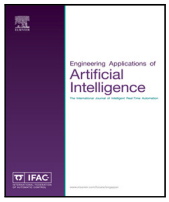
Engineering Applications of Artificial Intelligence 126 (2023) 106892

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



# Adaptive error bounded piecewise linear approximation for time-series representation

Zhou Zhou [a,b,c], Mitra Baratchi [b], Gangquan Si [c,*], Holger H. Hoos [b,d,e], Gang Huang [f]

[a] Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China
[b] Leiden Institute of Advanced Computer Science, Leiden University, The Netherlands
[c] State Key Laboratory of Electrical Insulation and Power Equipment, School of Electrical Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[d] Chair for AI Methodology, RWTH Aachen University, Germany
[e] University of British Columbia, Canada
[f] College of Electrical Engineering, Zhejiang University, Hangzhou, Zhejiang 310027, China

ARTICLE INFO

*Keywords:*
Time series
Data representation
Piecewise linear approximation
Adaptive error bounds

ABSTRACT

Error-bounded piecewise linear approximation ($l_\infty$-PLA) has been proven effective in addressing challenges in data management and analytics. It works by approximating the original time series with linear segments such that the approximation error on each data point does not exceed a pre-defined threshold. To achieve this goal, most prior works on $l_\infty$-PLA use a fixed error threshold during the entire approximation process, assuming a stable time series. However, in many real-world applications, the distribution of values on the time series undergoes substantial changes in different temporal stages. This introduces a need to adaptively tune the error threshold based on the properties of the time series while considering the trade-off between representation error and storage resources. In this work, we propose a general framework for constructing $l_\infty$-PLA with adaptive error bounds (AEPLA). It works by dividing the original time series into a set of intervals and assigns adaptive error bounds to these sub-sequences based on their fluctuation levels. Next, these sub-sequences are approximated using a user-defined $l_\infty$-PLA method. We implement this framework by employing three different types of $l_\infty$-PLA methods and evaluate the performance of our approach on an extensive set of real-world time-series datasets from industrial and scientific domains. Our experiments show that constructing $l_\infty$-PLA using the AEPLA framework can provide better a trade-off between representation error and storage resources and achieves lower time and space costs than applying the original $l_\infty$-PLA methods.

## 1. Introduction

Time-series data is being generated on an unprecedented scale in many application domains, such as power grid monitoring and operations, financial markets, manufacturing, and traffic observation and control. However, the consistent increase in the number of data samples creates challenges for data analytics tasks as well as data management (Fu, 2011). These challenges are twofold: on the one hand, the raw data provides redundant information for AI systems to make decisions; on the other hand, the large volume of data consumes a considerable amount of energy during transmission and/or storage. Therefore, it is often desirable to reduce raw data size to increase storage and processing efficiency. This is usually achieved by designing new approaches to optimise the data representation.

Various techniques are applicable in creating time-series representations. These include Discrete Fourier Transform (DFT) (Agrawal et al., 1993), Discrete Wavelet Transform (DWT) (Popivanov and Miller, 2002), Piecewise Aggregate Approximation (PAA) (Mishra et al., 2022) and Piecewise Linear Approximation (PLA) (Keogh and Smyth, 1997). Among these, PLA is one of the most widely used methods, due to its simplicity. PLA generally works by dividing the original data stream into a set of segments and representing the data within each segment by a linear function. This simple but effective method has been applied to many domains, such as similarity search (Chen et al., 2007), classification (Chen et al., 2018), and forecasting (Chen and Hao, 2020).

The problem of approximating the time series using PLA becomes more challenging when considering streaming time series. In this context, data points are generated in real-time, potentially for an indefinite time. This scenario calls for online PLA methods, which can decide whether to assign the upcoming data to the previous segment or to start a new one. Our paper, like many prior studies of online PLA as surveyed in Lovrić et al. (2014), focuses on the $l_\infty$-norm based PLA method,

* Corresponding author.
*E-mail address:* sigangquan@mail.xjtu.edu.cn (G. Si).

https://doi.org/10.1016/j.engappai.2023.106892
Received 23 October 2022; Received in revised form 29 June 2023; Accepted 27 July 2023
Available online 26 August 2023

referred to as $l_\infty$-PLA. Given a prescribed global error bound $\epsilon_0$, the goal of constructing $l_\infty$-PLA is to approximate the stream with a piecewise linear function, such that the approximation error on each data point does not exceed $\epsilon_0$. Extensive prior work on $l_\infty$-PLA tends to set a fixed value for $\epsilon_0$ during the entire approximation process. It means that the approximation is not adaptive with respect to the characteristics of a given time series. The focus of this paper, on the other hand, is time-series approximation with unfixed error thresholds, which allows creating a multi-resolution representation of the time series (Zhan et al., 2020) in different application fields. For instance, Si and Yin (2013) proposed a multi-resolution PLA method for the technical analysis that detects specific patterns in financial time series (e.g., the head-and-shoulder pattern). In this approach the represented time series with higher and lower error thresholds are used for longer- and shorter-duration technical pattern searching, respectively. Liu et al. (2016) proposed an online compression framework for streaming trajectories of moving entities such as cars with different error tolerances. In their framework, older trajectories will be compressed with larger error bounds to reflect the decay of the importance of the data over time. Hu et al. (2020) proposed a multi-resolution representation method for the analysis of Internet-of-Things sensor data. Different error bounds are applied to meet the diverse needs of users, e.g., a higher error bound is given with respect to a stricter fitting error limitation. Zhan et al. (2021) proposed a multi-resolution piecewise representation method for time-series anomaly detection, where the segmented original time series is projected into the feature space with different error bounds. Furthermore, Luo et al. (2021) presented an efficient time-series retrieval method also based on a multi-resolution representation. Deng and Li (2022) proposed a PLA method with optimised fitting errors to remove the noise and fluctuations in hydraulic fracturing time series. The previously mentioned for multi-resolution time-series representation are all based on specific PLA methods or designed for specific applications. In other words, none of these works has provided a general and systematic framework that can be used for different applications with user-specified $l_\infty$-PLA methods.

To bridge the gap and provide a more flexible $l_\infty$-PLA-based representation, we revisit the determination of $\epsilon_0$ by considering the objective of setting an error bound. Generally, the constraint on the approximation error of each data point is not extremely strict in many real-world tasks. In those cases, it is preferable to adjust the error bound in the approximation process based on the characteristics of the given time series. Specifically, a relatively small error bound should be used when rapid changes occur within a period to retain more features and to reduce the local representation error. In contrast, a larger error bound is suitable for periods when the data stream is stable to increase the compression ratio at the cost of a tolerable increase of representation error. In this manner, we can achieve a better trade-off between precision and compression.

In this paper, we characterise different classes of $l_\infty$-PLA functions and propose a general framework for $l_\infty$-PLA with adaptive error bounds (AEPLA). This framework aims at adjusting the error bound in the approximation process, depending on the stability of the time series at different temporal stages. Compared to using a fixed $\epsilon_0$, the proposed approach can better preserve the key features of a given time series with limited storage resources. More specifically, our contributions in this paper can be summarised as follows:

- We introduce a general framework for $l_\infty$-PLA with adaptive error bounds that can be used in combination with different user-defined or user-selected $l_\infty$-PLA methods. This framework can provide better trade-offs between representation error and storage resources compared to applying the original $l_\infty$-PLA methods with a fixed error bound.
- We implement AEPLA based on different types of $l_\infty$-PLA methods to demonstrate how different PLA methods can integrate within this framework. We evaluate different aspects of its approximation performance on a broad range of real-world time-series datasets from industrial and scientific domains.
- We further improve this framework in an offline mode by proposing an approach for automatically tuning its input parameters (in this case, a scaling factor) using automated algorithm configuration methods such that its performance cannot be affected by ad-hoc user-defined parameters.

The rest of this study is organised as follows. Section 2 summarises prior works on piecewise linear approximation. Section 3 briefly introduces the error criteria used in quantifying the approximation quality and the concept of error-bound PLA. Section 4 provides the formal definition of the problem and a detailed description of our new framework. Details on our empirical evaluation and the results we obtained from it are presented in Section 5. Finally, we draw some general conclusions and provide a brief discussion of future work in Section 6.

## 2. Related work

Time-series representation by PLA has been studied in many different contexts; hence, many methods have been proposed earlier. Generally, the proposed methods can be divided into two classes: offline and online (Lovrić et al., 2014).

**Offline PLA**: The offline versions of PLA have to work on time series with finite length, meaning that the available data points should remain unchanged during the whole approximation process. Dynamic Programming (DP) can be used to find the optimum solution for offline time-series segmentation (see e.g., Pang et al. (2022) and Wu et al. (2021)) Given the maximum number of segments and the error tolerance of each segment, the goal of the DP approach is to find the optimal segmentation solution with minimal overall error. Carmona-Poyato et al. (2020) proposed an optimal offline time-series segmentation method based on the A* algorithm (a classic graph traversal algorithm Salotti, 2002). Later, they proposed another optimal offline solution (Carmona-Poyato et al., 2021) based on feasible space, which is an area where any straight line within can approximate the data points in a segment with a given error bound (Liu et al., 2008). However, the high computational complexity of these optimal methods motivates the development of heuristic methods that produce sub-optimal results in most cases. Top-down (Ji et al., 2016) and Bottom-up (Keogh et al., 2004) approaches are two typical heuristic solutions for time-series segmentation. The top-down approach considers the whole time series as one segment at first and splits it attractively. The Bottom-up approach starts with small segments of equal length and merges them to attain the final results. Besides, Evolutionary Computation algorithms (Durán-Rosal et al., 2019), Bayesian inference (Oliver et al., 1998), and Hidden Markov Models (HMM) (Mori et al., 2005) have also been applied to offline time-series segmentation. Generally, the offline versions lack the ability to process real-time data since the length of the time series extends over time.

**Online PLA:** The online versions of PLA work on infinite-length time series. Online PLA methods use greedy strategies to determine whether the upcoming data point belongs to the previous segment or a new one. Hence they are more suitable for streaming time-series segmentation. Various online PLA methods can be categorised based on the error metric used for evaluation of the approximation error, that is, $l_p$ norm ($p = 1, 2, \infty$). The sliding window (Keogh et al., 2004) is a classical method defined based on *l1/l2* norm (*i.e.*, Manhattan and Euclidean distance, respectively). This method works by accumulating each new data point into the current segment as long as the total representation error does not exceed a predefined error bound. SWAB (Keogh et al., 2004) which combines SW and Bottom-up, is also widely used in this context. However, setting a proper error budget is never an easy task without knowledge about the whole time series. On the other hand, constructing PLA based on the $l_\infty$ norm, also defined as error-bounded PLA ($l_\infty$-PLA), puts an error bound on each data point. The $l_\infty$-PLA methods can be categorised according to the form of piecewise function generated, that is, continuous, disjoint, mixed type,

**Table 1**
Overview of important works on offline and online PLA.

| Authors | Years | Fundamental algorithm | Online | Offline | Characteristic |
|---|---|---|---|---|---|
| Pang et al. (2022) | 2022 | Dynamic programming | | ✓ | Optimal |
| Wu et al. (2021) | 2021 | Dynamic programming | | ✓ | Optimal |
| Carmona-Poyato et al. (2021) | 2021 | Feasible space algorithm (Liu et al., 2008) | | ✓ | Optimal |
| Carmona-Poyato et al. (2020) | 2020 | A* algorithm | | ✓ | Optimal |
| Ji et al. (2016) | 2016 | Top-down | | ✓ | Heuristic |
| Keogh et al. (2004) | 2004 | Bottom-up | | ✓ | Heuristic |
| Hu et al. (2018) | 2018 | Turning points identification | ✓ | | Continuous |
| Zhao et al. (2016) | 2016 | Feasible space algorithm (Liu et al., 2008) and OptimalPLR (Xie et al., 2014) | ✓ | | Continuous |
| Elmeleegy et al. (2009) | 2009 | Swing Filter | ✓ | | Continuous |
| Liu et al. (2008) | 2008 | Feasible space windows | ✓ | | Continuous |
| Lin et al. (2020) | 2020 | Swing Filter | ✓ | | Disjoint |
| Xie et al. (2014) | 2014 | Convex hull-based | ✓ | | Disjoint |
| O'Rourke (1981) | 1981 | Polygon-based | ✓ | | Disjoint |
| Zhao et al. (2020) | 2020 | OptimalPLR (Xie et al., 2014) | ✓ | | Mixed-type |
| Luo et al. (2015) | 2015 | Extended polygon-based | ✓ | | Mixed-type |
| Zhao et al. (2022) | 2022 | OptimalPLR (Xie et al., 2014) | ✓ | | Semi-connected |
| Hakimi and Schmeichel (1991) | 1991 | Polygon-based | ✓ | | Semi-connected |

and semi-connected. We call it continuous $l_\infty$-PLA, if it is completely comprised of connected linear representations; otherwise, it is referred to as disjoint $l_\infty$-PLA. Mixed-type $l_\infty$-PLA consists of both connected and disconnected linear representations. Finally, semi-connected $l_\infty$-PLA allows the connection point of two adjacent segments to be located within two timestamps (Zhao et al., 2020).

For the continuous case, Elmeleegy et al. (2009) reinvented Swing Filter based on the idea from Gritzali and Papakonstantinou (1983). Liu et al. (2008) also proposed an improved version of that in Gritzali and Papakonstantinou (1983) named Feasible Space Window (FSW) which introduced the idea of feasible space to search for the farthest endpoint of a segment. Hu et al. (2018) proposed CSMR_TP, which uses turning points to refine the result of FSW. Zhao et al. (2016) proposed ConnSegAlg, which uses a backwards-checking strategy to minimise the number of segments. For the disjoint case, O'Rourke (1981) proposed a linear-time optimal method ParaOptimal for PLA under $l_\infty$ norm. Xie et al. (2014) proposed OptimalPLR, which is theoretically equivalent to ParaOptimal. Lin et al. (2020) proposed Swing-RR to further reduce the space cost by resolution reduction. For the mixed-type case, Luo et al. (2015) constructed an extended polygon to fit a mixed-type PLA. Zhao et al. (2020) proposed SemiMixedAlg, which employs a series of theoretically proven deductions as shortcuts to DP-based computing. It achieves the same results in terms of the compression ratio as in Luo et al. (2015), but with lower running time and memory costs. For the semi-connected case, Hakimi and Schmeichel (1991) first proposed a pipeline-based method for constructing optimal semi-connected segments. Different from the pipeline-based method, Zhao et al. (2022) proposed a linear-time semi-connected PLA method, which constructs the optimal number of semi-connected segments by progressively revising the segmentation given by Xie et al. (2014); this has been proven to achieve state-of-the-art performance in terms of time and memory complexity. A number of important works on offline and online PLA are summarised in Table 1.

Despite the various $l_\infty$-PLA methods that have been proposed for online time-series approximation, applying these methods directly is ill-suited for the problem we considered, *i.e.*, constructing PLA with data storage resource limits. In this case, we prefer to reduce the overall approximation error as much as possible with the same amount of storage resources or under the same value of compression ratio. One solution is to adjust the value of the error bound during the approximation process. In a recent study (Hu et al., 2018), the authors proposed a method for $l_\infty$-PLA with different error bounds by refining the segmentation results given by Liu et al. (2008). Inherently, this approach is equivalent to only decreasing the error bound during the periods with rapid changes in the time series. Although this can result in a lower approximation error, it comes at the cost of a lower compression ratio. In contrast, the framework we propose in this paper is able to better adapt the error bound in the approximation process, respecting the characteristics of time series. In this manner, it can provide a lower overall approximation error than applying the original $l_\infty$-PLA method without increasing the amount of data storage resources used. Besides, the new framework is not tied to a specific PLA algorithm and can be combined with any user-defined PLA method.

## 3. Preliminaries

Before introducing our new framework, we provide a number of basic definitions and principles for piecewise linear approximation. Table 2 summarises the notations used in this paper, and Fig. 1, visualises them.

### 3.1. Error criteria

Given a time series $S = (y_1, y_2, \dots, y_n)$, the quality of its approximation $S' = (y'_1, y'_2, \dots, y'_n)$ can be measured based on different error forms. The most commonly used criterion is $l_p$-error, i.e.,

$$\left(\sum_{i=1}^{n} |y'_i - y_i|^p\right)^{\frac{1}{p}}. \tag{1}$$

The widely used $l_1$-norm and $l_2$-norms are also called Manhattan and Euclidean distance, respectively. These norms essentially represent the sum of errors over the entire time series, which renders them unsuitable for online PLA. In other words, it is impossible to define a proper error budget without information on the upcoming data. In light of this, a better choice in this context is the $l_\infty$-norm, defined as

$$\lim_{p\to\infty}\left(\sum_{i=1}^{n} |y'_i - y_i|^p\right)^{\frac{1}{p}} = \max_i |y'_i - y_i|, \tag{2}$$

which gives the maximal error bound on all data records.

### 3.2. Error-bounded piecewise linear approximation

Given time series $S$, the idea of error-bounded PLA is to split $S$ into a set of segments and then fit a polynomial model for each segment, such that the approximation error on each data point does not exceed a predefined error threshold $\epsilon_0$.

Formally, $S = (y_1, y_2, \dots, y_n)$ can be seen as a polygonal line $\Gamma$, which is indicated by the dashed line connecting $y_1, y_2, \dots, y_n$ in Fig. 1. Then, the approximation is equivalent to fitting a set of linear functions $f_1, f_2, \dots, f_k$ to $\Gamma$, such that $\forall i = 1...n$, $|y_i - y'_i| \le \epsilon_0$, and the number of segments $k$ is minimised. Considering the different forms of piecewise functions, all $l_\infty$-PLA methods can be categorised as discussed in Section 2. Fig. 2 illustrates the approximation results obtained

**Table 2**
An overview of the notations used in the article.

| Symbol | Meaning |
|---|---|
| $S$ | A time series $S = (y_1, y_2, \dots, y_n)$ of length $n$, where $y_i$ is the value at the timestamp $i$ |
| $S'$ | The representation of each data point of $S$, i.e., $S' = (y'_1, y'_2, \dots, y'_n)$ |
| $\epsilon_0$ | The global error bound on each data point |
| $\epsilon_m$ | The local error bound on each data point |
| $\Gamma$ | A polygonal line connecting $(y_1, y_2, \dots, y_n)$ |
| $f_m$ | The linear function to represent the $m$th segment of $S$ |
| $buf$ | A time-series interval |
| $\alpha$ | A scaling factor |
| $lmax$ | The maximum value in $buf$ |
| $lmin$ | The minimum value in $buf$ |
| $gmax$ | The maximum value in $S$ |
| $gmin$ | The minimum value in $S$ |
| $lr$ | The local range $lr = lmax - lmin$ |
| $gr$ | The global range $gr = gmax - gmin$ |
| $\beta$ | The fluctuation coefficient $\beta = lr/gr$ |
| $p$ | An error percentage satisfying $\epsilon_0 = p \cdot gr$ |

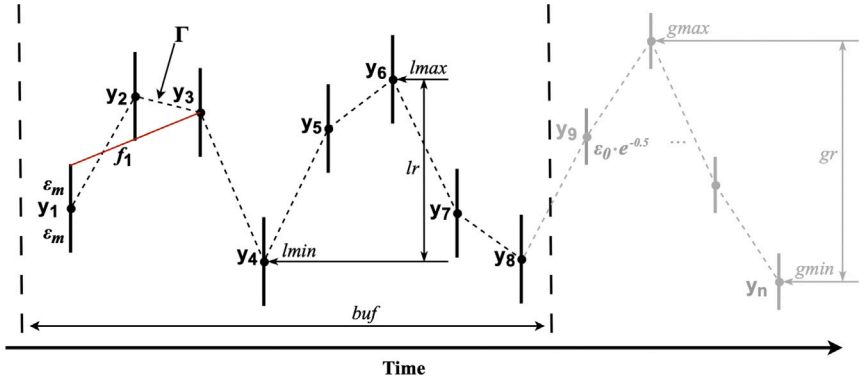



**Fig. 1.** An example time series for the illustration of the notations used in this article.

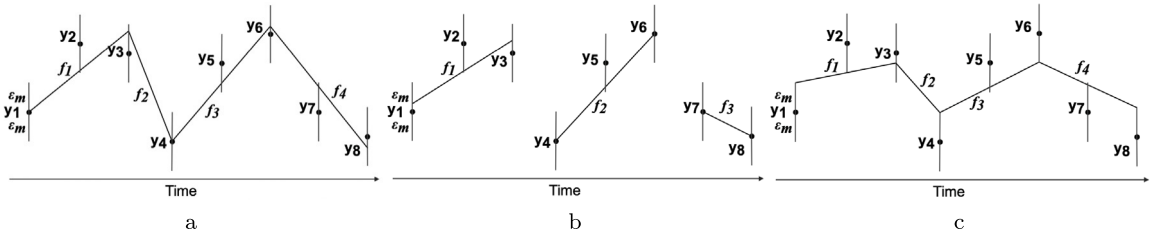



**Fig. 2.** Approximations of the time series within the interval in Fig. 1 using three different piecewise functions, i.e., (a) FSW as continuous PLA, (b) OptimalPLR as disjoint PLA, and (c) SemiOpt as semi-connected PLA.

from different types of $l_\infty$-PLA methods, *i.e.*, continuous, disjoint and semi-connected. Herein, we applied PLA, OptimalPLR, and SemiOpt, respectively, to approximate the time series within the interval as shown in Fig. 1.

## 4. PLA with adaptive error bounds

Given a time series $S$ and the global error bound $\epsilon_0$, the idea of constructing $l_\infty$-PLA with adaptive error bounds is equivalent to fitting a set of linear functions to $S$, where the error threshold for each approximation function is determined based on $\epsilon_0$ and the fluctuation level of $S$ at different temporal stages.

Formally, $S$ will be divided into a set of consecutive segments, represented by $k$ linear functions $f_1, f_2, \dots, f_k$, such that $\forall i = 1...n : |y_i - y'_i| \le \epsilon_m$, where $y_i$ is supposed to be within the temporal range of $f_m$. The relation between $\epsilon_0$ and the error threshold $\epsilon_m$ for $f_m$ will be discussed in Section 4.3. The connection between two adjacent functions $f_{j-1}$ and $f_j$ can be based on any one of the three types illustrated in Fig. 2.

### 4.1. Framework overview

In this section, a new framework to construct $l_\infty$-PLA with adaptive error bounds (AEPLA) is proposed. The new framework allocates different error bounds based on the stability of a given time series at different stages. First, the original time series is divided into consecutive intervals. The fluctuation level of data within each interval is evaluated by comparing its local range with the global range of the time series (to be defined later). The error bound is dynamically adjusted based on this comparison. Next, each interval can be approximated by means of user-defined $l_\infty$-PLA methods with these dynamically adjusted error bounds to obtain the final, flexible approximation. The workflow of AEPLA is shown in Fig. 3.

### 4.2. Determining the intervals

Before the allocation of dynamic error bounds, we need to evaluate the stability of the original time series at different temporal stages. In AEPLA, this evaluation will be performed for consecutive intervals. The

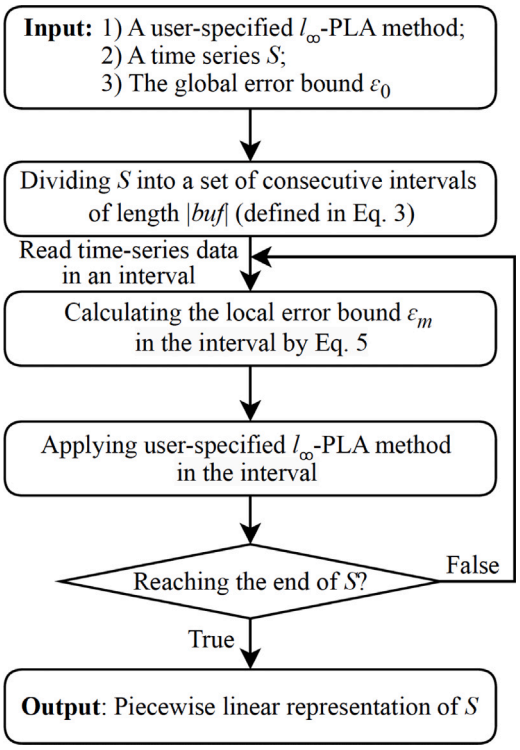



**Fig. 3.** The workflow of AEPLA.

first step is to determine the proper length of these intervals. If the length is allowed to be set arbitrarily large, it will be equivalent to evaluating the stability of the entire time series; on the other hand, an interval that is too small will result in overly fragmented approximations. In this step, a series of data points from a given interval are read into a buffer (*buf*), representing intervals of variable length. As discussed above, *buf* should be long enough to include at least one segment and also be restricted from evaluating the stability of the entire data stream. In AEPLA, this balance is struck by considering the length of time series and $\epsilon_0$, that is,

$$|buf| = \alpha \cdot n \cdot \epsilon_0 \tag{3}$$

where $|\cdot|$ indicates the length, $n$ is the length of the given time series, and $\alpha$ is a scaling factor between $2/(n \cdot \epsilon_0)$ and 1. The lower bound of this variable ensures that the minimum number of data points in the buffer will be 2, which is sufficient for forming at least one line segment. The upper bound ensures that the size of *buf* will not exceed $n$. The rationale behind the formula is that more data points are allowed to be included in a segment with a lower error bound (*i.e.*, larger $\epsilon_0$). Therefore, the size of *buf* should be increased in this case. The parameter $\alpha$ in Eq. (3) also affects the length of *buf*. The choice of this scaling factor can be performed in an automated and data-driven manner. Optimisation of $\alpha$ will be discussed in the Experiments Section (see Section 5.6). The original time series is read into *buf* in chronological order for evaluation, which will be discussed in Section 4.3.

### 4.3. Tuning the error bound

Next, the dynamic error bound $\epsilon_m$ is assigned to the intervals based on the degree of stability of data contained in *buf*. In our new framework, the fluctuation of the sub-sequence contained in *buf* is evaluated in terms of its local range ($lr$) value. The local range is defined as $lr = lmax - lmin$, where *lmax* and *lmin* are the maximum and minimum values of the data in *buf*, respectively. Additionally, the global range ($gr$) is defined as $gr = gmax - gmin$, where *gmax* and *gmin* are the maximum and minimum values of the whole time series, respectively. The value of $gr$ is updated over time as new data arrives. The stability of the original time series within a specific period is indicated by a fluctuation coefficient $\beta = lr/gr$, which is essential for determining the error bound. Using $\beta$, the dynamic error bound for each interval is defined as

$$\epsilon_m = \epsilon_0 \cdot e^{(0.5-\beta)}, \tag{4}$$

where $\epsilon_0$ is a user-defined global error bound.

In general, we would like $\epsilon_m$ to be adjusted based on $\epsilon_0$. Furthermore, we would like $\epsilon_m$ to be sensitive to the level of stability of the time series within a given interval, as captured by $\beta$. For decreasing values of $\beta$, we would like $\epsilon_m$ to increase. To achieve this, in Eq. (4), $\epsilon_0$ is multiplied by an exponential function to provide a nonlinear and monotonic increase. Considering that the range of $\beta$ varies between 0 and 1, the exponent is re-scaled by adding a constant 0.5, such that the local error bound $\epsilon_m$ is allowed to be adjusted around the value of $\epsilon_0$. Moreover, the value of the exponent is adjusted to be symmetric, indicating an unbiased adjustment of the error bound regarding the change of $\beta$.

In the literature, the value of $\epsilon_0$ is usually determined based on the Maximum Error Percentage (MEP) (Liu et al., 2008; Xie et al., 2014). Using MEP, for a given error percentage $p$, $\epsilon_0$ is defined as $\epsilon_0 = p \cdot gr$. As a result, Eq. (4) can be rewritten as

$$\epsilon_m = \epsilon_0 \cdot e^{(0.5-\beta)} = p \cdot gr \cdot e^{(0.5-\beta)}. \tag{5}$$

Based on Eq. (5), a large value of $\beta$, corresponding to large fluctuations in the given interval, results in a relatively small $\epsilon_m$. On the other hand, large $\epsilon_m$ will be assigned to more stable intervals characterised by small values of $\beta$. Compared to having a fixed error bound, calculating adaptive error bounds using Eq. (5) makes it possible to better approximate fluctuating time series using limited storage. While the original value of Eq. (5) still needs to be defined by the user, the goal is to provide a more reasonable estimate that is not overly sensitive to the original $\epsilon_0$.

### 4.4. Applying PLA methods

After the allocation of dynamic error bounds, the original time series can be approximated using user-defined or user-selected $l_\infty$-PLA methods for each interval. Generally, a specific $l_\infty$-PLA method is applied to the data in *buf*, and the linear approximations for each segment are generated in chronological order. Next, *buf* is updated with the last segment of the previous interval and the additional new data from the time series. The idea is that the last segment generated within the current *buf* is generally incomplete. As these data are read into *buf* again, we could refine the approximation of data close to the breakpoints of two intervals. The process of applying $l_\infty$-PLA methods (mentioned in Section 3.2) to *buf*, generating linear approximations and updating *buf* is repeated as long as data is streaming.

To summarise, AEPLA consists of the following three steps: (i) dividing the original time series into a sequence of intervals; (ii) assigning dynamic error bounds to each interval, based on the degree of fluctuation within it; and (iii) generating a piecewise linear representation of the original time series within each interval using $l_\infty$-PLA methods. The process is described in the form of pseudo-code in Algorithm 1.

### 4.5. Properties

In this section, we further discuss the properties of the proposed framework. For simplicity, hereinafter, we denote the original $l_\infty$-PLA methods with $M$ and the corresponding adaptive version obtained using our new framework with $M^+$, where $M$ is a specific PLA method (e.g., FSW). First, as $M^+$ is built based on $M$, we prove that the performance of $M^+$ in terms of representation quality is also bounded by that

**Algorithm 1** AEPLA: PLA with adaptive error bounds.

**Input**: $l_\infty$-PLA_method (specified by user);
$S$: time series of length $n$;
$\epsilon_0$: global error bound;
$\alpha$: scaling factor (default = 0.2);
**Output**: piecewise linear representation of $S$ with adaptive error bounds.

1: $interval\_start \leftarrow 0$;
2: $segment\_total \leftarrow []$;
3: **while** $interval\_start < n$ **do**
4: $gr \leftarrow gmax - gmin$;
5: bufferLength $\leftarrow \alpha \cdot n \cdot \epsilon_0$;
6: $buf \leftarrow$ S[$interval\_start$:($interval\_start$+bufferLength)];
7: $lr \leftarrow lmax - lmin$;
8: $\beta \leftarrow lr/gr$;
9: $\epsilon_m \leftarrow \epsilon_0 \cdot e^{(0.5-\beta)}$;
10: $seg \leftarrow l_\infty$-PLA_method($buf, \epsilon_m$);
11: $segment\_total$.append($seg$[1:(|$seg$| -1)]); \\ |$seg$| denotes the length of $seg$
12: $interval\_start \leftarrow$ original index of first data point of $seg[|seg|]$ in $S$;
13: $buf \leftarrow []$;
14: **end while**
15: **return** $segment\_total$;

of $M$, and that $M^+$ will be degraded to $M$ under a special circumstance. Then, we show that performing $M$ within the new framework will not increase the time complexity, i.e., $M^+$ is also a linear-time algorithm as $M$, which can be applied for online time-series segmentation.

**Property 4.1.** *For given $\epsilon_0$, let $r^+$ be the number of segments generated by $M^+$. Let $r_{max}$ be the number of segments for $M$ with fixed error bound $\epsilon_0 \cdot e^{-0.5}$. Also, let $r_{min}$ be the number of segments for $M$ with fixed error bound $\epsilon_0 \cdot e^{0.5}$. We can prove that $r_{min} \leq r^+ \leq r_{max}$.*

**Proof.** This property can be proven by considering two extreme cases. First, suppose that the given time series $S$ is a sawtooth function, such that the local range $lr$ equals the global range $gr$ (i.e., $\beta = 1$) within any interval of $S$, $M^+$ is equivalent to applying $M$ with fixed $\epsilon_0 \cdot e^{-0.5}$. Thus the maximum number of segments generated by $M^+$ is bounded by $r_{max}$. On the other hand, suppose that $S$ is similar to a horizontal line, such that the local range $lr$ equals zero (i.e., $\beta = 0$) within any interval of $S$, $M^+$ is equivalent to applying $M$ with fixed $\epsilon_0 \cdot e^{0.5}$, thus the minimum number of segments generated by $M^+$ is bounded by $r_{min}$. Since the number of segments generated by typical $l_\infty$-PLA methods decreases monotonically with respect to the increase of $\epsilon_m$ (Xie et al., 2014), we conclude that $r_{min} \leq r^+ \leq r_{max}$.

**Property 4.2.** *Given a time series $S$ and an $\epsilon_0$, $M^+$ is degraded to $M$, if we set all the local error bounds $\epsilon_m$ to $\epsilon_0$.*

**Proof.** Clearly, if we replace the procedure in line 9 of Algorithm 1 by $\epsilon_m \leftarrow \epsilon_0$, $M^+$ is equivalent to performing PLA within consecutive intervals using the original $l_\infty$-PLA $M$ and a fixed error bound $\epsilon_0$. Thus, the overall approximations given by $M^+$ and $M$ are the same in this case.

**Property 4.3.** *Given a time series $S$ of length $n$, the computational complexity of applying $l_\infty$-PLA using AEPLA is in the order of the original $l_\infty$-PLA method.*

**Proof.** Specifically, let $O_{original}$ be the time complexity of any original $l_\infty$-PLA method, $l$ the length of the buffer $buf$, and $k$ the total number of times the buffer is filled (i.e., $n = l \cdot k$). In this case, the computational complexity of Algorithm 1 is primarily determined by: (i) the time complexity for searching $gr$, $O(n)$; (ii) the time complexity for searching $lr$, $O(l)$; (iii) the time complexity for running the original method on $buf$, $\frac{l}{n} \cdot O_{original}$. The overall time complexity is $O(n) + k \cdot O(l) + k \cdot \frac{l}{n} \cdot O_{original} = O(n) + O_{original}$. Given the time complexity of the widely used $l_\infty$-PLA methods ( Table 3), we can see that $O_{original}$ is greater than or equal to $O(n)$ in general. Hence the asymptotic time complexity of AEPLA is the same as $O_{original}$, with relatively minor differences in the constants.

**Table 3**
Time complexity of widely used $l_\infty$-PLA methods. $M$ and $P$ are the number of segments and the average length of segments, respectively.

| Method | Time complexity |
|---|---|
| SW (Keogh et al., 2004) | $O(P \cdot n)$ |
| SWAB (Keogh et al., 2004) | $O(P \cdot n)$ |
| FSW (Liu et al., 2008) | $O(M \cdot n)$ |
| ConnSegAlg (Zhao et al., 2016) | $O(n)$ |
| ParaOptimal (O'Rourke, 1981) | $O(n)$ |
| OptimalPLR (Xie et al., 2014) | $O(n)$ |

## 5. Empirical analysis

In this section, we present a set of experiments to evaluate the performance of our proposed framework from different perspectives. Three types of $l_\infty$-PLA approaches (disjoint, continuous, and semi-connected), as illustrated in Fig. 2, are tested under the AEPLA framework (detailed in Section 5). We compare the results of using these methods within our framework against the results achieved by the original methods. The code to reproduce the experiments is available in a repository.[1]

Our experiments are designed with the goal of answering the following research questions:

- RQ1: What is the advantage obtained by the new AEPLA framework in terms of time-series approximation quality? Since both the original ($M$) and the adaptive version achieved by AEPLA ($M^+$) depend on a user-defined value of $\epsilon_0$, we compare the approximation quality achieved by each method using the same user-defined values of $\epsilon_0$.
- RQ2: What is the advantage obtained by the new AEPLA framework in time-series compression? As the time-series approximation can be seen as a compression task, we carry out another comparison of $M$ and $M^+$ to investigate how each method performs when limited to using the same memory space.
- RQ3: What are the time and space costs of the new AEPLA framework? In Section 4.5, we proved that the performance of $M^+$ is bounded by $M$. Here, we compare the actual running time and space cost of $M$ and $M^+$ to affirm this claim.
- RQ4: How does the choice of parameters impact the AEPLA method? Since in our framework, we introduce a scaling factor $\alpha$, we analyse the extent to which the choice of this parameter affects the performance of AEPLA. We further discuss an optional, offline optimisation process for automatically tuning this parameter.

### 5.1. Experiment setup

In this section, we provide the details of our experimental setup to address the previous research questions.

**Benchmark datasets:** For our evaluations, we require raw (unlabelled) and relatively long time-series data with different characteristics (*e.g.*, being stationary/non-stationary, cyclical/non-cyclical). We have chosen 84 datasets covering various application areas, including

[1] https://github.com/zhourongleiden/AEPLA.

**Table 4**
Details of the benchmark datasets.

| Datasets | Sources | Application areas |
|---|---|---|
| A1–A20 | CompEngine | Astrophysics |
| B1–B20 | CompEngine | Finance |
| C1–C20 | CompEngine | Meteorology |
| D1–D20 | CompEngine | Finance |
| CinC_ECG_torso | UCR | Physiology |
| InlineSkate | UCR | Kinesiology |
| MALLAT | UCR | Signal processing |
| StarLightCurves | UCR | Astrophysics |

physics, finance, and meteorology from CompEngine.[2] and the UCR archive.[3] (see the complete list in Appendix). CompEngine is a self-organising library of time-series data, originally designed to allow the collection of a large set of time-series data from diverse industrial and scientific domains. Following previous work (e.g., Xie et al. (2014) and Zhao et al. (2016, 2020)), datasets chosen from the UCR archive are concatenated to simulate long-streaming data (10–100 MB in size). We note that only 4 relatively longer datasets from the entire UCR database are utilised, as the primary goal here is to demonstrate the applicability of our method to large volumes of data. Table 4 summarises the details of the benchmark datasets.

**Baseline PLA methods:** We test our proposed framework on three types of error-bounded PLA methods, namely (i) continuous PLA, (ii) disjoint PLA, and (iii) semi-connected PLA.

- **FSW** (Liu et al., 2008): We have also chosen FSW as a heuristic method within the category of continuous PLA since no method in this category can guarantee optimal approximation in linear time (Zhao et al., 2020). However, FSW was chosen as an example within this category because it can provide a near-optimal solution (Xu et al., 2012) and is easy to implement.
- **OptimalPLR** (Xie et al., 2014): We have chosen OptimalPLR within the category of disjoint PLA methods since it can provide an optimal approximation of the time series in terms of the number of segments in linear time.
- **SemiOpt** (Zhao et al., 2022): We have chosen SemiOpt within the category of semi-connected PLA methods, as it is proven to achieve state-of-the-art performance in terms of compression ratios in linear time.

All of the methods were implemented in Python, based on the pseudo-code provided by the authors. We note that other $l_\infty$-PLA methods, such as the ones discussed in Section 2, can also be easily used within our new framework. Besides, the notion of the adaptive error bounds can also be integrated with any compressor (e.g., SZ Zhao et al., 2021 and ZFP Lindstrom, 2014) as long as there is an error bound that should be determined by users in the data reduction or approximation process. However, we will not further explore the idea in this paper.

**Performance metrics:** We evaluate the performance of our proposed framework from two different perspectives. On the one hand, we compare the approximation of the baseline and adaptive PLA algorithms under the same external conditions imposed by how the parameter $\epsilon_0$ is selected by users. For this purpose, we consider the representation error per segment (RE/S), which represents the approximation quality achieved for a fixed global error bound $\epsilon_0$. We evaluate the performance of PLA methods in terms of RE/S as a function of different $\epsilon_0$ values. On the other hand, we are also interested in the trade-off between the approximation error and the number of resulting segments, both of which depend on the global error bound, $\epsilon_0$. To quantitatively assess the trade-off between these two competing objectives, we chose the hypervolume indicator (HI) (Zhang et al., 2022), a metric commonly used in the context of evaluating multi-objective optimisation algorithms. More specifically, the details of these performance metrics are as follows:

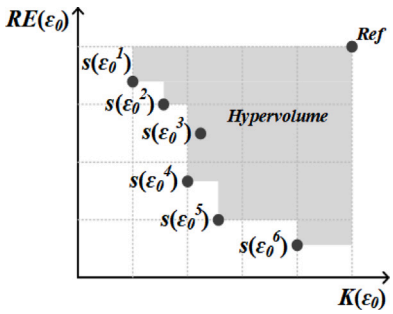



**Fig. 4.** An illustration of the hypervolume indicator (two objectives).

- Representation error (RE): The root-mean-square error (RMSE) between each $y_i$ and its approximation $y_i'$.
- Representation error per segment (RE/S): The representation error divided by the number of segments $K$, where $n$ is the total length of the original time series, i.e.,

$$RE/S = \frac{1}{K} \cdot \left( \frac{1}{n} \sum_{i=1}^{n} |y_i' - y_i|^2 \right)^{\frac{1}{2}}. \quad (6)$$

- Hypervolume indicator (HI): A mapping of the set of trade-off solutions (considering two or more objectives) given by an optimiser into a unary value. (Please refer to Emmerich and Deutz (2018) for a more comprehensive definition.) Fig. 4 provides an illustration of this indicator. Here, $RE(\epsilon_0)$ and $K(\epsilon_0)$ represent the two metrics that we wish to minimise (i.e., the representation error and the number of segments) as functions of $\epsilon_0$. Furthermore, $s(\epsilon_0^i)(i = 1...6)$ represents the set of solutions provided by a PLA method (either the original PLA method, $M$, or the corresponding adaptive version, $M^+$). Each solution is produced using a different value $\epsilon_0^i$ for the $\epsilon_0$ parameter. The point *Ref* denoted in the figure is the so-called reference point used to calculate the hypervolume; here, the area marked in grey defines the value of the hypervolume indicator. A higher hypervolume generally denotes better performance. Thus, when comparing two PLA methods using this indicator, a higher value denotes better overall approximation performance irrespective of how the $\epsilon_0$ parameter is set by the user. We note that, in the figure, all solutions except for $s(\epsilon_0^3)$ are non-dominated points (Ehrgott, 2012) and thus contribute to the calculation of HI (when dealing with two optimisation objectives, a solution is non-dominated when neither of its two objective values can be improved without degrading the other).

**Initial parameter setting:** In Section 4.2, we mentioned that to use our framework, the scaling factor $\alpha$ needs to be initialised to a default value; based on preliminary experiments on all datasets, we have set this default value to $\alpha = 0.2$. For the two baseline methods used in our experiments, the error bound is determined as $\epsilon_0 = p \cdot gr$ ($p$ ranges from 0.03 to 1) using MEP (Liu et al., 2008; Xie et al., 2014).

### 5.2. Experiments on continuous PLA

In the first set of experiments, we address RQ1 by applying our proposed framework to FSW and denote the resulting method by FSW$^+$. These experiments are essentially testing the applicability of our approach to continuous $l_\infty$-PLA methods.

Figs. 5 to 8 show the results of RE/S using different values of $\epsilon_0$ for 80 datasets divided into 4 categories, respectively. Due to the broadly

[2] https://www.comp-engine.org/.
[3] https://www.cs.ucr.edu/~eamonn/time_series_data/.

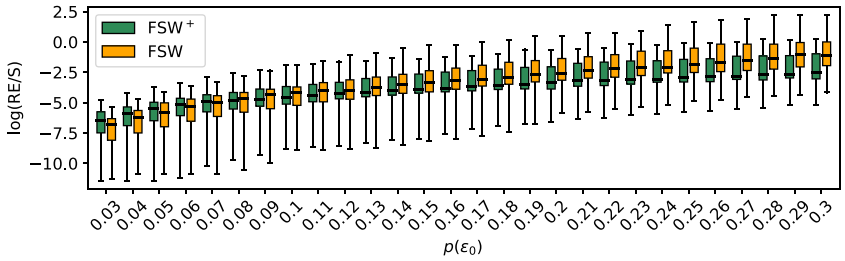



**Fig. 5.** RE/S achieved by FSW$^+$ versus FSW on datasets A1-A20 for different error bounds.

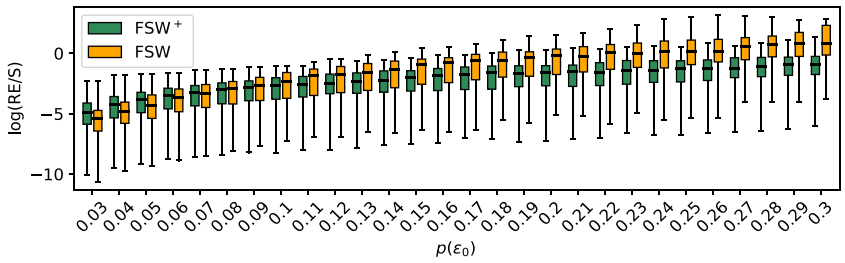



**Fig. 6.** RE/S achieved by FSW$^+$ versus FSW on datasets B1-B20 for different error bounds.

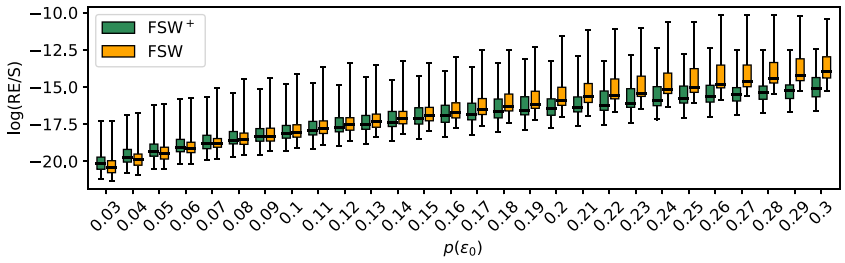



**Fig. 7.** RE/S achieved by FSW$^+$ versus FSW on datasets C1-C20 for different error bounds.

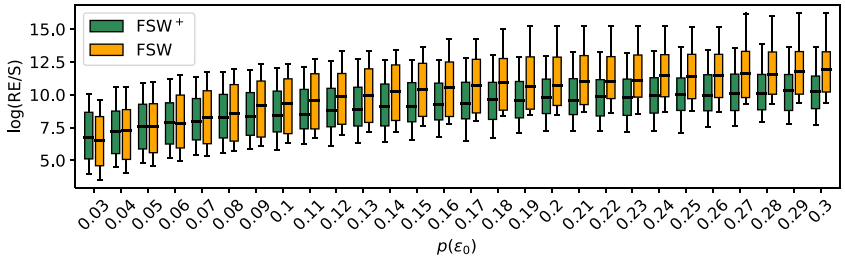



**Fig. 8.** RE/S achieved by FSW$^+$ versus FSW on datasets D1-D20 for different error bounds.

varied range of values, we present RE/S values on a logarithmic scale. The numbers on the horizontal axis represent different values of $p$ (the user-specified error percentage in Eq. (5)), which is proportional to $\epsilon_0$, denoted as $p(\epsilon_0)$.

In each figure, results are shown in form of box plots for datasets in the same category. For instance, each box in Fig. 4 summarises the distribution of results of RE/S for a given $p$ obtained from all datasets (A1–A20) in the *Astrophysics* category. Since the nature of the time-series data in each category is similar, it is reasonable to present the results obtained from them in the form of a distribution. As discussed before, allocating adaptive error thresholds to different intervals properly maintains more features of the original time series after the compression. Therefore, we expect FSW$^+$ to lead to a better approximation quality with lower RE/S under the same global error bound.

As seen in Figs. 5 to 8, FSW$^+$ clearly outperforms FSW for tight storage restrictions (i.e., large $\epsilon_0$) by having lower RE/S values. However, FSW$^+$ and FSW provide similar results in terms of median RE/S under low compression ratios (error bounds below 0.07) and FSW tends to perform better for extremely low error thresholds (close to 0). The reason is that such tight restrictions will lead to over-fragmented approximations of the entire time series by FSW, with low representation error but a large number of segments. For high compression ratios, *i.e.*, when using looser representation error restrictions, fewer segments should be used during the approximation process. FSW$^+$ is designed to assign fewer segments within relatively stable periods (with small $\beta$) and more segments for the intervals containing more fluctuations. This means that the overall approximation quality of FSW$^+$ could be better than that of FSW by avoiding unnecessary over-fragmentation, regardless of a higher value of RE/S. We will inspect this with further analysis.

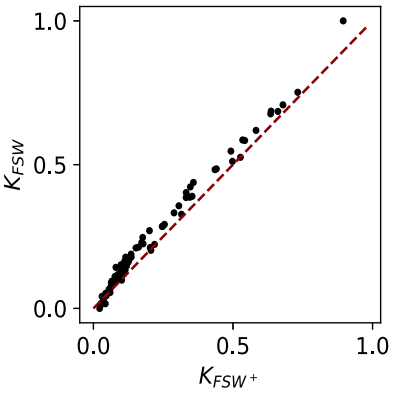



**Fig. 9.** Comparison between $K_{FSW^+}$ and $K_{FSW}$ on all datasets under a tight error bound ($\epsilon_0 = 0.03$).

**Table 5**
The average compression ratio of FSW$^+$ and FSW on all datasets under the same global error bound ($\epsilon_0 = 0.03$). Results presented in boldface denote statistically significantly higher performance, which is verified by means of a Wilcoxon signed-rank test with a significance level of 0.05 ($p < \mathbf{10^{-5}}$).

| Datasets | | A1–A20 | B1–B20 | C1–C20 | D1–D20 | Average |
|---|---|---|---|---|---|---|
| $\rho$ | FSW$^+$ | **7.583** | **5.323** | **6.386** | **4.296** | **5.897** |
| | FSW | 6.028 | 4.554 | 6.158 | 3.872 | 5.153 |

So far, we have not explicitly considered the effect of $\epsilon_0$ on the number of segments, which under some conditions, can cause over-fragmentation. In Fig. 9, we, therefore, compare the number of segments produced by FSW$^+$ and FSW across all datasets under a tight error bound ($\epsilon_0 = 0.03$) (the number of segments from the two methods, denoted by $K_{FSW^+}$ and $K_{FSW}$, have been normalised using Min-Max Normalisation). As can be clearly seen from these results, FSW generally produces a larger number of segments. While this can lead to better results in terms of RE/S (as previously observed in Figs. 5 to 8), it generally does not yield better approximations, as over-fragmentation should be avoided in time-series data compression (Keogh et al., 2004). In Table 5, we show the average compression ratio (denoted by $\rho$) of FSW$^+$ and FSW on all datasets under the same global error bound ($\epsilon_0 = 0.03$). The compression ratio is defined as the ratio of the original data size to the compressed data size. The compression ratio attained by FSW$^+$ is higher than that achieved by FSW (10%, in general). The significance of the differences is verified by means of a Wilcoxon signed-rank test (since the data do not follow a normal distribution) with a significance threshold of 0.05 ($p < 10^{-5}$). Our results clearly indicate that the proposed method can yield a more compact representation of the original time series with adaptive approximation error constraints.

Inevitably, with smaller values of $\epsilon_0$, FSW$^+$ also generates a large number of segments (much smaller than FSW). However, a closer inspection of the results in Fig. 10 shows that a large proportion of segments generated by FSW$^+$ are from parts of the time series with large fluctuations, which cause a decrease in $\epsilon_m$. On the other hand, the adaptive increase of $\epsilon_m$ performed by FSW$^+$ within stable periods of a given time series helps to reduce overall over-fragmentation. Fig. 10 illustrates this phenomenon for FSW$^+$ and FSW on dataset A4 with $\epsilon_0 = 0.03$. Here, fluctuating periods of the time series are highlighted in red. FSW$^+$ generates considerably fewer segments than FSW (235 *vs.* 539) within the stable periods, while FSW$^+$ and FSW both generate a similar number of segments (547 *vs.* 543) for the fluctuating periods. This is shown in Figs. 10(c) and 10(d), where we visualise the piecewise linear approximation results given by FSW$^+$ and FSW within the same stable period. We can see that FSW$^+$ allocates a fewer number of segments than FSW in this period. These results illustrate the poor (and somewhat extreme) trade-off between the number of segments and approximation error achieved by FSW, which is avoided by FSW$^+$. This also explains the smaller RMSE of FSW compared to FSW$^+$, which results from over-fragmentation of the stable periods of the time series (in this case, the RMSE value is 7.278 for FSW and 16.188 for FSW$^+$), while FSW$^+$ performs slightly better than FSW in the fluctuating periods (in this case, the RMSE value is 2.114 for FSW and 2.102 for FSW$^+$).

Our comparison with respect to RE/S presented in Figs. 5 to 8 clearly demonstrates the advantage achieved by AEPLA for tight storage restrictions (i.e., large $\epsilon_0$). This advantage becomes even clearer when comparing the approximation quality of FSW$^+$ and FSW for the same amount of storage space, i.e., the representation error when using the same number of segments, as per our second research question (RQ2).

To address RQ2, we now turn towards analysing the trade-off between the number of segments and representation error using the hypervolume indicator using Fig. 11. In Fig. 11(a), we show the representation error as a function of the number of segments for both FSW$^+$ and FSW on the dataset (A1). As can be clearly seen in the figure, for all but the highest numbers of segments, FSW$^+$ achieves considerably lower representation error than FSW. Figs. 11(b) and 11(c) illustrate how the sets of solutions obtained from FSW$^+$ and FSW, respectively, are used to calculate HI on dataset A1. In these figures, the non-dominated solutions are marked with squares, and the hypervolume is indicated by the shaded area, bounded from above by a reference point, which is determined by the maximal possible value of the two objective functions, considering all the non-dominated solutions produced by FSW$^+$ and FSW. As seen in Figs. 11(b) and 11(c), the dominated solutions are not considered in the calculation of HI because they fall into the hypervolume determined by the non-dominated points. Generally, the indicator quantifies the space which is dominated by a set of solutions $S$. The larger the value of this indicator, the better the performance of the algorithm in question.

Fig. 12 compares the values of the hypervolume indicator obtained from FSW$^+$ versus FSW on all datasets, in terms of relative difference (RD) defined as

$$RD = \frac{HI_{FSW^+} - HI_{FSW}}{HI_{FSW}} \tag{7}$$

where $HI_{FSW^+}$ and $HI_{FSW}$ refer to the hypervolume of FSW$^+$ and FSW respectively. Before the comparison, the values of both objective functions have been normalised using the normalisation mechanism proposed in Ishibuchi et al. (2017); let $max(f_i)$ and $min(f_i)$ be the maximum and minimum values of the objective functions ($f_i$ ($i = 1, 2$) represents $RE(\epsilon_0)$ and $K(\epsilon_0)$, respectively) among non-dominated solutions given by FSW$^+$ and FSW on the same dataset, then the objective value $n(f_i)$ of the $i$th objective $f_i$ is normalised as

$$n(f_i) = \frac{f_i - max(f_i)}{max(f_i) - min(f_i)} \quad for\ i = 1, 2 \tag{8}$$

Positive RD values (within the shaded area) indicate that the hypervolume achieved by FSW$^+$ is larger than the one achieved by FSW (i.e., FSW$^+$ produces a better performance trade-off than FSW); larger absolute values of RD indicate larger differences in hypervolume. We observe that FSW$^+$ outperforms FSW on 58.75% (47/80) of our benchmark datasets according to this metric.

Table 6 shows the detailed distribution of the RD values for FSW$^+$ and FSW on all benchmarks. We can see that FSW$^+$ achieves significantly better performance (5% or more) in at least 22 cases, while the advantage of FSW over FSW$^+$ observed in some cases is rather small.

We further analysed the statistical significance of these results through hypothesis testing. Noting that the values of RD are not normally distributed, we have chosen the non-parametric one-sided Wilcoxon signed-rank test that does not assume a normal distribution of data. For a standard significance level of 0.05, the test rejects the null hypothesis that the median of RD is lower than or equal to zero ($p = 0.0022$), suggesting that FSW$^+$ can provide a significantly better trade-off between the number of segments and representation accuracy.

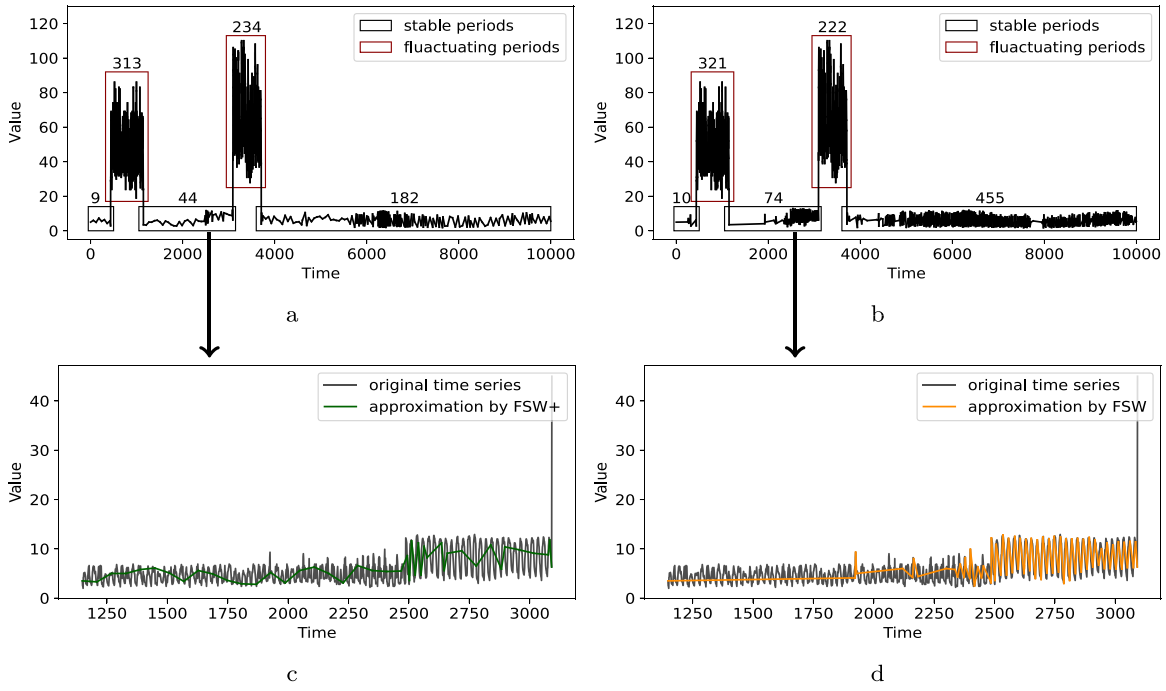



**Fig. 10.** The approximation results produced by FSW$^+$ (a) and FSW (b) on dataset A4 with $\epsilon_0 = 0.03$, where the numbers above the rectangles indicate the number of segments generated in the corresponding intervals. The piecewise linear approximation results given by FSW+ and FSW within the same stable period are shown in (c) and (d), respectively.

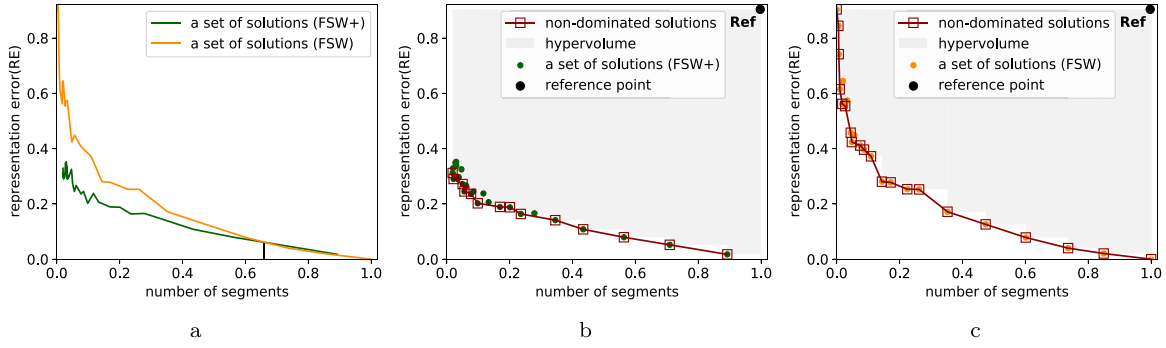



**Fig. 11.** (a) Two sets of solutions given by FSW$^+$ and FSW against the sample set (A1). The non-dominated solutions and the corresponding hypervolume of FSW$^+$ (b) and FSW (c), respectively.

**Table 6**
Distribution of the RD values considering FSW$^+$ and FSW on all benchmarks. Based on the non-parametric one-sided Wilcoxon signed-rank test, the null hypothesis that the median of RD is lower than or equal to zero is rejected with a significance level of 0.05 ($p = 0.0022$).

| RD values | Below −10% | Below −5% | Below −3% | Above 0% | Above +3% | Above +5% | Above +10% |
|---|---|---|---|---|---|---|---|
| Number of cases | 0 | 4 | 10 | 47 | 28 | 22 | 8 |

The comparison of HI values shows that AEPLA has achieved a better balance between the two specific objectives. This can be more intuitively understood by quantifying how much fewer segments are needed by FSW$^+$ to reach the same RE as FSW. For this purpose, we compare the number of segments generated by FSW$^+$ and FSW to achieve a certain RE level, $\gamma$, across all benchmarks. Without loss of generality, we set $\gamma$ as half of the worst RE for the better of the two methods (denoted $RE_{0.5}$, see Fig. 13) on each of the 80 datasets, and compare the corresponding number of segments generated by FSW$^+$ and FSW. In Fig. 13, we show the representation error as a function of the number of segments for both FSW$^+$ and FSW on dataset A1, where $K_{\gamma^+}$ and $K_\gamma$ are the number of segments needed by FSW$^+$ and FSW to reach the same RE level $\gamma$. The average of $K_\gamma/K_{\gamma^+}$ on each of the four sets of benchmarks, as well as on all benchmarks are summarised in Table 7. For the same $\gamma = 0.5$, more segments (a factor of 1.4 in general) are needed by FSW than FSW$^+$, clearly indicating the advantage of AEPLA for time-series compression in terms of the compression ratio.

**Table 7**
Comparison between the number of segments generated by FSW$^+$ ($K_{\gamma^+}$) and FSW ($K_\gamma$) to achieve a certain RE level $\gamma$ (0.5). The average of $K_\gamma/K_{\gamma^+}$ on all benchmarks is shown in the last column.

| Datasets | A1–A20 | B1–B20 | C1–C20 | D1–D20 | Average |
|---|---|---|---|---|---|
| $K_\gamma/K_{\gamma^+}$ | 1.047 | 1.768 | 1.060 | 1.674 | 1.387 |

Finally, we evaluate our method on larger time-series data. Table 8 summarises the compression evaluation results of FSW$^+$ and FSW on four longer time series. Specifically, we present the relative difference between the hypervolume indicator RD, and the average RE/S ($\overline{RE/S}$) with a different global error bound on a logarithmic scale. As seen in

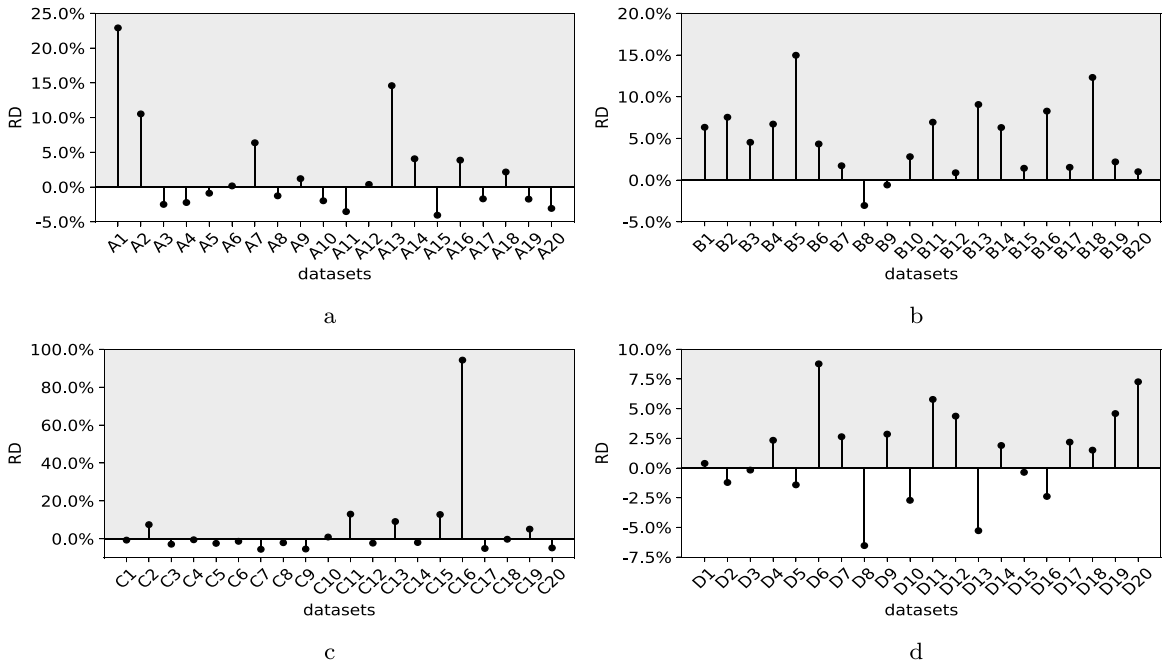



**Fig. 12.** Comparison of FSW$^+$ and FSW with respect to the hypervolume indicator. (a) Results on *Astrophysics* benchmarks (A1-A20). (b) Results on *High-low pricing* benchmarks (B1-B20). (c) Results on *Precipitation rate* benchmarks (C1-C20). (d) Results on *Trade-volume* benchmarks (D1-D20).

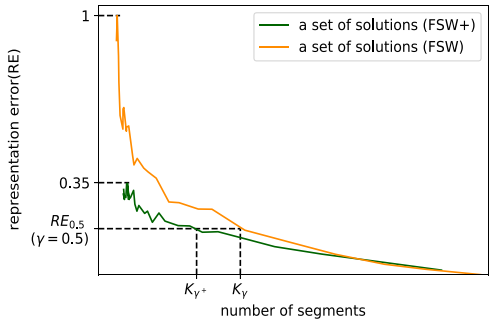



**Fig. 13.** An illustration of $K_\gamma/K_{\gamma^+}$ on dataset A1 where $K_{\gamma^+}$ and $K_\gamma$ are the number of segments needed by FSW$^+$ and FSW to reach the same RE level $\gamma = 0.5$ ($RE_{0.5}$).

**Table 8**
Comparing the performance of FSW$^+$ and FSW on four longer time series.

| Datasets | RD | $\log(\overline{RE/S}_{FSW^+})$ | $\log(\overline{RE/S}_{FSW})$ |
|---|---|---|---|
| CinC_ECG_torso | 3.645% | −1.343 | −1.019 |
| InlineSkate | 1.117% | −1.366 | −1.054 |
| MALLAT | 4.018% | −2.494 | −1.182 |
| StarLightCurves | −0.033% | −1.597 | −1.010 |

Table 8, FSW$^+$ generally achieves better performance on RD, while the advantage of FSW is rather small (in one of the four cases). Furthermore, the average RE/S of FSW$^+$ is significantly lower than that of FSW in all cases, indicating the advantage of our method in terms of approximation quality regardless of the volume of data to be compressed.

### 5.3. Experiments on disjoint PLA

So far, we have addressed RQ1 and RQ2 for the case of continuous PLA methods. In this section, we analyse AEPLA for disjoint $l_\infty$-PLA methods to address the same research questions; here, we chose OptimalPLR as a representative method in this category.

**Table 9**
The average compression ratio of OptimalPLR$^+$ and OptimalPLR on all datasets under the same global error bound ($\epsilon_0 = 0.03$). Results presented in boldface denote statistically significantly higher performance, which is verified through Wilcoxon signed-rank test with a significance level of 0.05 ($p < \mathbf{10^{-5}}$).

| Datasets | | A1–A20 | B1–B20 | C1–C20 | D1–D20 | Average |
|---|---|---|---|---|---|---|
| $\rho$ | OptimalPLR$^+$ | **7.779** | **5.201** | **5.718** | 3.334 | **5.508** |
| | OptimalPLR | 6.757 | 4.785 | 5.237 | **3.624** | 5.101 |

Regarding RQ1, Figs. 14 to 17 show the performance of OptimalPLR$^+$ versus OptimalPLR across all benchmarks for different global error bound settings. This comparison yields similar results to those seen for FSW$^+$ versus FSW: The performance of OptimalPLR$^+$ compared to OptimalPLR increases with the error bound. The observed performance advantage of OptimalPLR$^+$ is more pronounced for higher compression ratios (i.e., higher error bounds), especially on datasets B1-B20 and D1-D20. Table 9 shows the average compression ratio of OptimalPLR$^+$ and OptimalPLR on all datasets with the same global error bound ($\epsilon_0 = 0.03$). The significance of the differences is analysed using a Wilcoxon signed-rank test ($p < 10^{-5}$). The results show that OptimalPLR$^+$ often yields significantly higher compression ratios.

Regarding RQ2, the comparison of the hypervolume indicators shown in Fig. 18 and the RD values presented in Table 10 shows that OptimalPLR$^+$ outperforms OptimalPLR on 58.75% (47/80) of our benchmarks. These results are statistically significant according to the one-sided Wilcoxon signed-rank test at a standard significance level of 0.05 ($p = 0.0046$).

We also compare the number of segments generated by OptimalPLR$^+$ and OptimalPLR for achieving a given RE level $\gamma$ across all benchmarks. The averages of $K_\gamma/K_{\gamma^+}$ for each of the four categories of benchmarks as well as across all benchmarks are summarised in Table 11. For the same $\gamma = 0.5$, substantially more segments (a factor of 1.7 in general) are generated by OptimalPLR than by OptimalPLR$^+$.

Further inspection of results reveals that for several datasets and error bounds, OptimalPLR generates unreliable approximations; Fig. 19 shows examples of these unreliable approximations on benchmarks (A1 and A6). The reason for this undesirable behaviour is that the original disjoint $l_\infty$-PLA method may fail to recognise some relatively small

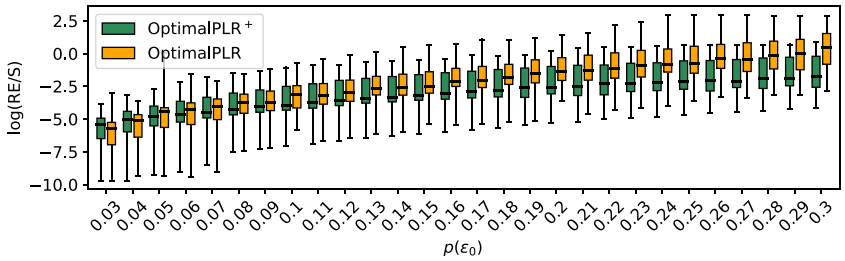



**Fig. 14.** RE/S achieved by OptimalPLR$^+$ versus OptimalPLR on datasets A1-A20 for different error bounds.

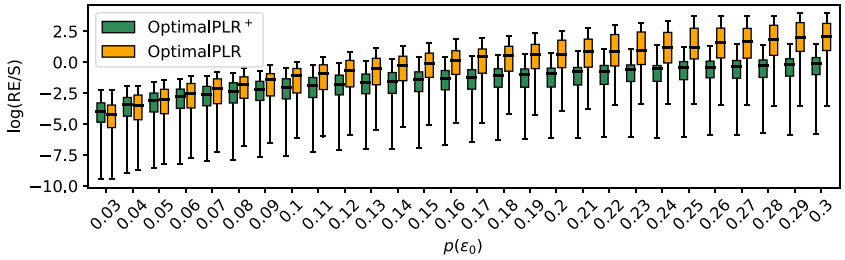



**Fig. 15.** RE/S achieved by OptimalPLR$^+$ versus OptimalPLR on datasets B1-B20 for different error bounds.

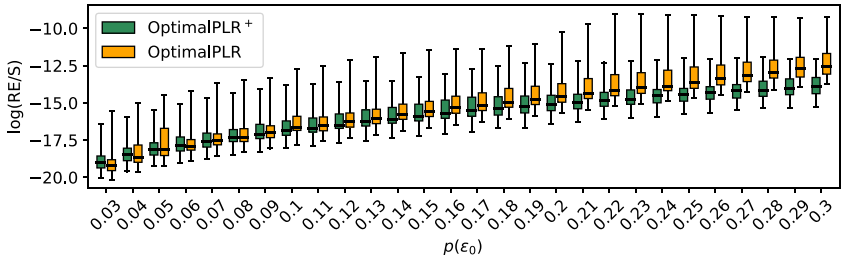



**Fig. 16.** RE/S achieved by OptimalPLR$^+$ versus OptimalPLR on datasets C1-C20 for different error bounds.

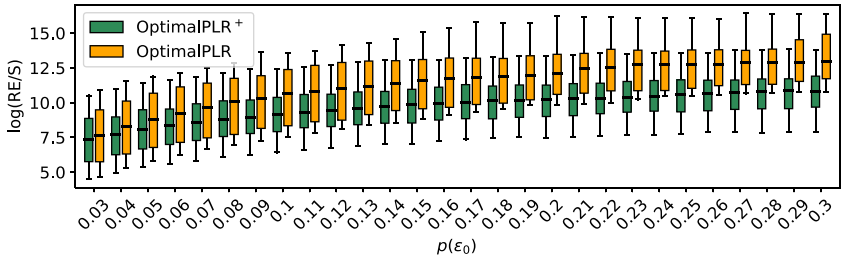



**Fig. 17.** RE/S achieved by OptimalPLR$^+$ versus OptimalPLR on datasets D1-D20 for different error bounds.

**Table 10**
Distribution of the RD values considering OptimalPLR$^+$ and OptimalPLR on 80 benchmarks. Based on the non-parametric one-sided Wilcoxon signed-rank test, the null hypothesis that the median of RD is lower than or equal to zero is rejected with a significance level of 0.05 ($p = 0.0046$).

| RD values | Below −10% | Below −5% | Below −3% | Above 0% | Above +3% | Above +5% | Above +10% |
|---|---|---|---|---|---|---|---|
| Number of cases | 0 | 2 | 7 | 47 | 14 | 6 | 2 |

**Table 11**
Comparison between the number of segments used in OptimalPLR$^+$ ($K_{\gamma^+}$) and OptimalPLR ($K_\gamma$) to achieve a certain RE level $\gamma$ (0.5). The average of $K_\gamma/K_{\gamma^+}$ on all benchmarks is shown in the last column.

| Datasets | A1–A20 | B1–B20 | C1–C20 | D1–D20 | Average |
|---|---|---|---|---|---|
| $K_\gamma/K_{\gamma^+}$ | 1.706 | 2.324 | 1.472 | 1.184 | 1.672 |

fluctuations with a fixed error bound. For instance, if the difference between a local maximum and minimum is lower than the specified error bound, the original time series within the corresponding period will be approximated by one segment, leading to an increased loss of information and higher approximation error. On the other hand, OptimalPLR$^+$ avoids this problem by adjusting the error bound accordingly, leading to a more stable performance.

We finally compare the approximation quality of OptimalPLR$^+$ and OptimalPLR on larger time-series data. As seen in Table 12, we can

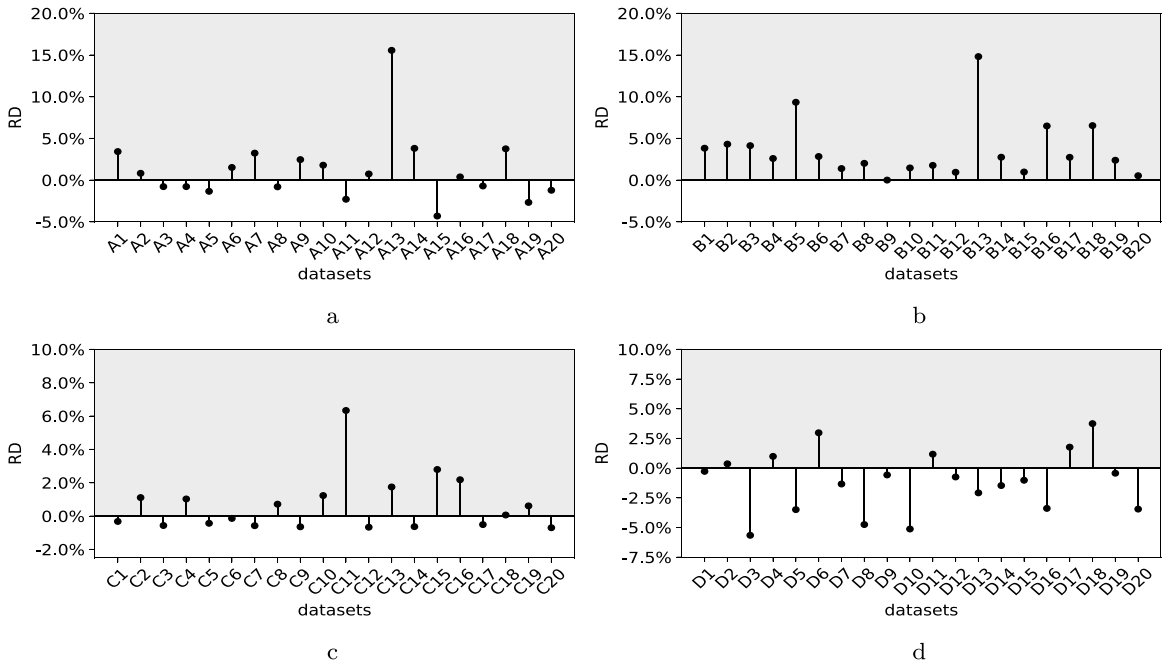



**Fig. 18.** Comparison of OptimalPLR$^+$ and OptimalPLR with respect to the hypervolume indicator against all datasets. (a) Results on *Astrophysics* datasets (A1-A20). (b) Results on *High-low pricing* datasets (B1-B20). (c) Results on *Precipitation rate* datasets (C1-C20). (d) Results on *Trade-volume* datasets (D1-D20).

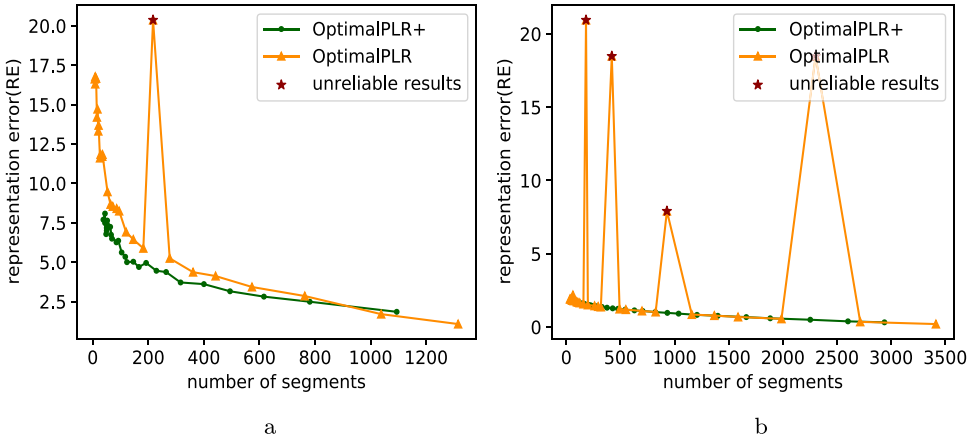



**Fig. 19.** Trade-off between the number of segments and representation error for OptimalPLR$^+$ versus OptimalPLR for two datasets (A1 and A6) where OptimalPLR produces unreliable results for some error bounds.

**Table 12**
Comparing the performance of OptimalPLR$^+$ and OptimalPLR on four longer time series.

| Datasets | RD | $\log(\overline{RE/S}_{OptimalPLR^+})$ | $\log(\overline{RE/S}_{OptimalPLR})$ |
|---|---|---|---|
| CinC_ECG_torso | 3.650% | −1.300 | −1.027 |
| InlineSkate | 1.222% | −1.161 | −1.014 |
| MALLAT | 1.789% | −1.353 | −1.067 |
| StarLightCurves | 0.644% | −1.147 | −0.921 |

draw a similar conclusion to those comparing FSW$^+$ against FSW: OptimalPLR$^+$ can yield a better performance trade-off with higher RD and better approximation quality with a lower RE/S.

### 5.4. Experiments on semi-connected PLA

In this section, we further verify the applicability of AEPLA for semi-connected $l_\infty$-PLA methods by addressing research questions RQ1 and RQ2; here, we chose SemiOpt as a representative method in this category.

Regarding RQ1, the performance of SemiOpt$^+$ versus SemiOpt across all benchmarks are compared in Figs. 20 to 23. We can draw similar conclusions from the comparisons as those in continuous and disjoint cases: SemiOpt$^+$ can avoid over-fragmentation under tight error bounds; the performance advantage of SemiOpt$^+$ over SemiOpt becomes more pronounced with the increase of the error bound. In Table 13, we show the average compression ratio $\rho$ of SemiOpt$^+$ and SemiOpt on all datasets under a tight global error bound ($\epsilon_0 = 0.03$). We can see that the compression ratio attained by SemiOpt$^+$ is higher than that achieved by SemiOpt by 18% in general. The significance of the differences is verified by Wilcoxon signed-rank test ($p < 10^{-5}$). These results confirm that the proposed method can yield a more compact representation of the original time series.

Regarding RQ2, we compare the values of the hypervolume indicator obtained from SemiOpt$^+$ versus SemiOpt on all datasets in terms of relative difference (RD) in Fig. 24 and Table 14. We can observe

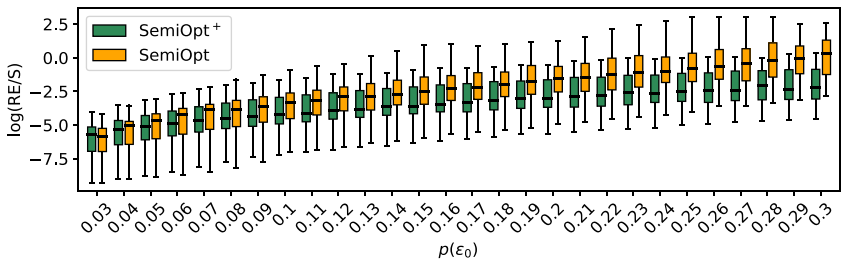



**Fig. 20.** RE/S achieved by SemiOpt⁺ versus SemiOpt on datasets A1-A20 for different error bounds.

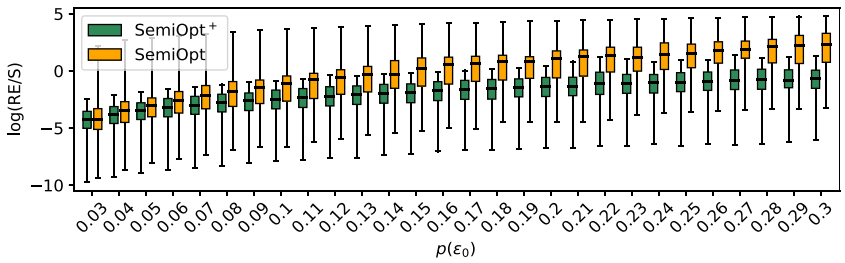



**Fig. 21.** RE/S achieved by SemiOpt⁺ versus SemiOpt on datasets B1-B20 for different error bounds.

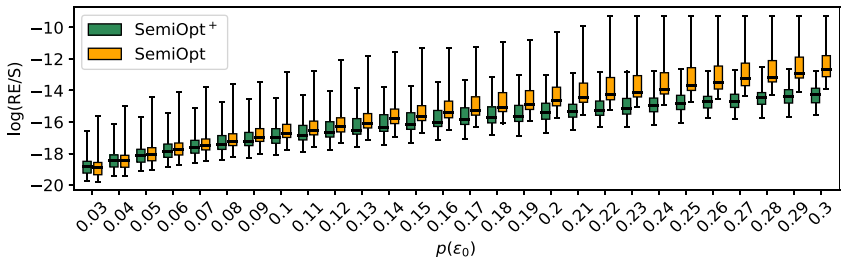



**Fig. 22.** RE/S achieved by SemiOpt⁺ versus SemiOpt on datasets C1-C20 for different error bounds.

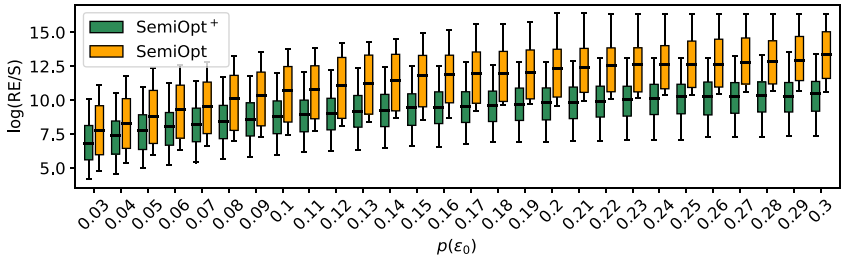



**Fig. 23.** RE/S achieved by SemiOpt⁺ versus SemiOpt on datasets D1-D20 for different error bounds.

**Table 13**
The average compression ratio of SemiOpt⁺ and SemiOpt on all datasets under the same global error bound ($\epsilon_0 = 0.03$). Results presented in boldface denote statistically significantly higher performance, which is verified through Wilcoxon signed-rank test with a significance level of 0.05 ($p < \mathbf{10^{-5}}$).

| Datasets | | A1-A20 | B1-B20 | C1-C20 | D1-D20 | Average |
|---|---|---|---|---|---|---|
| $\rho$ | SemiOpt⁺ | **7.974** | **5.270** | **6.105** | **3.288** | **5.659** |
| | SemiOpt | 6.377 | 4.157 | 5.601 | 2.922 | 4.764 |

an obvious advantage of SemiOpt⁺ from the comparison: SemiOpt⁺ outperforms SemiOpt on 91.25% (73/80) of our benchmarks. The difference is statistically significant according to the one-sided Wilcoxon signed-rank test at a standard significance level of 0.05 ($p < 10^{-5}$).

In Table 15, we compare the number of segments generated by SemiOpt⁺ and SemiOpt for achieving a given RE level $\gamma$ across all benchmarks. For the same $\gamma = 0.5$, SemiOpt generates substantially more segments (on average a factor of 2.1 in general) than SemiOpt⁺. In other words, SemiOpt⁺ can achieve a higher compression ratio under the same RE level.

We finally demonstrate the applicability of AEPLA based on semi-connected $l_\infty$-PLA methods to large volumes of data. Table 16 summarises the hypervolume indicator RD and the average RE/S ($\overline{RE/S}$) of SemiOpt⁺ and SemiOpt on four longer time series. We can see that SemiOpt⁺ achieves better performance in all cases, with positive RD values and lower $\overline{RE/S}$ values, respectively.

### 5.5. Time and space costs

We have addressed RQ1 and RQ2 by applying the proposed framework to different types of PLA methods. In this section, we turn to RQ3 to compare the actual running time and maximum space usage between $M^+$ and $M$ under the same global error bound ($\epsilon_0 = 0.03$).

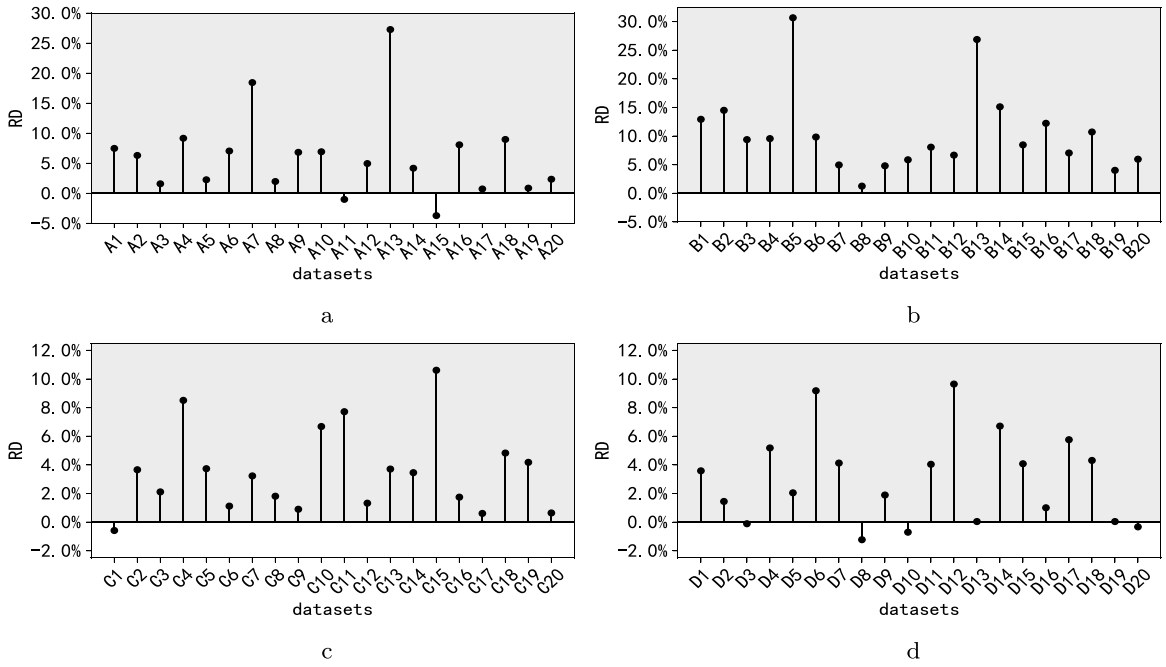



**Fig. 24.** Comparison of SemiOpt$^+$ and SemiOpt with respect to the hypervolume indicator against all datasets. (a) Results on *Astrophysics* datasets (A1-A20). (b) Results on *High-low pricing* datasets (B1-B20). (c) Results on *Precipitation rate* datasets (C1-C20). (d) Results on *Trade-volume* datasets (D1-D20).

**Table 14**
Distribution of the RD values considering SemiOpt$^+$ and SemiOpt on 80 benchmarks. Based on the non-parametric one-sided Wilcoxon signed-rank test, the null hypothesis that the median of RD is lower than or equal to zero is rejected with a significance level of 0.05 ($p < \mathbf{10^{-5}}$).

| RD values | Below −10% | Below −5% | Below −3% | Above 0% | Above +3% | Above +5% | Above +10% |
|---|---|---|---|---|---|---|---|
| Number of cases | 0 | 0 | 1 | 73 | 52 | 35 | 10 |

**Table 15**
Comparison between the number of segments used in SemiOpt$^+$ ($K_{\gamma^+}$) and SemiOpt ($K_\gamma$) to achieve a certain RE level $\gamma$ (0.5). The average of $K_\gamma/K_{\gamma^+}$ on all benchmarks is shown in the last column.

| Datasets | A1–A20 | B1–B20 | C1–C20 | D1–D20 | Average |
|---|---|---|---|---|---|
| $K_\gamma/K_{\gamma^+}$ | 1.964 | 3.445 | 1.428 | 1.722 | 2.140 |

**Table 16**
Comparing the performance of SemiOpt$^+$ and SemiOpt on four longer time series.

| Datasets | RD | $\log(\overline{RE/S}_{SemiOpt^+})$ | $\log(\overline{RE/S}_{SemiOpt})$ |
|---|---|---|---|
| CinC_ECG_torso | 2.34% | −8.198 | −7.304 |
| InlineSkate | 0.21% | −8.572 | −7.982 |
| MALLAT | 1.44% | −11.771 | −10.980 |
| StarLightCurves | 4.67% | −10.944 | −10.155 |

In Table 17, we first show the average actual running time of $M^+$ and $M$ on the 80 datasets. We can see that the average time costs of $M^+$ are close to that of $M$, with minor increases (4.5%, 5.6% and 4.5% for FSW, OptimalPLR and SemiOpt in average, respectively) due to the construction of the buffer and the computation of the fluctuation coefficient $\beta$, which verifies the claim made in Property 4.3. Besides, let $t(F)$, $t(O)$, and $t(S)$ be the average time cost of FSW, OptimalPLR, and SemiOpt respectively, the results in Table 17 confirm that $t(F) < t(O) < t(S)$, which is determined by the complexity of the corresponding PLA methods. This is also in line with the conclusions drawn from the original papers (Liu et al., 2008; Xie et al., 2014; Zhao et al., 2022).

We then compare the average space usage of $M^+$ and $M$. In Table 18, we also give the space costs of $M$ with a fixed error bound $\epsilon_0 \cdot e^{-0.5}$ (denoted as $M^{-0.5}$), and $M$ with a fixed error bound $\epsilon_0 \cdot e^{0.5}$ (denoted as $M^{0.5}$). The results in Table 18 verify that: (i) $M^+$ achieves a bit lower space costs than $M$ (0.08%, 0.02% and 0.04% for FSW, OptimalPLR and SemiOpt in average, respectively) since $M^+$ generates more compact representation; (ii) the space usage of $M^+$ is bounded by that of $M^{-0.5}$ and $M^{0.5}$ as claimed in Property 4.1.

### 5.6. Choice of the scaling factor $\alpha$

So far, we have addressed RQ1 to RQ3 for continuous, disjoint, and semi-connected PLA methods. In this section, we turn to RQ4 and study the impact of the scaling factor $\alpha$ parameter in Eq. (3), which affects the length of the buffer, *buf*. Generally, the choice of $\alpha$ is related to the length of the original time series. If the time series is extremely long, the length of the buffer determined by Eq. (3) will be relatively large. In this case, the local error bounds determined by Eq. (4) might be lower in general, which could undermine the influence of adjusting the error bound. In previous sections, $\alpha$ has been set to a default value of 0.2 based on preliminary tests. In this section, we introduce a procedure for the optimisation of $\alpha$ in a data-driven manner to address this issue.

To evaluate the approximation quality of AEPLA with adaptive $\alpha$, we would like to automatically select the optimal $\alpha$ for each dataset. For this purpose, we use an algorithm configuration tool that is typically used for efficiently optimising algorithm hyperparameters. Specifically, we chose SMAC (sequential model-based algorithm configuration) (Hutter et al., 2011), a freely available, prominent automated configuration method known to be able to optimise algorithm parameters substantially more efficiently than traditional alternatives, such as (exhaustive) grid search.

Using SMAC, we can automatically determine parameter settings that maximise the performance of a given algorithm according to an

**Table 17**
Comparison between $M^+$ and $M$ in terms of the time costs (ms).

| Datasets | FSW | | OptimalPLR | | SemiOpt | |
|---|---|---|---|---|---|---|
| | $M^+$ | $M$ | $M^+$ | $M$ | $M^+$ | $M$ |
| A1–A20 | 34.398 ± 8.125 | 34.064 ± 6.121 | 129.010 ± 23.715 | 127.160 ± 25.309 | 270.68 ± 41.986 | 239.807 ± 54.897 |
| B1–B20 | 16.866 ± 5.404 | 15.731 ± 5.162 | 57.562 ± 24.402 | 53.492 ± 20.410 | 881.359 ± 37.201 | 849.986 ± 36.268 |
| C1–C20 | 38.064 ± 10.540 | 37.448 ± 9.585 | 136.517 ± 25.977 | 134.255 ± 27.338 | 256.419 ± 31.470 | 255.443 ± 34.875 |
| D1–D20 | 8.583 ± 2.066 | 7.926 ± 2.275 | 25.841 ± 10.875 | 23.142 ± 10.159 | 30.489 ± 13.608 | 30.174 ± 13.744 |

**Table 18**
Comparison between $M^+$ and $M$ in terms of the space costs (mb).

| Datasets | FSW | | | | OptimalPLR | | | | SemiOpt | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $M^+$ | $M$ | $M^{-0.5}$ | $M^{0.5}$ | $M^+$ | $M$ | $M^{-0.5}$ | $M^{0.5}$ | $M^+$ | $M$ | $M^{-0.5}$ | $M^{0.5}$ |
| A1–A20 | 83.941 | 84.091 | 84.508 | 83.636 | 83.386 | 83.464 | 83.702 | 83.28 | 83.31 | 83.352 | 83.462 | 83.297 |
| B1–B20 | 83.28 | 83.464 | 83.732 | 83.176 | 82.949 | 82.977 | 83.216 | 82.912 | 83.016 | 83.078 | 83.219 | 82.975 |
| C1–C20 | 84.724 | 84.639 | 85.065 | 84.336 | 83.866 | 83.827 | 84.095 | 83.628 | 83.589 | 83.572 | 83.628 | 83.538 |
| D1–D20 | 82.703 | 82.744 | 82.877 | 82.683 | 82.646 | 82.661 | 82.764 | 82.579 | 82.658 | 82.702 | 82.756 | 82.568 |
| Average | 83.641 | 83.756 | 84.046 | 83.458 | 83.202 | 83.242 | 83.444 | 83.100 | 83.143 | 83.176 | 83.266 | 83.095 |

objective function. Here, we maximise $HI$, motivated by our goal of finding a buffer length that maintains the balance between approximation quality and the compression ratio regardless of the choice of $\epsilon_0$. For this purpose, $M^+$ (the corresponding adaptive versions of the original PLA methods, as introduced earlier) is optimised using SMAC over the entire range of $\alpha$ for each dataset. The configurations with the largest value of the hypervolume indicator and the corresponding optimal value of $\alpha$ are recorded. The goal is to test whether $M^+$ with adaptive $\alpha$ ($M^+_\alpha$) can outperform $M^+$ in terms of this indicator. Details on our experimental setup are given below.

For each dataset, we performed one run of SMAC v.3 (Lindauer et al., 2017), with the objective of maximising *HI*, using a time budget of 50 runs of $M^+_\alpha$, the target algorithm to be optimised (i.e., up to 50 parameter configurations were evaluated in each run of SMAC). This time budget was chosen based on preliminary experiments, in which we observed that $M^+_\alpha$ performed better than $M^+$ after 50 runs and that using more runs did not result in significant further performance improvements. There is no running time limit for each target algorithm run. The configuration space of parameter $\alpha$ has been set to the range of $2/(n \cdot \epsilon_0)$ to 1. All other settings of SMAC have been left at their defaults. All our experiments were conducted on compute nodes, each equipped with Dual 16-Core Intel Xeon E5-2683 CPUs (2.10 GHz) with 40 MB cache and 94 GB RAM, running CentOS Linux 3.10.0-1127.

In Table 19, we show the optimal value of $\alpha$ for FSW$^+$ ($\alpha_F$), OptimalPLR$^+$ ($\alpha_O$), and SemiOpt$^+$ ($\alpha_S$) determined by SMAC on each of the 80 datasets. We can see that the optimal value of $\alpha$ for different types of PLA methods are similar for the same dataset and that the optimal value of $\alpha$ varies among different benchmark datasets. By looking into the nature of each dataset, we found that SMAC tends to determine a relatively small value of $\alpha$ for datasets containing consistently large fluctuations. For instance, we show the original time series B8 and C2 in Fig. 25. Evidently, B8 is relatively stable, while C2 represents a dataset with large fluctuations. The corresponding values of $\alpha_O$ determined by SMAC for these two datasets are 0.99 and 0.07, respectively. According to Eq. (3), the size of $buf$ for C2 will become considerably smaller. As a result, the value of $\epsilon_m$ will be adjusted within shorter and more stable temporal stages (by Eq. (4)) so that local features can be better preserved.

In Fig. 26, we present the overall comparison of $HI_{M^+_\alpha}$ and $HI_M$ on all datasets. The $x$-axis represents the value of HI for $M^+_\alpha$ and the $y$-axis for $M$. We can observe that the vast majority of points fall into the lower triangle region, corresponding to a positive relative difference between $HI_{M^+_\alpha}$ and $HI_M$. The results of the Wilcoxon signed-rank test verify the statistical significance with 95% confidence level (*p-value* $< 10^{-5}$ for all the three cases). FSW$^+_\alpha$ outperforms FSW in 69 out of 80 cases, OptimalRLR$^+_\alpha$ outperforms OptimalRLR in 71 out of 80 cases, and SemiOpt$^+_\alpha$ outperforms SemiOpt in 77 out of 80 cases. This suggests that constructing $l_\infty$-PLA under AEPLA with adaptive scaling factor $\alpha$ can generally provide a more reasonable trade-off for the approximation quality and the compression ratio.

It should be mentioned that this optimisation process requires knowledge of the original time series to select the optimal $\alpha$. In other words, this approach can be regarded as an optional pre-processing step in an offline mode. By taking this approach, users can either set $\alpha$ as default in a streaming mode or apply the optimisation results from SMAC to the segmentation process when having the full time series. We note that users are advised to adopt the optimisation process when targeting extremely long time series. This is because the influence of adjusting the error bound might be undermined when using a relatively large buffer size. Furthermore, the fact that we obtained similar optimised values of $\alpha$ for each category of benchmarks suggests that this offline optimisation process can be used successfully to select the value of $\alpha$ across similar datasets.

### 5.7. Further discussions

Piecewise linear approximation is essentially focused on achieving high representation quality and reducing storage usage. However, these two objectives are conflicting and optimising them both simultaneously requires achieving a trade-off. Current $l_\infty$-PLA methods are limited by the fixed error constraint on each data point, which does not allow finding an optimum solution when facing this conflict. Under the new framework, we relax the tight restriction on each data point and adjust the error bound in the approximation process. With an adaptive error bound, the proposed framework can better allocate limited storage resources to preserve more features of a given time series. This claim has been verified in our experiments by addressing the research questions. Regarding RQ1, we demonstrate the advantage of the proposed framework for tight storage restrictions with a lower $RE/S$; regarding RQ2, we prove that a better trade-off between the representation quality and the storage usage can be achieved under the new framework.

However, addressing RQ3, we also notice that the extra steps in the new framework can increase the processing time by up to 10% in some cases. Furthermore, as it has also been claimed in Property 4.1, the original $l_\infty$-PLA method can outperform the new framework under some special cases in terms of space cost. Let $r$ be the number of segments generated by $M$, which is proportional to the space cost, we can see in Table 18 that $r^+$ is higher than $r$ when dealing with datasets C1–C20. As it has been shown in Fig. 25, C1–C20 contains generally large fluctuations, thus the local error bound $\epsilon_m$ is generally lower than $\epsilon_0$ as $\beta > 0.5$ (see Eq. (4)). These results in an increase in the number of segments provided by $M^+$, i.e., $r^+$ can be larger than $r$.

**Table 19**
The optimal value of $\alpha$ for FSW$^+$ ($\alpha_F$), OptimalPLR$^+$ ($\alpha_O$), and SemiOpt$^+$ ($\alpha_S$) determined by SMAC on each of the 80 datasets.

| No. | $\alpha_F$ | $\alpha_O$ | $\alpha_S$ | No. | $\alpha_F$ | $\alpha_O$ | $\alpha_S$ | No. | $\alpha_F$ | $\alpha_O$ | $\alpha_S$ | No. | $\alpha_F$ | $\alpha_O$ | $\alpha_S$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A1 | 0.13 | 0.08 | 0.09 | B1 | 0.2 | 0.18 | 0.11 | C1 | 0.05 | 0.05 | 0.06 | D1 | 1 | 0.93 | 0.85 |
| A2 | 0.07 | 0.05 | 0.05 | B2 | 0.3 | 0.2 | 0.17 | C2 | 0.06 | 0.07 | 0.14 | D2 | 0.61 | 0.34 | 0.37 |
| A3 | 0.04 | 0.05 | 0.06 | B3 | 0.16 | 0.17 | 0.15 | C3 | 0.04 | 0.04 | 0.05 | D3 | 0.64 | 1 | 0.06 |
| A4 | 0.83 | 0.88 | 0.29 | B4 | 0.2 | 0.27 | 0.12 | C4 | 0.09 | 0.1 | 0.12 | D4 | 0.3 | 0.38 | 0.2 |
| A5 | 0.83 | 0.98 | 0.1 | B5 | 0.36 | 0.33 | 0.3 | C5 | 0.05 | 0.04 | 0.05 | D5 | 0.99 | 0.98 | 0.63 |
| A6 | 0.95 | 0.95 | 0.02 | B6 | 0.23 | 0.37 | 0.6 | C6 | 0.06 | 0.05 | 0.04 | D6 | 0.25 | 0.24 | 0.23 |
| A7 | 0.07 | 0.11 | 0.11 | B7 | 0.22 | 0.43 | 0.36 | C7 | 0.09 | 0.05 | 0.04 | D7 | 0.31 | 0.27 | 0.26 |
| A8 | 0.11 | 0.17 | 0.16 | B8 | 0.61 | 0.99 | 0.96 | C8 | 0.05 | 0.05 | 0.04 | D8 | 0.91 | 0.91 | 0.2 |
| A9 | 0.17 | 0.15 | 0.13 | B9 | 0.77 | 0.82 | 0.28 | C9 | 0.04 | 0.05 | 0.06 | D9 | 0.48 | 0.56 | 0.66 |
| A10 | 0.36 | 0.4 | 0.43 | B10 | 0.42 | 0.43 | 0.25 | C10 | 0.07 | 0.09 | 0.1 | D10 | 1 | 0.97 | 0.75 |
| A11 | 0.35 | 0.28 | 0.29 | B11 | 0.54 | 0.28 | 0.17 | C11 | 0.27 | 0.32 | 0.08 | D11 | 0.43 | 0.36 | 0.35 |
| A12 | 0.06 | 0.06 | 0.06 | B12 | 0.24 | 0.35 | 0.21 | C12 | 0.11 | 0.04 | 0.03 | D12 | 0.9 | 0.98 | 0.04 |
| A13 | 0.24 | 0.24 | 0.24 | B13 | 0.54 | 0.21 | 0.21 | C13 | 0.16 | 0.15 | 0.21 | D13 | 0.96 | 0.8 | 0.07 |
| A14 | 0.15 | 0.4 | 0.44 | B14 | 0.31 | 0.18 | 0.13 | C14 | 0.05 | 0.04 | 0.05 | D14 | 0.53 | 0.87 | 0.47 |
| A15 | 0.5 | 0.45 | 0.43 | B15 | 0.3 | 0.38 | 0.3 | C15 | 0.08 | 0.07 | 0.09 | D15 | 0.42 | 0.35 | 0.73 |
| A16 | 0.06 | 0.06 | 0.08 | B16 | 0.37 | 0.37 | 0.3 | C16 | 0.18 | 0.3 | 0.58 | D16 | 0.98 | 0.94 | 0.82 |
| A17 | 0.5 | 0.67 | 0.15 | B17 | 0.3 | 0.26 | 0.24 | C17 | 0.03 | 0.04 | 0.05 | D17 | 0.3 | 0.32 | 0.31 |
| A18 | 0.13 | 0.09 | 0.11 | B18 | 0.74 | 0.48 | 0.63 | C18 | 0.06 | 0.06 | 0.07 | D18 | 0.32 | 0.34 | 0.37 |
| A19 | 0.21 | 0.38 | 0.2 | B19 | 0.35 | 0.2 | 0.45 | C19 | 0.08 | 0.07 | 0.06 | D19 | 0.27 | 0.32 | 0.44 |
| A20 | 0.58 | 0.52 | 0.24 | B20 | 0.26 | 0.26 | 0.12 | C20 | 0.04 | 0.05 | 0.05 | D20 | 0.61 | 0.71 | 0.55 |

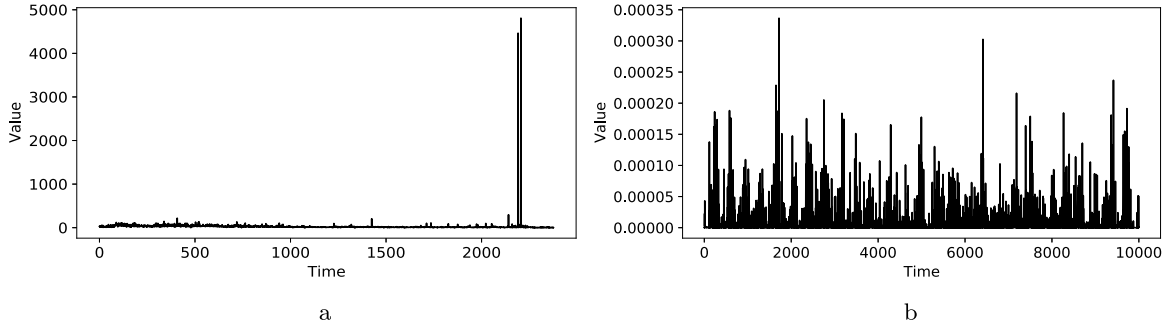



**Fig. 25.** The original versions of time series B8 (a) and C2 (b).

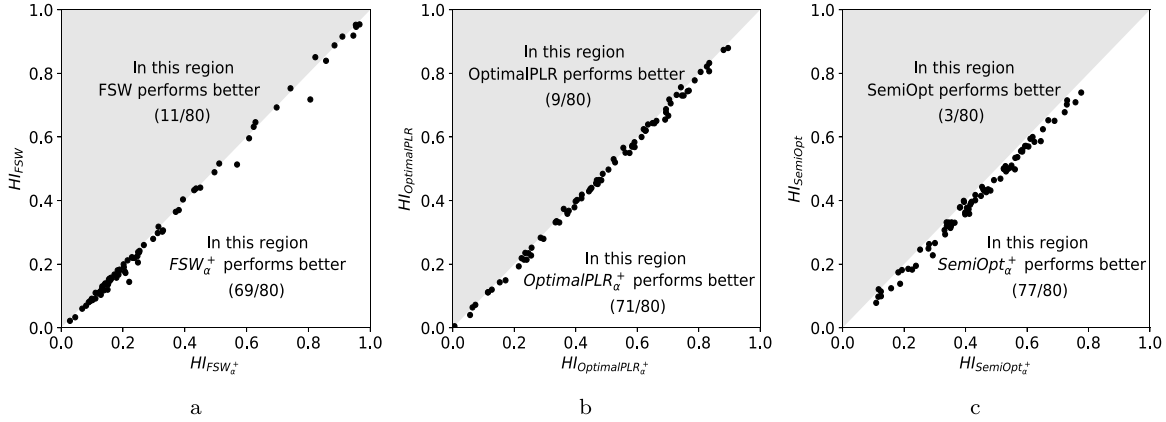



**Fig. 26.** Comparison between $HI_{method_\alpha^+}$ and $HI_{method}$ on all datasets.

## 6. Conclusions

We introduced PLA with adaptive error bounds (AEPLA) – a time-series representation framework that allocates dynamic error bounds based on the fluctuation of the original time series in different temporal stages. We implemented AEPLA based on different types of $l_\infty$-PLA methods and tested it on a broad range of real-world datasets. Our experimental results demonstrate the superiority of AEPLA compared to fixing the error bound from three aspects: (i) compared to original $l_\infty$-PLA baselines, we achieved a lower average representation error per segment, and thus, more features of the fluctuating time series could be maintained under high compression ratio (the global error bound greater than 0.07 in general); (ii) compared to these baselines, a lower representation error under the same number of segments can be achieved, which points to a more reasonable trade-off between the approximation error and the compression ratio; (iii) the proposed framework can yield a more compact representation of the original time series, leading to lower average time and space complexity; Furthermore, the optimisation of the scaling factor $\alpha$ in AEPLA can reduce the representation error when achieving the same compression ratio. The performance of AEPLA is highly dependent on the global error-bound hyperparameter and one possible solution for optimising this

**Table 20**
Benchmark datasets used in the experiment.

| The list of the datasets | | |
|---|---|---|
| Physics (Astrophysics) | | |
| No. | Data source | Name |
| A1 | SPIDR Solar Data | SPIDR L10900 Standard deviation std |
| A2 | SPIDR HPI DMSP | SPIDR hpidmsp L13300 F08 meas |
| A3 | SPIDR HPI NOAA | SPIDR hpinoaa L11700 NOAA12 power |
| A4 | SPIDR Geomagnetic annual means Ionosphere | SPIDR ionosph foF2 Capetown CT13M L16500 |
| A5 | SPIDR Interplanetary Magnetic Field | SPIDR by WIND L13300 minly |
| A6 | SPIDR Solar Data | SPIDR L12600 Number of observations nobs |
| A7 | SPIDR HPI NOAA | SPIDR hpinoaa L12000 NOAA16 power |
| A8 | SPIDR Interplanetary Magnetic Field | SPIDR by WIND L8200 minly |
| A9 | SPIDR Solar Data | SPIDR L13000 Number of solar spots nspots |
| A10 | SPIDR Geomagnetic annual means Ionosphere | SPIDR ionosph foF2 Capetown CT13M L19000 |
| A11 | SPIDR Interplanetary Magnetic Field | SPIDR bz ACE L10700 minly |
| A12 | SPIDR HPI DMSP | SPIDR hpidmsp L13600 F12 meas |
| A13 | SPIDR Solar Data | SPIDR L15500 Number of observations nobs |
| A14 | SPIDR Geomagnetic annual means Ionosphere | SPIDR ionosph foF2 Moscow MO155 L16400 |
| A15 | SPIDR Interplanetary Magnetic Field | SPIDR bz ACE L11200 minly |
| A16 | SPIDR HPI NOAA | SPIDR hpinoaa L13800 NOAA12 power |
| A17 | SPIDR Geomagnetic | SPIDR geomag L16900 DST |
| A18 | SPIDR Solar Data | SPIDR L15800 Number of solar spots nspots |
| A19 | SPIDR Interplanetary Magnetic Field | SPIDR bz ACE L11300 minly |
| A20 | SPIDR Geomagnetic | SPIDR geomag L6900 DST |
| Finance (High-low pricing) | | |
| B1 | Yahoo Finance | FI yahoo HL GDAXI |
| B2 | Yahoo Finance | FI yahoo HL GSPC |
| B3 | Yahoo Finance | FI yahoo HL HSI |
| B4 | Yahoo Finance | FI yahoo HL IBEX |
| B5 | Yahoo Finance | FI yahoo HL IIX |
| B6 | Yahoo Finance | FI yahoo HL IPC |
| B7 | Yahoo Finance | FI yahoo HL IRX |
| B8 | Yahoo Finance | FI yahoo HL ISCI |
| B9 | Yahoo Finance | FI yahoo HL ISEQ |
| B10 | Yahoo Finance | FI yahoo HL ITEQ |
| B11 | Yahoo Finance | FI yahoo HL IXBK |
| B12 | Yahoo Finance | FI yahoo HL IXFN |
| B13 | Yahoo Finance | FI yahoo HL IXIC |
| B14 | Yahoo Finance | FI yahoo HL IXID |
| B15 | Yahoo Finance | FI yahoo HL IXIS |
| B16 | Yahoo Finance | FI yahoo HL IXK |
| B17 | Yahoo Finance | FI yahoo HL IXTR |
| B18 | Yahoo Finance | FI yahoo HL IXUT |
| B19 | Yahoo Finance | FI yahoo HL JKSE |
| B20 | Yahoo Finance | FI yahoo HL KLSE |
| Meteorology (Precipitation rate) | | |
| C1 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.33 15 |
| C2 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.21 21 |
| C3 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.60 9 |
| C4 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.69 27 |
| C5 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.87 33 |
| C6 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.40 29 |
| C7 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.25 9 |
| C8 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.44 37 |
| C9 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.37 23 |
| C10 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.83 15 |
| C11 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.74 47 |
| C12 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.18 15 |
| C13 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.13 25 |
| C14 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.79 37 |
| C15 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.96 1 |
| C16 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.64 47 |
| C17 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.51 41 |
| C18 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.91 21 |
| C19 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.2 13 |
| C20 | Precipitation rate, NCEP/NCAR, CRU | CM prate.sfc.gauss.1948–2007.d.qs eurasia.48 45 |
| Finance (Trade-volume) | | |
| D1 | Yahoo Finance Shares | FI yahoo V CPW.L |
| D2 | Yahoo Finance Shares | FI yahoo V CUK.L |
| D3 | Yahoo Finance Shares | FI yahoo V CW.L |
| D4 | Yahoo Finance Shares | FI yahoo V DCG.L |
| D5 | Yahoo Finance Shares | FI yahoo V DEB.L |

(*continued on next page*)

**Table 20** (*continued*).

| | | |
|---|---|---|
| D6 | Yahoo Finance Shares | FI yahoo V DLN.L |
| D7 | Yahoo Finance Shares | FI yahoo V DNO.L |
| D8 | Yahoo Finance Shares | FI yahoo V DOM.L |
| D9 | Yahoo Finance Shares | FI yahoo V EFM.L |
| D10 | Yahoo Finance Shares | FI yahoo V ENRC.L |
| D11 | Yahoo Finance Shares | FI yahoo V EVG.L |
| D12 | Yahoo Finance Shares | FI yahoo V EZJ.L |
| D13 | Yahoo Finance Shares | FI yahoo V FCCN.L |
| D14 | Yahoo Finance Shares | FI yahoo V FP.L |
| D15 | Yahoo Finance Shares | FI yahoo V FTO.L |
| D16 | Yahoo Finance Shares | FI yahoo V GEMD.L |
| D17 | Yahoo Finance Shares | FI yahoo V GMG.L |
| D18 | Yahoo Finance Shares | FI yahoo V GNK.L |
| D19 | Yahoo Finance Shares | FI yahoo V GNS.L |
| D20 | Yahoo Finance Shares | FI yahoo V GRG.L |

hyperparameter is to consider its impact on a downstream task. Time-series representation can essentially be considered a pre-processing step within a machine learning pipeline based on time-series data. In future work, the impact of this step on a variety of downstream machine learning tasks and whether this hyperparameter can be optimised within a full pipeline will be investigated, considering the downstream task performance along with space and representation quality constraints. Furthermore, to expand AEPLA's capabilities, consolidating the applicability of AEPLA for multivariate time series or spatiotemporal data representation with multiple PLA methods can be proposed to replace the fixed PLA method.

## CRediT authorship contribution statement

**Zhou Zhou:** Conceptualization, Methodology, Software, Writing – original draft. **Mitra Baratchi:** Formal analysis, Investigation. **Gangquan Si:** Project administration, Funding acquisition. **Holger H. Hoos:** Resources, Supervision. **Gang Huang:** Validation, Supervision.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Data will be made available on request.

## Acknowledgements

This work is supported in part by the National Key Research and Development Program (2018AAA0101501), the National Natural Science Foundation of China (Grant No:61304118) and by TAILOR, a project funded by EU Horizon 2020 research and innovation programme under Grant No. 952215.

## Appendix

See Table 20.

## References

Agrawal, Rakesh, Faloutsos, Christos, Swami, Arun, 1993. Efficient similarity search in sequence databases. In: Foundations of Data Organization and Algorithms: 4th International Conference, FODO'93 Chicago, Illinois, USA, October 13–15, 1993 Proceedings 4. Springer, pp. 69–84.

Carmona-Poyato, Ángel, Fernández-García, Nicolás Luis, Madrid-Cuevas, Francisco José, Durán-Rosal, Antonio Manuel, 2020. A new approach for optimal time-series segmentation. Pattern Recognit. Lett. 135, 153–159.

Carmona-Poyato, Ángel, Fernández-Garcia, Nicolás Luis, Madrid-Cuevas, Francisco José, Durán-Rosal, Antonio Manuel, 2021. A new approach for optimal offline time-series segmentation with error bound guarantee. Pattern Recognit. 115, 107917.

Chen, Qiuxia, Chen, Lei, Lian, Xiang, Liu, Yunhao, Yu, Jeffrey Xu, 2007. Indexable PLA for efficient similarity search. In: Proceedings of the 33rd International Conference on Very Large Data Bases. pp. 435–446.

Chen, Yingjun, Hao, Yijie, 2020. A novel framework for stock trading signals forecasting. Soft Comput. 24 (16), 12111–12130.

Chen, Yao, Wang, Xiao, Jung, Yonghan, Abedi, Vida, Zand, Ramin, Bikak, Marvi, Adibuzzaman, Mohammad, 2018. Classification of short single-lead electrocardiograms (ECGs) for atrial fibrillation detection using piecewise linear spline and XGBoost. Physiol. Meas. 39 (10), 104006.

Deng, Dan, Li, Bo, 2022. An online piecewise linear representation method for hydraulic fracturing time series. Chem. Technol. Fuels Oils 58 (2), 391–402.

Durán-Rosal, Antonio M, Gutiérrez, Pedro A, Carmona-Poyato, Angel, Hervás-Martínez, César, 2019. A hybrid dynamic exploitation barebones particle swarm optimisation algorithm for time series segmentation. Neurocomputing 353, 45–55.

Ehrgott, Matthias, 2012. Vilfredo Pareto and multi-objective optimization. Doc. Math. 447–453.

Elmeleegy, Hazem, Elmagarmid, Ahmed K., Cecchet, Emmanuel, Aref, Walid G., Zwaenepoel, Willy, 2009. Online piece-wise linear approximation of numerical streams with precision guarantees. Proc. VLDB Endow. 2 (1), 145–156.

Emmerich, Michael T.M., Deutz, André H., 2018. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. Nat. Comput. 17, 585–609.

Fu, Tak-chung, 2011. A review on time series data mining. Eng. Appl. Artif. Intell. 24 (1), 164–181.

Gritzali, F., Papakonstantinou, G., 1983. A fast piecewise linear approximation algorithm. Signal Process. 5 (3), 221–227.

Hakimi, S. Louis, Schmeichel, Edward F., 1991. Fitting polygonal functions to a set of points in the plane. CVGIP: Graph. Models Image Process. 53 (2), 132–136.

Hu, Yupeng, Guan, Peiyuan, Zhan, Peng, Ding, Yiming, Li, Xueqing, 2018. A novel segmentation and representation approach for streaming time series. IEEE Access 7, 184423–184437.

Hu, Yupeng, Ji, Cun, Zhang, Qingke, Chen, Lin, Zhan, Peng, Li, Xueqing, 2020. A novel multi-resolution representation for time series sensor data analysis. Soft Comput. 24 (14), 10535–10560.

Hutter, Frank, Hoos, Holger H., Leyton-Brown, Kevin, 2011. Sequential model-based optimization for general algorithm configuration. In: Learning and Intelligent Optimization: 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers 5. Springer, pp. 507–523.

Ishibuchi, Hisao, Doi, Ken, Nojima, Yusuke, 2017. On the effect of normalization in MOEA/D for multi-objective and many-objective optimization. Complex Intell. Syst. 3 (4), 279–294.

Ji, Cun, Liu, Shijun, Yang, Chenglei, Wu, Lei, Pan, Li, Meng, Xiangxu, 2016. A piecewise linear representation method based on importance data points for time series data. In: 2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design. CSCWD, IEEE, pp. 111–116.

Keogh, Eamonn, Chu, Selina, Hart, David, Pazzani, Michael, 2004. Segmenting time series: A survey and novel approach. In: Data Mining in Time Series Databases. World Scientific, pp. 1–21.

Keogh, Eamonn, Smyth, Padhraic, 1997. A probabilistic approach to fast pattern matching in time series databases. In: Proceedings of the Third International Conference on Knowledge Discovery and Data Mining. KDD '97, AAAI Press, pp. 24–30.

Lin, Jeng-Wei, Liao, Shih-wei, Leu, Fang-Yie, 2020. A novel bounded-error piecewise linear approximation algorithm for streaming sensor data in edge computing. In: Advances in Intelligent Networking and Collaborative Systems: The 11th International Conference on Intelligent Networking and Collaborative Systems (INCoS-2019). Springer, pp. 123–132.

Lindauer, Marius, Eggensperger, Katharina, Feurer, Matthias, Falkner, Stefan, Biedenkapp, André, Hutter, Frank, 2017. SMAC v3: Algorithm configuration in Python. https://github.com/automl/SMAC3.

Lindstrom, Peter, 2014. Fixed-rate compressed floating-point arrays. IEEE Trans. Vis. Comput. Graphics 20 (12), 2674–2683.

Liu, Xiaoyan, Lin, Zhenjiang, Wang, Huaiqing, 2008. Novel online methods for time series segmentation. IEEE Trans. Knowl. Data Eng. 20 (12), 1616–1626.

Liu, Jiajun, Zhao, Kun, Sommer, Philipp, Shang, Shuo, Kusy, Brano, Lee, Jae-Gil, Jurdak, Raja, 2016. A novel framework for online amnesic trajectory compression in resource-constrained environments. IEEE Trans. Knowl. Data Eng. 28 (11), 2827–2841.

Lovrić, Miodrag, Milanović, Marina, Stamenković, Milan, 2014. Algoritmic methods for segmentation of time series: An overview. J. Contemp. Econ. Bus. Issues 1 (1), 31–53.

Luo, Wei, Li, Yongqi, Yao, Fubin, Wang, Shaokun, Li, Zhen, Zhan, Peng, Li, Xueqing, 2021. Multi-resolution representation for streaming time series retrieval. Int. J. Pattern Recognit. Artif. Intell. 35 (6), 2150019.

Luo, Ge, Yi, Ke, Cheng, Siu-Wing, Li, Zhenguo, Fan, Wei, He, Cheng, Mu, Yadong, 2015. Piecewise linear approximation of streaming time series data with max-error guarantees. In: 2015 IEEE 31st International Conference on Data Engineering. IEEE, pp. 173–184.

Mishra, Kakuli, Basu, Srinka, Maulik, Ujjwal, 2022. Graft: A graph based time series data mining framework. Eng. Appl. Artif. Intell. 110, 104695.

Mori, Taketoshi, Nejigane, Yu, Shimosaka, Masamichi, Segawa, Yushi, Harada, Tatsuya, Sato, Tomomasa, 2005. Online recognition and segmentation for time-series motion with hmm and conceptual relation of actions. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 3864–3870.

Oliver, Jonathan J., Baxter, Rohan A., Wallace, Chris S., 1998. Minimum message length segmentation. In: Research and Development in Knowledge Discovery and Data Mining: Second Pacific-Asia Conference, PAKDD-98 Melbourne, Australia, April 15–17, 1998 Proceedings 2. Springer, pp. 222–233.

O'Rourke, Joseph, 1981. An on-line algorithm for fitting straight lines between data ranges. Commun. ACM 24 (9), 574–578.

Pang, Yong, Shi, Maolin, Zhang, Liyong, Sun, Wei, Song, Xueguan, 2022. A multivariate time series segmentation algorithm for analyzing the operating statuses of tunnel boring machines. Knowl.-Based Syst. 242, 108362.

Popivanov, Ivan, Miller, Renee J., 2002. Similarity search over time-series data using wavelets. In: Proceedings 18th International Conference on Data Engineering. IEEE, pp. 212–221.

Salotti, Marc, 2002. Optimal polygonal approximation of digitized curves using the sum of square deviations criterion. Pattern Recognit. 35 (2), 435–443.

Si, Yain-Whar, Yin, Jiangling, 2013. OBST-based segmentation approach to financial time series. Eng. Appl. Artif. Intell. 26 (10), 2581–2596.

Wu, Chi-Jen, Zeng, Wei-Sheng, Ho, Jan-Ming, 2021. Optimal segmented linear regression for financial time series segmentation. In: 2021 International Conference on Data Mining Workshops. ICDMW, IEEE, pp. 623–630.

Xie, Qing, Pang, Chaoyi, Zhou, Xiaofang, Zhang, Xiangliang, Deng, Ke, 2014. Maximum error-bounded piecewise linear representation for online stream approximation. VLDB J. 23, 915–937.

Xu, Zhenghua, Zhang, Rui, Kotagiri, Ramamohanarao, Parampalli, Udaya, 2012. An adaptive algorithm for online time series segmentation with error bound guarantee. In: Proceedings of the 15th International Conference on Extending Database Technology. pp. 192–203.

Zhan, Peng, Hu, Yupeng, Chen, Lin, Luo, Wei, Li, Xueqing, 2021. Spar: Set-based piecewise aggregate representation for time series anomaly detection. Sci. China Inf. Sci. 64, 1–3.

Zhan, Peng, Sun, Changchang, Hu, Yupeng, Luo, Wei, Zheng, Jiecai, Li, Xueqing, 2020. Feature-based online representation algorithm for streaming time series similarity search. Int. J. Pattern Recognit. Artif. Intell. 34 (5), 2050010.

Zhang, Jieyu, Li, Miqing, Liu, Weibo, Lauria, Stanislao, Liu, Xiaohui, 2022. Many-objective optimization meets recommendation systems: A food recommendation scenario. Neurocomputing 503, 109–117.

Zhao, Kai, Di, Sheng, Dmitriev, Maxim, Tonellot, Thierry-Laurent D., Chen, Zizhong, Cappello, Franck, 2021. Optimizing error-bounded lossy compression for scientific data by dynamic spline interpolation. In: 2021 IEEE 37th International Conference on Data Engineering. ICDE, IEEE, pp. 1643–1654.

Zhao, Huanyu, Dong, Zhaowei, Li, Tongliang, Wang, Xizhao, Pang, Chaoyi, 2016. Segmenting time series with connected lines under maximum error bound. Inform. Sci. 345, 1–8.

Zhao, Huanyu, Li, Tongliang, Chen, Genlang, Dong, Zhaowei, Bo, Mengya, Pang, Chaoyi, 2020. An online PLA algorithm with maximum error bound for generating optimal mixed-segments. Int. J. Mach. Learn. Cybern. 11 (7), 1483–1499.

Zhao, Huanyu, Pang, Chaoyi, Kotagiri, Ramamohanarao, Pang, Christopher K., Deng, Ke, Yang, Jian, Li, Tongliang, 2022. An optimal online semi-connected PLA algorithm with maximum error bound. IEEE Trans. Knowl. Data Eng. 34 (1), 164–177.