Multi-Window-Finder: Domain Agnostic Window Size for Time Series Data

Shima Imani
Microsoft Corporation
shimaimani@microsoft.com

Alireza Abdoli
University of California, Riverside
aabdo002@ucr.edu

Ali Beyram
VideoAmp
ali@videoamp.com

Azam Imani
Independent Researcher
i.azam48@gmail.com

Eamonn Keogh
University of California, Riverside
eamonn@cs.ucr.edu

ABSTRACT

Finding the right window size has always been a challenging task for many domains in data mining such as classification, clustering, motif discovery, anomaly detection, and time series prediction. The window size parameter is mostly given by experts or is based on domain knowledge. Failure to provide an appropriate window size may result in poor outcomes in underlying time series data mining tasks. In this work, we present Multi-Window-Finder, a domain agnostic algorithm that can find different window sizes for various behaviors in the time series data. Our algorithm uses neighborhood information to find the right window size. We will demonstrate the utility and performance of our algorithm on diverse case studies and experiments, including a large collection of 250 datasets as part of the KDD Cup.

KEYWORDS

time series, window size, multi window sizes, subsequence length

ACM Reference Format:
Shima Imani, Alireza Abdoli, Ali Beyram, Azam Imani, and Eamonn Keogh. 2021. Multi-Window-Finder: Domain Agnostic Window Size for Time Series Data. In *MileTS '21: 7th KDD Workshop on Mining and Learning from Time Series, August 14th, 2021, Singapore.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Many time series analysis tasks require a window size as an input of their underlying algorithms, including but not limited to classification, clustering, motif discovery, anomaly detection, chain discovery, segmentation, time series prediction, etc.

Most recently, Matrix Profile has emerged as the state-of-the-art framework for finding time series motifs and anomaly detection. Time series motifs are approximately repeated patterns in real-valued data [3] [7] and anomaly detection refers to finding rare and occasional events that are significantly different compared to majority of the data [8]. Finding motifs and anomalies in the time series are useful for exploratory data mining tasks and the window size would be the single input parameter to achieve this task.

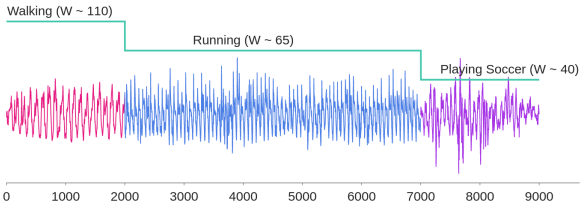

Figure 1: The activities of a soccer player. The player walks for a period, then proceeds with running and playing soccer. The green vector represents the ground truth. Note that the corresponding window size is shown next to each behavior in parentheses.

Another area that the window size could play an important role is in whole time series clustering. Given a set of individual time series data, the objective is to group similar time series into the same cluster [6][9]. Here the individual items are assumed to have been extracted and typically normalized to have the same length. Extracting these individual time series data, need the knowledge of the right window size.

Similarly time series chains use Matrix Profile as a subroutine in its algorithm. As we mentioned earlier, Matrix profile takes one parameter which is the window size. Most of the time this parameter is given by an expert or domain knowledge is needed for tuning it. A poor choice of window size parameter can result in failing to find meaningful insights in the data.

Also, in some cases providing a variable window size can increase the performance of these algorithms and generates more accurate results. The authors [4] emphasized on the disadvantage of providing a fixed window size as "*discords are limited (because) a fixed length must be specified in advance, making it a clearly sub-optimal approach for applications dealing with climate data events of varying length*".

In this work, we introduce Multi-Window-Finder algorithm that can find different window lengths for time series data. Fig .1 shows a time series that represents the training session of a soccer player. The player initially warms up with exercises such as walking and running, and then she starts to play soccer. This time series consists of three different behaviors, each of which with different window lengths. The Multi-Window-Finder algorithm is able to find all three different window lengths as will be discussed later in Section 4.2.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
MileTS '21, August 14th, 2021, Singapore
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

We organize the rest of the paper as follows: in Section 2, we introduce the necessary notation and definitions. Section 3 introduces related works. We explain our work, in Section 4. In Section 5, we perform a comprehensive empirical evaluation. Section 6 draws conclusions and suggests directions for future work.

2 DEFINITIONS AND NOTATION

We begin by describing the necessary notation and definitions. The data type of interest is *time series*:

Definition 1 (Time series): A time series T of length n is a sequence of real-valued numbers: $t_i : T = t_1, t_2, \ldots, t_n$.

A local region of time series is called a *subsequence*:

Definition 2 (Subsequence): A subsequence $T_{i,m}$ of a time series T is a continuous ordered subset of the values from T of length m starting from position i. $T_{i,m} = t_i, t_{i+1}, \ldots, t_{i+m-1}$, where $1 \leq i \leq n - m + 1$.

Definition 3 (Moving mean): A moving mean is a calculation used to analyze data points by creating a series of averages of different subsequences in the time series. Most of the time, moving mean is used to mitigate the effects of small fluctuations in the time series data.

3 RELATED WORK

Many data mining tasks require window size which ought to be provided based on expert knowledge or field domain. Given the fact that window size is a crucial parameter towards optimal performance of such tasks, here we briefly discuss some of the existing research on time series window size calculation.

Madrid et al. [7] introduced Pan Matrix Profile (PMP), for all subsequences of all lengths, which allows for parameter-free motif discovery. Note, this algorithm uses an upper bound for the window size. In classic motif discovery algorithm, the user can run the matrix profile to find the pattern. However, matrix profile needs a window size parameter. Pan Matrix Profile runs the matrix profile for all different window sizes and since it computes the Pearson correlation, the algorithm can compare patterns of different window sizes to each other. Then it finds the motif that has highest Pearson correlation.

Also Gao et al. [5] introduced a series of tools to allow interactive discovery of variable-length time series motifs. However, this work is different from PMP. Firstly, because they use a discretized representation of the data (for efficiency), they end up finding approximate answers. Secondly, they only return information about a small subset of the patterns, whereas PMP contains exact distances for all subsequences of all lengths.

In a similar effort, [8] proposed a variant of Matrix Profile, namely MERLIN, which is suited at finding discords (i.e. subsequences of a time series that are maximally far away from their nearest neighbors) of arbitrary length in the time series.

It is worthwhile to mention the Fast Fourier Transform (FFT) can be used to find optimal window size. The FFT converts the signal from time domain into the frequency domain and vice versa. However, FFT generates a poor result when you have time-variant and non-stationary data or when working with sharp corners. As we show later for some of the time series data due to the existence of spikes, dropouts or noise FFT algorithm would generate poor results.

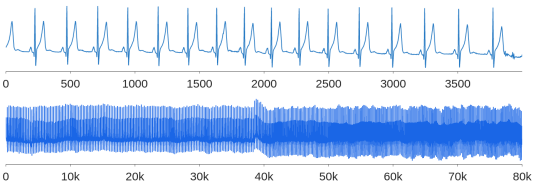

Figure 2: top) The zoom in version of the time series data bottom) The time series corresponds to Arterial Blood Pressure.

4 FINDING WINDOW SIZES OF TIME SERIES

In this section we introduce Multi-Window-Finder algorithm, which allows us to find all window lengths relevant to the time series data. The assumption of this algorithm is that each behavior is repeated for couple of times.

One might say that through asking an expert or domain knowledge might be enough to find the right window size. However, human error can never undermined. Moreover, in many cases where we have a steam of data or a lot of time series, asking an expert or domain knowledge is not a scalable solution. Note that our algorithm is very efficient and fast, which can automate the burdensome and time-consuming task of determining window size.

4.1 Multi-Window-Finder

The intuition behind this algorithm is that the error towards moving average for any given window size will decrease as we get closer to the actual window size of time series, and increases as we get farther.

To make this clear, let us explain through an example. Suppose we have a time series of Arterial Blood Pressure (ABP) of a patient, as in Fig. 2 (bottom). A snippet of the Arterial Blood Pressure time series is shown in Fig. 2 (top).

Let us compute the moving average of this time series for different window sizes. Fig. 3 shows the small section of the moving averages for different window sizes. The black dashed line is the mean value of these moving averages. From the left subplots we can see that as we increase the window size of the moving average, the moving average distance to the mean (black dashed line) decreases, however, in the right subplots as we increase the window size the moving average distance to the mean increases. The right window size is the one that the moving average has the lowest distance to the mean.

From this observation we can compute the time series actual window size. First we compute the moving average for different window sizes and then we compute the distance of these moving averages from the total mean of moving average for each window which we call it moving-dist meta time series, as shown in Fig. 4. Next, we locate the first valley of the moving-dist meta time series, which represents the minimum window size for the time series data. Note that the next valley is almost two times bigger in size compared with the first valley and the third valley is three times bigger than the first valley and so on. Using this information, we can compute the confidence interval of the window size. First we compute the error using the standard deviation of these windows and then compute the confidence interval using error. For this time series the window size is 240 with the confidence of % 96.

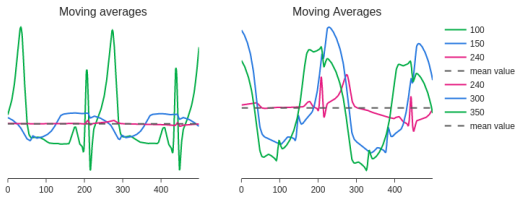

Figure 3: left) The moving average corresponding to window sizes 100, 150 and 240. The black dashed line is the mean of the moving average. Right) The moving average corresponding to window sizes of 240,300 and 350. The black dashed line is the mean of the moving average.

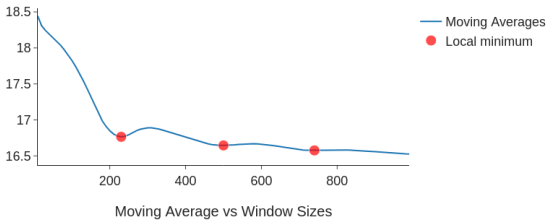

Figure 4: Moving-dist corresponding to the time series of arterial blood pressure for different window sizes. The red dots are the local minimums.

The main algorithm for finding window size is outlined in Algorithm 1, and its subroutine that computes the moving average for different window sizes is outlined in Algorithm 2. In Algorithm 1, we compute all the moving averages and then we compute the residuals which would be the absolute distance of moving average from its mean value. After computing residuals, we compute local minimums and subsequently compute the window size and confidence interval. For simplicity in Algorithm 1 we set the parameter 'e' which is the maximum length of window size to some large value (e.g. 1000 here). In general, we continue increasing the window size until we find at least three local minimum as shown in Fig. 3, which means this algorithm is parameter free.

4.2 Time series with different window sizes

Often the time series consists of different behaviors. For example, most of the time series that is generated by human activities consist of different activities such as running, walking, cycling and etc. A data analyst might ask: *How can we find window size for such a time series?* In this section, we expand our method of finding one window size to find a meta-time series that represents a variable window size for each section of time series.

For illustration of our method, we use PAMAP physical activity monitoring dataset [1]; which contains a wide range of everyday household and fitness activities performed by a number of human subjects. The data is generated by subjects wearing 3D inertial measurement units (IMUs) and a heart rate (HR) monitor. This dataset has been used for many other tasks such as activity recognition, segmentation, feature extraction, and classification.

Algorithm 1 Multi-Window-Finder

Input: time series T
Output: window size w, confidence c

$s \leftarrow 10, e \leftarrow 1000, ws \leftarrow empty$
for $w \leftarrow s$ to e **do**
    MA=moving-avg(T, w) //Algroithm 2
    moving-dist$\leftarrow Sum(Log(abs(\text{MA}-mean(\text{MA})))$
    $ws \leftarrow w$
end for
local-min=diff(sign(diff(moving-dist))) > 0).nonzero()[0] + 1
for i in local-min **do**
    $res \leftarrow ws[i]/(i+1)$
end for
$w = mean(\text{res})$
$c = 1 - std(\text{res})$
return w, c

Algorithm 2 Moving Average

Input: time series T, window-size w
Output: moving average moving-avg

moving-avg=cumsum[T] //cumulative sum
moving-avg$[w:]$ = moving-avg$[w:]$ − moving-avg$[:w]$
$moving-avg$ = moving-avg$[w-1:]/w$
return moving-avg

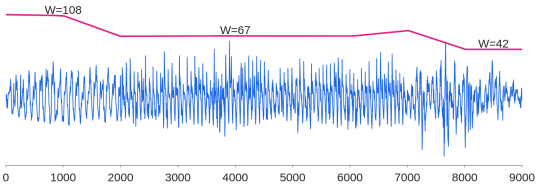

Figure 5: The meta-time series (pink) generated by Multi-Window-Finder algorithm which corresponds to variable window sizes of time series. Time series corresponding to walking, running and playing soccer behaviors (blue).

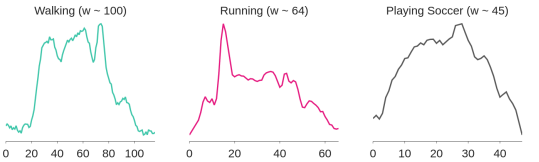

Figure 6: Three instances of (left) walking, (center) running and (right) playing soccer behaviors.

Consider a time series that represents training session of a soccer player, as in Fig .1. The player initially warms up with exercises such as walking and running, and then starts to play soccer. Fig .5 (bottom) shows a time series consisted of walking, running and finally playing soccer from a hip-mounted accelerometer collected at 100 HZ. To find the best window size for this time series, we will run Algorithm 1 for small batches of this time series. This can help us to generate a meta-time series that represents window size for each section of time series. The result is shown in Fig .5 (red). As

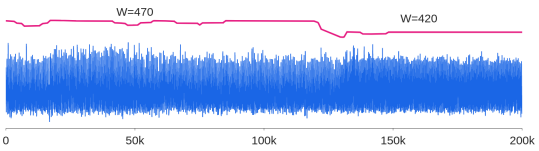

Figure 7: bottom) Time series data. top) The red vector represents the results of Multi-Window-Finder algorithm.

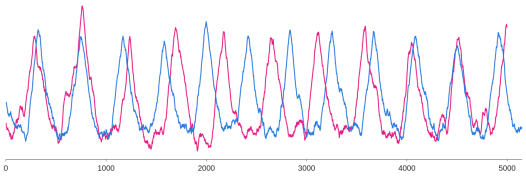

Figure 8: The blue snippet corresponds to the position of 1000 and the red snippet corresponds to the position of 150k of the original time series as shown in Fig. 7

you can see this time series has a different window size for each behavior. To compare our results with the ground truth, we extract one generic pattern from each behavior as shown in Fig .6. As you can see, our results are very close to the value of the ground truth window sizes. The capability of finding different window sizes can help us to find the better quality motifs, discords and subsequently do better with other data mining tasks.

Let us consider another example. Suppose a user wants to analyze the time series of Fig. 7. The first question the user might ask is *What is the window size of this time series so I can run motif discovery algorithm?* By randomly sampling parts of this time series, the user learns that this time series consist of one behavior. However, If the user runs Multi-Window-Finder algorithm, surprisingly it shows two different window sizes for this time series.

Fig. 8 shows the zoomed in version of this time series from two different regions. The red region belongs to the beginning of the time series (staring position at 1000) and the blue snippet belongs to the last part (starting position at 150k). For visualization purposes, Fig. 8 shows this two snippets on top of each other. For the period of 5000 points, one behavior in the red snippet appears ten times however the blue behavior repeats twelve times which means there are two different window sizes. The behavior of this time series has changed after some time. Note, this information provides the user with better insight of time series and this user might investigate the reason for this change.

4.3 Time Complexity

For computing the window size, we need to compute moving mean of time series which takes $O(n)$ where n is the length of time series. Given that we are running the algorithm for different window sizes, the complexity is $O(kn)$. Note, we are assuming the window size is small in comparison to n which means k is small.

5 EXPERIMENTAL EVALUATION

In this section, we demonstrate performance and superiority of Multi-Window-Finder through experiments. To ensure that our experiments are easy to reproduce, we have created a website that contains all

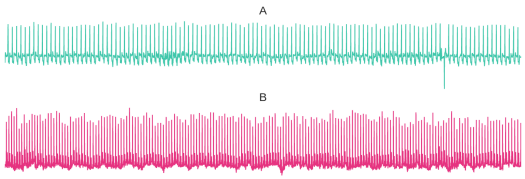

Figure 9: Two snippets of time series dataset from KDD cup. top) data with drop out bottom) data with noise.

code and materials for all experiments [2]. We used the time series collection of KDD Cup 2021 that contains 250 time series datasets. This collection contains a variety of time series related to different domains such as human behavior and health domain. Although the collection in the KDD Cup is aimed at detecting anomaly, we will use this collection for the purpose of finding window size which is needed for most anomaly detection algorithms.

Fig .9 shows a snippet of time series for a few datasets in this collection. We ran FFT algorithm as well as Multi-Window-Finder algorithm to find the window size as shown in Table 1. As we can see from the results, the existence of noise or spikes negatively impacts the output of FFT algorithm and produces inaccurate results.

Table 1: The performance of FFT and Multi-Window-Finder algorithms.

Time Series	Ground Truth	FFT	Multi-Window-Finder
A	150	83	161
B	100	930	105

We further asked an expert in the time series domain to label window size of all the time series in this collection. This can help us to compare the results of Multi-Window-Finder algorithm against the ground truth. We measure the performance of Multi-Window-Finder algorithm using the following formula.

$$success\_rate = \begin{cases} 1 & \frac{abs(\text{Window}_{\text{ground truth}} - \text{Window})}{\text{Window}_{\text{ground truth}}}) \leq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

For the success rate, if the output of our Multi-Window-Finder is close to the ground truth window, then we count it as success, otherwise we count it as no success. Then we sum up all successes and report the total success for the whole collection. The results are shown in Table 2.

Table 2: Total success rate of FFT and Multi-Window-Finder algorithms.

KDD cup dataset	FFT	Multi-Window-Finder
250 time series	171	190

As you can see the results are very promising. Out of 250 time series, in 190 of these time series Multi-Window-Finder algorithm is accurate. We further investigated the discrepancy between the results of Multi-Window-Finder algorithm and the ground truth. For some of the cases, the time series has more than one window size which reporting just one was not ideal. In other cases, the Multi-Window-Finder result is twice bigger than the ground truth and we do not know what the correct size for these time series.

6 DISCUSSION AND CONCLUSIONS

We have introduced Multi-Window-Finder algorithm which enables the user to find an appropriate window size for the time series. In addition, the expanded version of this algorithm is able to find a variable window size for time series data. Window size is a parameter that is needed for many of the algorithms in data mining discovery analysis and in many cases it is given by an expert of the domain. Using Multi-Window-Finder algorithm we are able to make these algorithms, parameter-free. In future work, we want to expand this algorithm to handle multi dimensional time series data and test on more dataset of time series data.

REFERENCES

[1] Barbara E Ainsworth, William L Haskell, Melicia C Whitt, Melinda L Irwin, Ann M Swartz, Scott J Strath, WILLIAM L O Brien, David R Bassett, Kathryn H Schmitz, Patricia O Emplaincourt, et al. 2000. Compendium of physical activities: an update of activity codes and MET intensities. *Medicine and science in sports and exercise* 32, 9; SUPP/1 (2000), S498–S504.

[2] Author. 2021. Multi-Window-Finder Website. https://sites.google.com/view/multi-window-finder/.

[3] Arvind Balasubramanian, Jun Wang, and Balakrishnan Prabhakaran. 2016. Discovering multidimensional motifs in physiological signals for personalized healthcare. *IEEE journal of selected topics in signal processing* 10, 5 (2016), 832–841.

[4] Björn Barz, Yanira Guanche Garcia, Erik Rodner, and Joachim Denzler. 2017. Maximally divergent intervals for extreme weather event detection. In *OCEANS 2017-Aberdeen*. IEEE, 1–9.

[5] Yifeng Gao, Jessica Lin, and Huzefa Rangwala. 2017. Iterative grammar-based framework for discovering variable-length time series motifs. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 111–116.

[6] Eamonn Keogh and Jessica Lin. 2005. Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and information systems* 8, 2 (2005), 154–177.

[7] Frank Madrid, Shima Imani, Ryan Mercer, Zachary Zimmerman, Nader Shakibay, and Eamonn Keogh. 2019. Matrix profile xx: Finding and visualizing time series motifs of all lengths using the matrix profile. In *2019 IEEE International Conference on Big Knowledge (ICBK)*. IEEE, 175–182.

[8] Takaaki Nakamura, Makoto Imamura, Ryan Mercer, and Eamonn Keogh. 2020. MERLIN: Parameter-Free Discovery of Arbitrary Length Anomalies in Massive Time Series Archives. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1190–1195.

[9] Xiaozhe Wang, Kate A Smith, Rob Hyndman, and Damminda Alahakoon. 2004. A scalable method for time series clustering. *Technical report* (2004).