	76	207
InlineSkate	1,035,650	5,989	5,348	235	587
MedicalIma-ges	76,000	6,099	5,781	19	57
MoteStrain	106,420	9,009	8,482	27	80
SonyAIBOR-obotSurfac-e	42,671	11,716	10,982	13	30
SonyAIBOR-obotSurfac-eII	62,899	20,790	19,390	19	44
Symbols	397,006	7,503	7,084	91	288
TwoLeadEC-G	94,537	10,410	9,836	24	69
WordsSyno-nyms	172,898	4,441	4,292	40	126
CinC_ECG_t-orso	2,263,201	13,731	12,909	506	1,335
FacesUCR	270,601	40,293	37,393	70	188
ItalyPower-Demand	25,726	8,833	8,332	8	19
MALLAT	2,403,626	51,071	48,653	541	1,626
StarLightCu-rves	8,441,901	61,779	60,575	1,836	5,839
uWaveGest-ureLibrary-_X	1,131,913	28,463	27,478	253	780
uWaveGest-ureLibrary-_Y	1,131,913	28,649	27,657	252	761
uWaveGest-ureLibrary-_Z	1,131,913	29,960	28,959	254	768
ZipF	100,000	87,099	85,068	42	65
Random	100,000	86,374	84,389	42	63

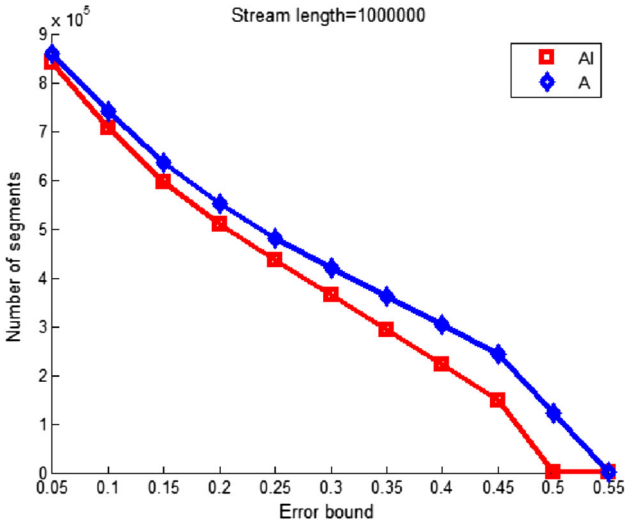

Fig. 9 The effect of error bound on number of segments

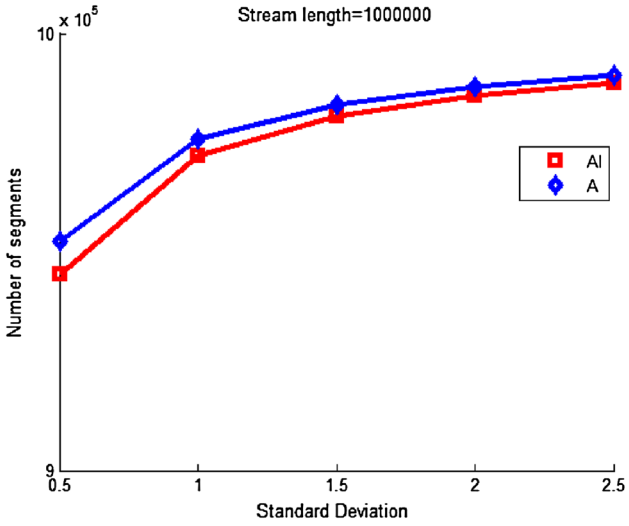

Fig. 11 The effect of data variation on number of segments

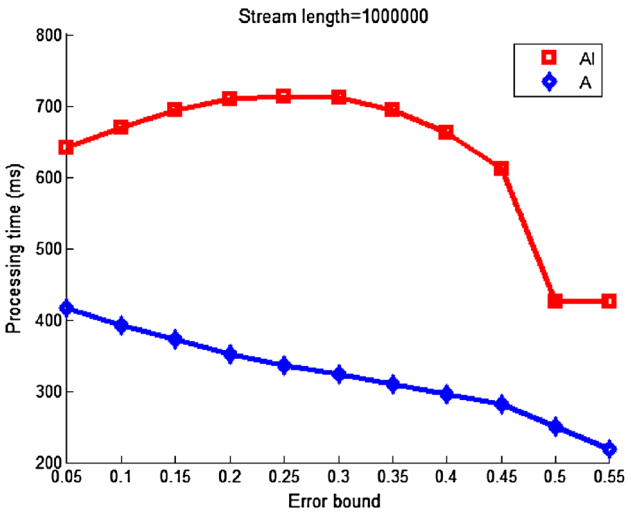

Fig. 10 The effect of error bound on processing time

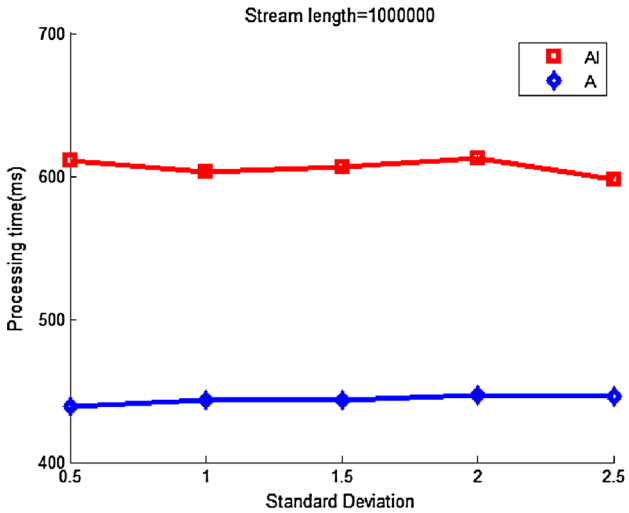

Fig. 12 The effect of data variation on processing time

In Fig. 9, we find that our algorithm is better than the FSW algorithm with all error bound. This is because that we consider the initial area of the feasible line for each segment. The bigger of initial area, the farther of the end point. So for the same time series, the number of segments can be smaller.

In Fig. 10, we find that the time of ConnAlg is decreasing from 0.05 to 0.55, while the ConnAlgI is increasing and decreasing. The obvious reason is that FSW only adopts one strategy while our algorithm combines the DisconnAlg. The calculation of DisconnAlg is more than ConnAlg. So in general, the consuming time is more. In addition, another reason is that the computation amount of ConnAlgI is affected by the number of segments and number of points for each segments. So with kinds of data, the curves of time consume are different.

4.3 The effect of data variation

In this subsection, we will find the effect with variation of the synthetic data sets. For the convenience of parameter tuning, we choose the deviation of the normal distribution and tune the deviation from 0.5 to 2.5 with step 0.5. Fix the mean 0, length 10^6 and the error bound 0.025. The results are showed in Figs. 11 and 12.

In Fig. 11, we can find that the number of segments are increasing with the standard deviation adding for ConnAlgI and ConnAlg. The reason is obvious. When the deviation increases and the error bounds of algorithms are fixed the same, it will influence the area of feasible line to include more points. In Fig. 12, we can see that the processing time do not be influenced more, which is because that the number of segments with kinds of deviation are similar to

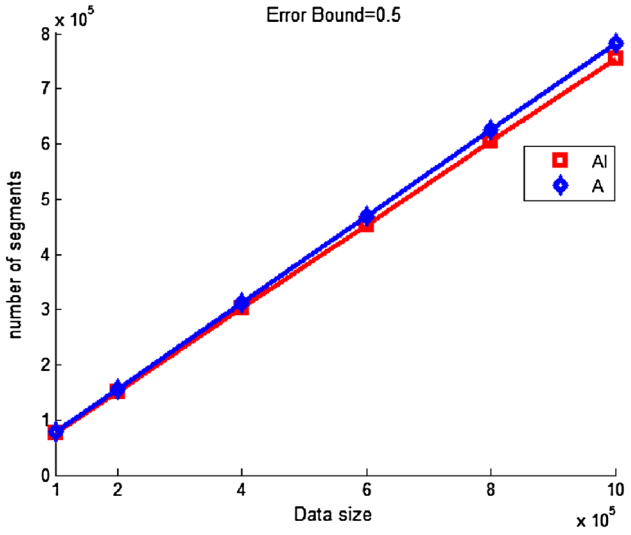

Fig. 13 The effect of data size on number of segments

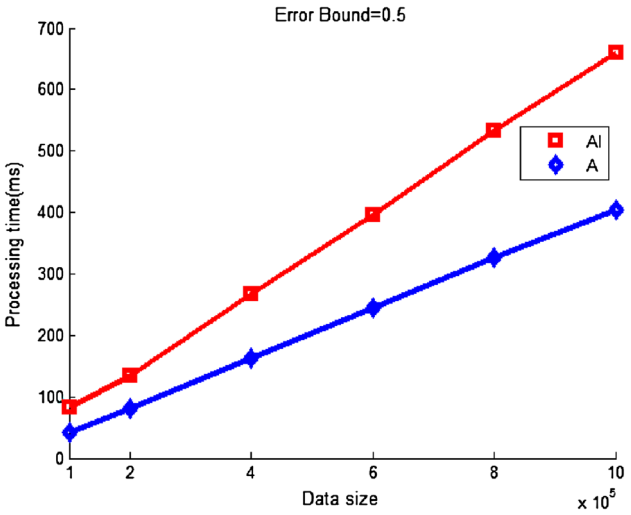

Fig. 14 The effect of data size on processing time

each others. If we give the error bound bigger, the consuming time will be different.

4.4 The length of data set

For comparing the effect of the data length, we gradually add the number of the time series. In this part, we range the length from 10^5 to 10^6 with step 2×10^5, the error bound 0.05, the mean 2, the standard deviation 2. The number of segments and the processing time are showed in Figs. 13 and 14.

From Figs. 13 and 14, we can see the influences of data size are approximate linear on the number and time. This is because both complexity is $O(n)$. With data size increasing, both are rising. In addition, we find that the time of our algorithm is increasing rapidly, when data size is more bigger. This is due to the constructing strategy and the amount of calculation. The more computation causes by computing "convex hulls" [8].

5 Conclusion and future research

The main goals of this paper are to hybrid two strategy to constructing the connected represent lines with error bound guarantee for online data series and to prove the hybrid method is feasible and effective. According to the characteristic of original question and two strategies, we improve our algorithm. By the analysis and comparison, we verify that the algorithm is feasible.

Our scheduled further development in this research topic includes two aspects. Firstly, we try to design a strategy to reduce the computation for processing time. Secondly, by theoretical analysis, we verify the optimal situation for connecting line segments.

Acknowledgments This work was supported by the Science and Technology Key Project of Hebei Academy of Sciences under Grant No. 2014055306 and the cooperation project between Chinese Academy of Sciences and Hebei Academy of Sciences.

References

1. Koski A, Juhola M, Meriste M (1995) Syntactic recognition of ECG signals by attributed finite automata. Pattern Recogn 28(12):1927–1940
2. Lee LC-H, Liu A, Chen W-S (2006) Pattern discovery of fuzzy time series for financial prediction. IEEE Transa Knowl Data Eng 18(5):613–625
3. Sripada SG, Reiter E, Hunter J, Yu J (2003) Segmenting time series for weather forecasting. Applications and innovations in intelligent systems X, pp 193–206
4. Wang X-Z, He Y-L, Wang D-D (2013) Non-Naive Bayesian classifiers for classification problems with continuous attributes. IEEE Trans Cybern. doi:10.1109/TCYB.2013.2245891
5. Appel U, Brandt AV (1983) Adaptive sequential segmentation of piecewise stationary time series. Inf Sci 29(1):27–56
6. Keogh E et al (2004) Segmenting time series: a survey and novel approach. In: Data mining in time series databases, vol 57, pp 1–22
7. Liu X, Lin Z, Wang H (2008) Novel online methods for time series segmentation. IEEE Trans Knowl Data Eng 20(12):1616–1626
8. Xie Q, Pang C, Zhou X, Zhang X, Deng K (2014) Maximum error-bounded piecewise Linear Representation for online stream approximation. VLDB J (accepted)
9. Qi J, Zhang R, Ramamohanarao K, Wang H, Wen Z, Wu D (2013) Indexable online time series segmentation with error bound guarantee. World Wide Web 1–43
10. Keogh E, Zhu Q, Hu B, Hao Y, Xi X, Wei L, Ratanama-hatana CA (2014) The ucr time series classification/clustering homepage. http://www.cs.ucr.edu/~eamonn/time_series_data/
11. Letchford Adrian, Gao Junbin, Zheng Lihong (2013) Filtering financial time series by least squares. Int J Mach Learn Cybern 4(2):149–154

12. Boucheham Bachir (2013) Efficient matching of very complex time series. Int J Mach Learn Cybern 4(5):537–550
13. Keogh EJ, Chu S, Hart D, Pazzani MJ (2001) An online algorithm for segmenting time series. In: ICDM, pp 289–296
14. Fu AC, Chung FL, Ng V, Luk, R (2001) Evolutionary segmentation of financial time series into subsequences. In: Evolutionary computation, pp 426–430
15. Bellman R (1961) On the approximation of curves by line segments using dynamic programming. In: Communication of ACM, p 284
16. Alpanas T, Vlachos M, Keogh EJ, Gunopulos D (2008) Streaming time series summarization using user-defined amnesic functions. IEEE Trans Knowl Data Eng 20(7):992–1006
17. Wang X-Z, Dong L-C, Yan J-H (2012) Maximum ambiguity based sample selection in fuzzy decision tree induction. IEEE Trans Knowl Data Eng 24(8):1491–1505