Matrix Profile XIII: Time Series Snippets: A New Primitive for Time Series Data Mining

Shima Imani[1], Frank Madrid[1], Wei Ding[2], Scott Crouter[3], Eamonn Keogh[1]
[1]*Department of Computer Science and Engineering*
University of California, Riverside
[2]*Department of Computer Science University of Massachusetts Boston*
[3]*College of Education, Health & Human Sciences University of Tennessee, Knoxville*
[1]siman003@ucr.edu, [1]fmadr002@ucr.edu, [1]eamonn@cs.ucr.edu
[2]Wei.Ding@umb.edu, [3]scrouter@utk.edu

***Abstract*—Perhaps the most basic query made by a data analyst confronting a new data source is "*Show me some representative/typical data.*" Answering this question is trivial in many domains, but surprisingly, it is very difficult in large time series datasets. The major difficulty is not time or space complexity, but defining what it means to be *representative* data in this domain. In this work, we show that the obvious candidate definitions: motifs, shapelets, cluster centers, random samples etc., are all poor choices. Thus motivated, we introduce *time series snippets*, a novel representation of typical time series subsequences. Beyond their utility for visualizing and summarizing massive time series collections, we show that time series snippets have utility for high-level comparison of large time series collections.**

***Keywords*—time series, motifs, sampling, diversification**

I. Introduction

In many domains, a common analytical query is "*Show me some representative/typical data*." This query might be issued by a human, attempting to explore a massive archive, or it might be issued by an algorithm as a subroutine in some higher-level analytics. There are definitions and algorithms to answer this question for a plethora of datatypes, including images [25], sets [16], words [21], graphs [12], videos [4], tweets [20], etc.

Surprisingly, to the best of our knowledge, the problems of finding representative time series subsequences has not been solved despite the ubiquity of time series in almost all human endeavors. Moreover, as we will show, the obvious candidates for this task: motifs, shapelets, cluster centers, and random samples, will not generally produce meaningful results. We propose an algorithm to discover such representative patterns, which we will call *time series snippets*, or just *snippets*, where there is no ambiguity.

We would like snippets to have the following properties:

- **Scalable Computability**: We wish to find snippets in datasets that defy rapid human inspection. Such datasets will be large. While we can often offload snippet discovery to offline batch preprocessing, we clearly cannot afford an algorithm that requires a large time or space overhead.
- **Diversity**: The top-1 snippet should clearly be the *single* most representative pattern. Clearly, the 2nd and subsequent snippets should not be redundant with previous snippets.
- **Diminishing Returns**: As an implication of diversity, the earlier *k*-snippets have the greatest coverage, just as the dimensions in Multidimensional scaling are sorted by their rapidly diminishing ability to "explain" the variability of the original data.
- **Quantifiability**: Suppose the top-1 snippet for a sleep study shows some typical healthy heartbeats. It may be that *all* the data looks like that, or it may be that there were also regions of arrhythmias. Thus, we need some metadata to tell us how *much* of the data each snippet can explain/represent.
- **Domain Agnosticism**: For specialized domains it may be possible to leverage domain knowledge and/or training data to achieve all of the above; however, we wish to have a general purpose algorithm to support data exploration.

In this work we introduce an algorithm, *Snippet-Finder*, which can discover snippets with all such properties. Fig. 1 shows one potential use of snippets: integrating summarizations of files directly into an operating system.

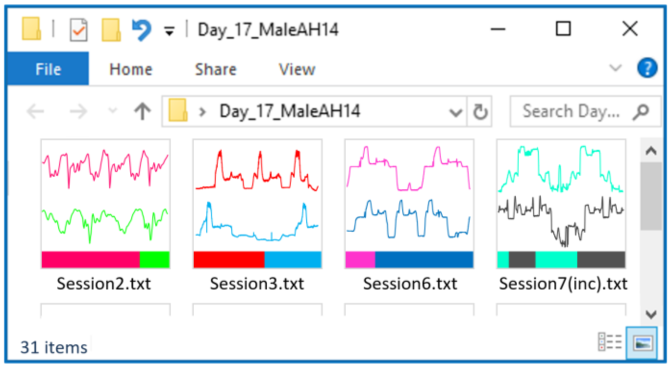
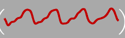
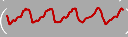


Fig. 1. One use of time series snippets is to replace standard file icons with icons that show snippets reflecting the file's content. This can allow an analyst to spot patterns and anomalies at a glance [11]. This is real data, see Section IV.C for more context for this example.

Another potential use of snippets is in the production of automatically generated reports. For example, one could compactly summarize a sleep study [1] with a report like this:

Patient Smith slept for 7.2 hours. This ten-second snippet () accounts for 78% of his respiration, and this () ten-second snippet accounts for 17% of his respiration. His maximum temperature was 98.7°...

Beyond the algorithm's utility for visualization and summarization [27], snippets can be used to support a host of higher-level tasks, including the comparison of massive data collections.

The rest of this paper is organized as follows. In Section II we briefly review related work and background material, then formally define time series snippets. We introduce an ultra-fast algorithm to compute time series snippets in Section III. Section IV shows the utility of time series snippet discovery. Finally, in Section V we offer conclusions and directions for future work.

II. RELATED WORK AND BACKGROUND

A. *Dismissing Apparent Solutions*

As noted above, the task at hand invites many *apparent* solutions. Here we take the time to dismiss them.

Motif Discovery: At first glance, *motif discovery* seems like an obvious solution to our problem [26][28]. Consider Fig. 2, and imagine we are tasked with finding the time series snippet of length 300, or snippet$_{300}$.

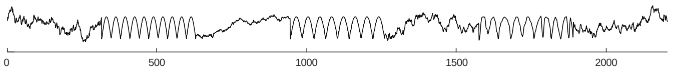


Fig. 2. A synthetic dataset. There are three repeated patterns embedded in a random walk; the best snippet$_{300}$ appears obvious to the naked eye.

Surprisingly, even in the face of such an apparently simple dataset, motif discovery will not produce a satisfactory answer. Consider the clustering shown in Fig. 3.

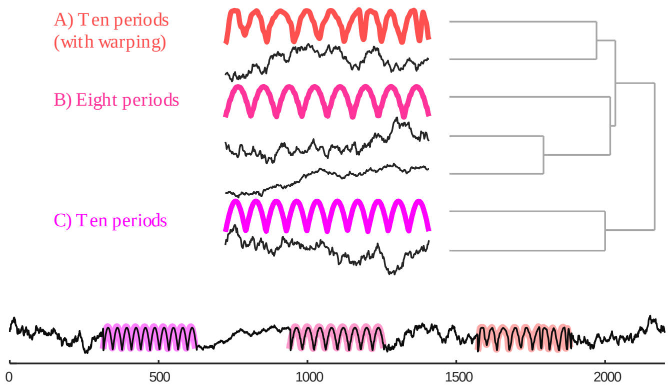


Fig. 3. *bottom*) The dataset shown in Fig. 2 annotated by the subjectively correct answer. *top*) The dataset divided into seven equal length regions, and clustered using complete linkage hierarchical clustering.

The three periodic elements are no more similar to each other under Euclidean distance than they are to random walks. One might imagine that the solution to this issue is to use a different distance measure, say Dynamic Time Warping (DTW). However, while DTW can be invariant to warping and find **A** and **C** similar, it cannot warp eight periods to ten periods. Two periods must be "unexplained" and accumulate a large error. Thus, **B** is still no closer to **A** or **C** than it is to random walks under DTW.

There are other reasons why motif discovery is not suitable to the task at hand. Consider the dataset shown in Fig. 4.*top*. At this scale, it is difficult to even guess its content. It is exactly the type of data that could benefit from snippet-based summarization. This dataset represents fifty-five minutes of Arterial Blood Pressure (ABP) collected during an experiment in which a volunteer strapped to a gimbal was subjected to sudden changes of orientation [7]. However, as Fig. 4.*bottom* shows, this dataset has an occasional sensor fault. When the sensor is not receiving true medical data, it instead sends a square wave calibration signal.

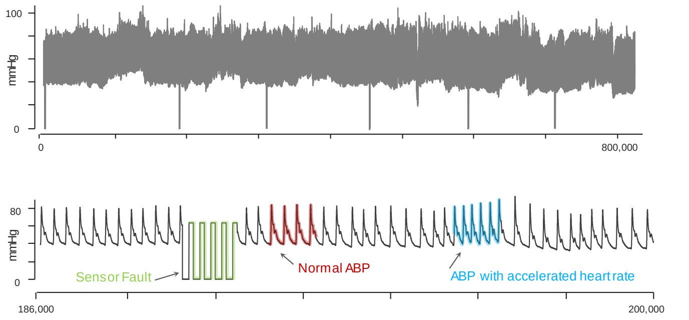


Fig. 4. *top*) Fifty-five minutes of APB data, taken from an individual experiencing occasional involuntary rapid changes in orientation [17]. *bottom*) A minute-long zoom-in starting at about the twelfth minute.

If we defined snippets based on time series motifs, then the five identical regions of such data would be the top snippet, even though such data only represents 0.14% of the dataset. However, there *are* snippets that are clearly much more representative of the data. For example, as the highlighted regions in Fig. 4.*bottom* suggest, normal ABP makes up the majority of this dataset, while a significant minority comprises of an ABP with an increased heart-rate induced by a change in gravity [14].

The issue can be summarized as the following: While motifs reward *fidelity* of conservation, we need a measure that also rewards *coverage*. Informally, coverage is some measure of how much of the data is explained or represented by a given snippet.

Representative Trends: By title, the highly cited research effort on "*Identifying Representative Trends in Massive Time Series Data Sets...*" [9], seems like it may offer a solution, or at least insight to the task at hand. However, the authors of this work are assuming that the time series has well-defined periods; for example, exactly twenty-four hours, and a well-defined starting point, say midnight. However, choosing another starting point, such as 1:00 am, could produce arbitrarily different results. Moreover, while the well-defined periods assumption might hold for traffic (both in the web and in automobile sense) and other quotidian human activities, it does not hold for most of the medical and scientific domains we are interested in. For example, the periodicity of heartbeats can vary by at least a factor of five, in just a few minutes. At a higher level, cardiac events clearly do not align themselves to midnight, 8:00 am, or any other arbitrary time frame[1].

Clustering: In many cases, summarizing a dataset into the *K* best exemplars is as simple as running *K*-means clustering and reporting the *K* centers (or the *K* exemplars closest to the centers). However, it is well understood that one cannot meaningfully cluster time series subsequences, with any distance measure, or with any algorithm [10]. This (at the time) surprising result was controversial a decade ago, but has since become universally accepted (see [10] and the references therein); we will avoid repeating the arguments here. In Section IV, we make a best faith effort to fix this issue, and compare to a *K*-means based algorithm.

[1] In a sense *midnight* is not arbitrary, as it marks the midpoint between sunset and sunrise. However, due to time zones and daylight-savings time, it rarely coincides with 12 midnight on the clock. Midnight is really an arbitrary cultural artifact.

Time Series Shapelets: Time series shapelets are defined as subsequences that are maximally representative of a class [26]. This sounds superficially like time series snippets, however shapelets are *supervised*, and we require an unsupervised technique. Moreover, shapelets are generally biased to be as short as possible. In contrast, we want snippets to be longer, to intuitively capture the "flavor" of the time series. By analogy with text, to *distinguish* between English and Lithuanian chapters of J. K. Rowling's famous heptalogy, it is sufficient to see the letter "Č", this is the equivalent of a shapelet. However, to *summarize* the latter with representative text, it would be much more informative to see something like "*Haris Poteris*", the equivalent of a snippet.

Random Sampling: Simple random sampling (SRS) has many desirable properties that makes it useful and competitive in many domains. There are clearly cases in which it would be sufficient. For example, if a time series dataset consists solely of normal regular heartbeats, then any random two-beat region we extract would summarize the data (two beats because cardiologists are used to visualizing beats from a fixed starting point, two beat will include that point). However, as we shall see, even cardiological datasets can have surprising variability, and random sampling would have to be very lucky to hit one each of, say, three diverse regions. Nevertheless, we will include random sampling as a baseline in our experimental section.

B. *Related Work*

Our review of related work is brief. To the best of our knowledge, there are simply no ideas closely related to domain agnostic in the *time series* domain.

In recent work, it was demonstrated how to exploit representative electrocardiogram heartbeat morphologies based on the CUR matrix decomposition technique [8]. Given a matrix *A*, CUR technique selects a subset of rows and columns of *A* to construct matrices *C* and *R*. Matrix *U* is computed in a way that makes the multiplication of *C*, *U*, and *R* the best approximation of *A*. However, this study is specific to a particular type of data, and requires substantial human effort to align data.

There is a significant work on the automated extraction of music snippets (also called music *thumbnails*) [15]. However, that literature addresses a specialized and limited form of time series data. Most songs are richly structured into some variant of *intro*, *verse*, *chorus*, *bridge*, and *outro*. Moreover, most songs are only a few minutes in length, or a few thousand time series data points in an MFCC representation. In contrast, we wish to consider unstructured datasets with tens of millions of data points.

Other related work was reviewed in Section II.*A*; we will not duplicate that here.

C. *Time Series Notation*

Before we formally define time series snippets, we need to review some related definitions (Definitions 1 to 3), and create some new ones (Definitions 4 to 6).

The data type of interest is *time series*:

Definition 1: A *time series* T is a sequence of real-valued numbers t_i: $T = t_1, t_2, ..., t_n$ where n is the length of T.

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

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

If we "*slide*" a window of length m across T we produce $n\text{-}m+1$ subsequences. However, we can also produce a set of *non-overlapping* subsequences:

Definition 3: A non-overlapping subsequence S_i of a time series T is a continuous subset of the values from T of length m, starting from the position $(i-1) \times m + 1$ and ending at the position $i \times m$, in which the value of i is an integer number chosen from $i = 1{:}\, n/m$. When n/m is not an integer number, zeros are padded to the end of the time series until n/m becomes an integer number.

Note that the number of non-overlapping subsequences is much smaller than the sliding windows, just $\lfloor n/m \rfloor$. For virtually any task, if working with Euclidean distance, you *must* use sliding windows. Otherwise, the higher-level algorithm would be brutally sensitive to the starting point [10]. In contrast, as we will show in Section III, using the much smaller set of non-overlapping subsequences is inconsequential if we use the MPdist, which we review in the next section.

We can now define *time series snippets*:

Definition 4: A time series snippet is a subsequence of T. Snippets are arranged in an ordered list of C, with the ith snippet denoted as C_i.

We can access some useful metadata from the snippet, such as its location within T, and the fraction of T that it is said to represent, by $C_{i.\text{index}}$ and $C_{i.\text{frac}}$ respectively.

Note that our definition means that the snippets are actual subsequences of T. This need not have been the case. For example, consider the related problem of finding representative *strings*. The most common solution, variants of consensus strings [23], can produce a string that is optimal under some definition, but never actually appears in the data.

D. *A Brief Review of MPdist*

MPdist is a recently introduced distance measure which considers two time series to be similar if they share many similar subsequences, regardless of the *order* of matching subsequences [6]. It was demonstrated in [6] that MPdist is robust to spikes, warping, linear trends, dropouts, wandering baseline and missing values, issues that are common outside of benchmark datasets. The MPdist requires a single parameter called a subsequence length, S. We denote the value of this parameter with a subscript, as in MPdist_{50}. In the limit, when S is equal to the full length of the query, the MPdist degenerates to the special case of the classic Euclidean distance. The time complexity of MPdist in the worst case is $O(n^2)$, however its *amortized* complexity for the subsequences similarity search is just $O(n \times (\text{queryLength} + m - 1))$.

Consider the small toy example of a time series shown in Fig. 5, which we will use as a running example.

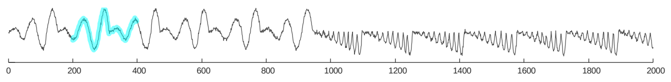


Fig. 5. A toy time series. The highlighted section, from 201 to 400, will be used in subsequent examples.

We can use the MPdist to create an *MPdist-profile*:

Definition 5: An MPdist-profile of time series T is a vector of the MPdist distances between a given query subsequence $T_{i,m}$, and each subsequence in time series T. Formally, $MPD_i = [d_{i,1}, d_{i,2},..., d_{i,n-m+1}]$, where $d_{i,j}$ $(1 \leq i, j \leq n - m + 1)$ is the distance between $T_{i,m}$ and $T_{j,m}$.

While there is no ambiguity, we will refer to MPdist-profiles simply as *profiles*. Consider the profile for the time series shown in Fig. 5, using the highlighted region as the query. The subsequence length of this query is $L = 200$. The result is shown in Fig. 6. Notice that the length of the profile is shorter than the length of the time series by the length of query. Furthermore, note that the distance is *exactly* zero in the region from which the query was extracted, since the MPdist between an object and itself must be zero.

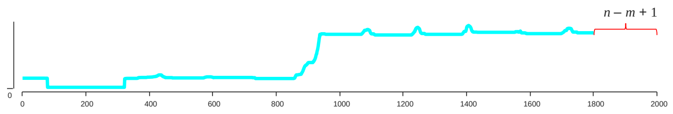


Fig. 6. The profile of the query highlighted in Fig. 5 using MPdist$_{70}$.

As we shall show in the next section, our snippet discovery algorithm essentially reduces to "reasoning" about these profiles.

Our desired properties of *diversity* and *diminishing returns* suggests that we frame snippet discovery in a familiar "top-*k*" framework, much like *k*-itemsets or *k*-frequent items, etc. However, there is a caveat. It may be that there are as few as one snippet in a dataset, such as when the time series is comprised solely of a pure sine wave. Therefore, we will also need to provide some metadata for each snippet to quantify how well it represents the data.

III. DISCOVERING TIME SERIES SNIPPETS

We begin with a demonstration that both previews our method for discovering time series snippets, and shows why the MPdist is critically needed for this task.

Consider again our running example time series shown in Fig. 7: this time series is annotated with two subsequences, which the reader will appreciate might serve as good snippets for this dataset.

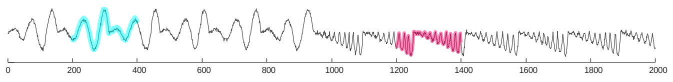


Fig. 7. A toy time series that will be used as a running example. The two highlighted sections, from 201 to 400 and from 1201 to 1400, will be used in subsequent examples.

Let us extract the two highlighted subsequences and then compute their profiles. The results are shown in Fig. 8.

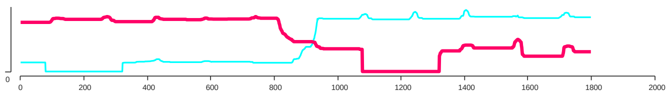


Fig. 8. The profiles of the two queries highlighted in Fig. 7. Note their mutually exclusive nature, when one in high, the other is low.

Note that these profiles offer strong clues to the locations of potential time series snippets. They are both approximately "step" functions with their respective low region corresponding to a region that contains subsequences that are similar to our two query patterns.

Furthermore, note that these profiles are "mutually exclusive;" that is to say, when the red one is low, the cyan one is high, and vice versa. This suggests that these two hand-chosen snippets would meet the diversity requirement listed in the introduction, as they both "explain" different and non-overlapping regions of the data.

Note that the shapes of these two profiles are very robust to both *location*, from which we extract the pattern, and to the query pattern's *length*. To see this, in Fig. 9 we recomputed the profiles for both a shifted and a longer version of our query.

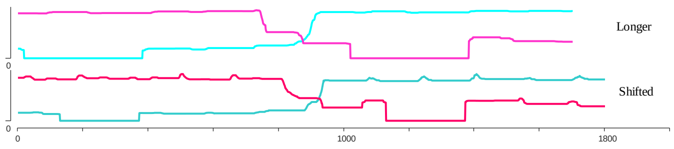


Fig. 9. The profiles of after: Shifting the queries 50 points to the right, and making the queries longer (extracted from 201 to 500, and from 1201 to 1500 respectively).

This result offers hope for an algorithm that is not too sensitive to the location or length of the candidate snippets. To see why this is a significant finding, Fig. 10 shows what happens if we computed these profiles with the Euclidean distance instead.

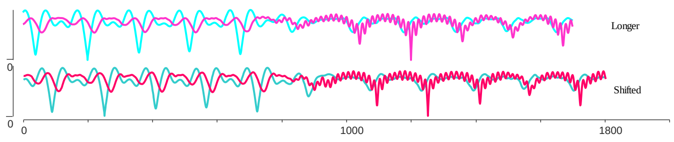


Fig. 10. The Euclidean Distance Profiles of after: Shifting the queries 50 points to the right, and making the queries longer (extracted from 201 to 500, and from 1201 to 1500 respectively). Contrast with Fig. 9.

In contrast to the MPdist-Profile, the Euclidean distance Profiles are more sensitive to the length and offset of the subsequence. More importantly, they only have a low value when the matching pattern is exactly in phase. Thus, in contrast to the MPdist-Profile, it is difficult to "reason" about them, to understand how large a region they could represent as a prototype.

A. *Snippet-Finder: Time Series Snippet Discovery*

We are finally in a position to explain our *Snippet-Finder* algorithm. Given a time series, a subsequence length, and the maximum number of snippets that user wishes to find, the *Snippet-Finder* algorithm is to identify *k* snippets, and the fraction of data that each snippet covers. Note that the fraction of data covered by a single snippet is not necessarily contiguous. For example, an accelerometer dataset may consist of bouts of "walk-run-walk-cycle-walk;" a snippet of *walking* would represent all three regions of the slower gait.

We begin with the following intuition. Recall that Fig. 7 showed a time series with two subsequences highlighted. Those subsequences would clearly make intuitive snippets for that dataset. Recall also that Fig. 8 showed two profiles that offer strong clues that these subsequences would be excellent top-2 snippets. Together, their respective low-value regions cover almost the entire length of the time series, swapping over at about location 950.

In contrast, suppose instead we had made a *poor* choice of two snippets for this dataset. As shown in Fig. 11, if we selected both snippets from the first half of the data, they would be highly redundant with each other, and this would be reflected in the high correlation of their profiles.

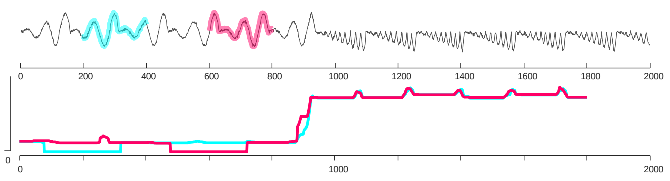


Fig. 11. (contrast with fig 7 and fig 8). *top*) Revisiting our running example, we extracted two similar snippets (highlighted). *bottom*) The redundancy of the snippets is revealed in the high correlation of the profiles.

This observation immediately suggests an objective function that we can use to score choices of top-*k* snippets. Consider a non-empty set of profiles. Create a new curve, *M*, by taking the minimum value from all *k* profiles at each location. This new curve allows an *objective function*:

Definition 6: The area under the curve *M* is denoted as *O*, and is an objective function where $0 \leq O$. We will refer to the area under the curve of profile as the *ProfileArea.*

This function rewards a snippet's *fidelity* and *coverage* exactly as required. Fidelity is rewarded by the snippet having a small distance to at least some of the *T*, lowering the profile and reducing the area under the curve in the corresponding region. Coverage is rewarded by the snippet representing large regions of *T*. Note that *O* has the intuitive property wherein if we used every non-overlapping subsequence as a snippet, its value would be exactly zero. Of course, we hope that in most cases, using just a few snippets will get us *close* to zero, achieving significant "compression" or more correctly, numerosity reduction.

In the simple example shown in Fig. 7, we found a low scoring value for *O* simply by using common sense to pick one snippet from each of the repeated patterns. More generally, if we had computed all of the non-overlapping profiles, we would have the "tangle" of profiles shown in Fig. 12.

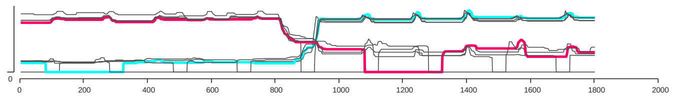


Fig. 12. All the non-overlapping profiles for the time series shown in Fig. 7.

For a more realistic dataset, finding the *k* profiles that minimize *O* from the *k*-choose-*P* possibilities $\binom{k}{P}$ would be untenable. Thus, we frame the snippet discovery problem as a classic search problem. An exhaustive combinatorial search is infeasible, so below we outline a greedy search strategy.

The main algorithm for Snippet-Finder is outlined in TABLE I, and its subroutine that computes profiles of each non-overlapping window is outlined in TABLE II.

The main algorithm begins in line 1 by initializing the list of snippets *C*. In line 2, we calculate the profile of each non-overlapping window with the time series (see TABLE II). At each iteration we calculate the minimum of each profile with the profiles within the snippet list *C*, and for each one, we find the area under the curve *ProfileArea*. For $k = 1$, the *ProfileArea* is simply equal to the area under the curve of each profile. The profile that has the minimum *ProfileArea* will be added to the snippet list *C*. We also add the location of each snippet within *T*. This is done in lines 4 to 15. We evaluate the terminal condition when we reached the number of snippets the user requested in line 16 to 18. After finding the top-k snippets, in line 20 we compute the minimum value from all *k* profiles *totalmin*. In line 21 to 24, we compare the top-k snippet profiles with *totalmin*. The number of points that these two curves have the same value, *f*, gives us the fraction of data each snippet represents $C_{i.frac}$. Finally, when the algorithm terminates, the list of snippets *C*, the fraction of data that *i*th snippet represents, the data $C_{i.frac}$ and the location of snippet within the time series $C_{i,index}$ is returned.

TABLE I. Snippet-Finder: SNIPPETS SELECTION ALGORITHM

```
Procedure: SnippetsSelectionAlgorithm (T, m, k)
Input: time series T, subsequence length m, number of snippets k
Output: list of snippets C, fraction of data that snippet represents C_{i.frac}
location of each snippet C_{i.index}
1   C ← Ø, snippetProfiles ← Ø,  Q ← Inf, n ← Length(T)
2       D ← GetAllProfile (T, m)                          //TABLE II
3   While True
4     minimumArea ← Inf
5     for i = 1:n/m
6       ProfileArea ← sum ( min (D [i], Q) )          // Definition 6
7       if minimumArea > ProfileArea
8         minimumArea ← ProfileArea
9         idx ← i
10      end if
11    end for
12    Q ← min (D [idx], Q)
13    C ← T [(idx - 1) × m + 1: idx × m]
14    C_{i.index} ← idx
15    snippetProfiles ← D [idx]
16    if Length (C) = k       // if reaching the number of user defined snippets
17      break
18    end if
19  end While
20  totalmin = min (snippetProfiles)   // minimum area under the profiles
21  for i = 1:k
22    f ← all the points in snippetProfiles[i] which is equal to totalmin
23    C_{i.frac} ← f / (n-m+1)
24  end for
25  return (C, C_{i.frac}, C_{i.index})
```

To calculate the profile of each non-overlapping window, we use the subroutine outlined in TABLE II.

The algorithm begins in line 1 by determining the length of the time series *T*, and initializes *D* to the empty list. From lines

2 to 4, we calculate all the profiles *D*, using each non-overlapping subsequence $T\,[(i\text{-}1) \times m + 1{:}i \times m]$ and the time series *T*. Finally, we return the result *D* in line 5.

TABLE II. CALCULATING DISTANCE PROFILES

```
Procedure GetAllProfiles (T, m)
Input: time series T, Subsequence length m
Output: list of profiles D
1  D ← Ø, n ← Length (T)
2  for i = 1:n/m
3     D ← Profile (T, T [(i-1) × m+1:i × m])
4  end for
5  return D
```

B. Snippet MetaData

In addition to computing the snippets themselves, it is useful to compute some metadata that reflects how well the current snippet set is representing the dataset in question. In this sense we have a close analog to how *K*-means clustering is used. In some cases, say quantizing a color space, the number of clusters requested of *K*-means may be fixed in advance, regardless of the data itself. This is similar to our use of snippets for creating icon "thumbnails," as shown in Fig. 1. Because the operating system limits us to 256 by 256 pixels, we hardcoded the number of snippets displayed to *k* = 2.

However, in other cases, we want the *K*-means clustering to reflect the "natural" clusters in the data. While the task of deciding what value of *k* best does that is outsourced to an external algorithm; *K*-means provides helpful information by reporting the objective function (sum of squares), a relative measure of how well the clustering reflects the data. We would like an analogous objective function for snippets.

We propose using the area under the curve *ProfileArea* as such a measure. As Fig. 13.*left* shows, for increasing values of *k*, this measure shows a classic diminishing returns behavior.

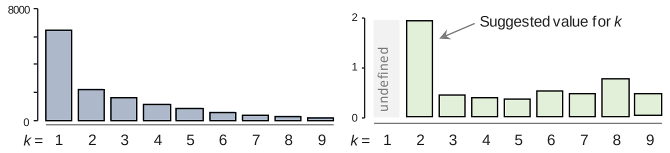


Fig. 13. *left*) The area under the curve *ProfileArea* for *k* = 1 to 9, for the running example shown in Fig. 7. *right*) For the same example we can also compute $change_k$ for *k* = 1 to 9.

This "scree-plot"-like curve suggests a heuristic that can be used to recommend a value for *k*. Similar to many suggested knee-finding algorithms in the literature, for each value of *k* we can compute $change_k = (ProfileArea_{k\text{-}1} / ProfileArea_k) - 1$. For a given value of $k > 1$, a small value for $change_k$ suggests that adding the k^{th} did not add much explanatory power; thus, we should use a value of *k* corresponding to a peak. As shown in Fig. 13.*right*, this heuristic suggests a value of *k* = 2 for our toy example, which is objectively correct. Unless otherwise stated, we use this simple heuristic in the rest of this work.

C. Complexity Analysis

The time complexity of our proposed Snippet-Finder is $O(n^2 \times (n-m)/m)$. We need $O(n^2)$ for computing each profile and the number of non-overlapping windows is $O((n-m)/m)$. The pseudocode is optimized for simplicity, with an $O((n-m)^2/m)$ space complexity. However, the space complexity in a slightly more sophisticated implementation is merely $O\big((n-m) \times k\big)$. In addition, the *ProfileArea* curves only need to be represented with one-byte precision.

To concretely ground this analysis, consider the following. For typical datasets recorded at ~100 Hz (gait, ECG, insect telemetry, etc.), we can discover snippets many orders of magnitude faster than the data is collected. Moreover, even very large data collections can be processed using a small fraction of the main memory of a modern machine. Thus, computational resource limitations are not barriers to adoption.

IV. Empirical evaluation

We begin by stating our experimental philosophy. We have designed all experiments such that they are easily reproducible. To this end, we have built a web page [24] that contains all of the datasets and code used in this work as well as the spreadsheets containing the raw numbers. The one hundred test datasets we created will be archived in perpetuity independent of this work. We hope that the archive will grow as the community donates additional datasets.

As we noted above, to the best of our knowledge there is no explicit strawman other than random sampling. However, we can create a simple strawman based on time series *clustering*, which is the most obvious, sensible "first thing to try." For this algorithm, we random sample *f*% of the data subsequences. We then cluster the subsequences with *K*-means. For each cluster we find the subsequence that has the minimum distance to all other subsequences in the cluster (i.e. the medoid) and report these as the *k*-snippets. This leaves open the issue of finding a good value for *f*; for simplicity we allow this algorithm to "cheat" by testing all values in the range 1 to 20%, and reporting only the *best* result.

We also report the results of random sampling, which plays a role similar to the "default rate" for classification, setting a lower bound for performance.

A. Objective Experiments

Our task at hand, to produce "typical" time series subsequences, seems hopelessly subjective and difficult to evaluate in an objective way. However, we can convert the problem to one for which objective truth is available. Suppose we have a dataset which comprises of a minute of `walking`, followed by a minute of `running`. If we queried such a dataset for the top-2 snippets, we would surely hope to find one snippet from each gait type. This we count as a *success*, while any other result we count as a *failure*.

Of course, here there is a ½ chance of success with random sampling. More generally, if the two behaviors are different fractions of the length of the full-time series *L* and *R*, with *L* + *R* = 1, the probability of random sampling returning a satisfactory pair of snippets is 2 × *L* × *R*.

Thus the task is made more difficult by having an asymmetric length of behaviors. Nevertheless, a single successful experiment would not be very convincing. Thus, to stress test our algorithm, we created a diverse collection of one hundred time series that we call MixedBag. Fig. 14 shows some representative examples.

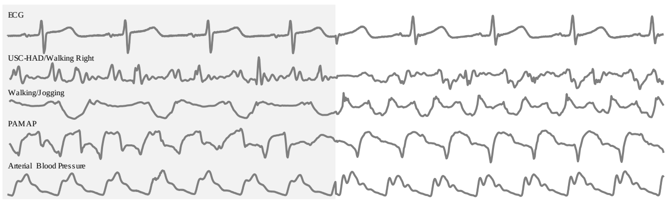


Fig. 14. Five examples from the MixedBag archive, aligned such that the behavior change takes place at the center. Note that the examples are much longer, as data was truncated from both ends for visual clarity.

Some of our examples are completely natural, and we are confident from some external knowledge that there are only two behaviors, and that we have correctly identified the transition point. For some datasets, we slightly contrived the data to give us an unambiguous ground truth; for example, we took two one-minute ECG traces of two different individuals and concatenated them. Our examples draw from robotics, human locomotion, ornithology (vocalizations), ornithology (flight), entomology, speech processing, pig physiology, etc.

The scoring function is simply the sum of all *successes* in our one-hundred experiments. Table III shows the result for the scoring function for three different algorithms.

Table III. The performance of three algorithms in snippet discovery

Dataset	*K*-means	Random Sampling	Snippet-Finder
MixedBag	57	47.3*	84

* Calculated exactly, not computed experimentally.

While the results are impressive, we should point out that even the 16% of cases where we failed may not represent true failures. For example, consider a dataset that is labeled {`walk`|`jog`}, as in Fig. 14. If we reported two snippets that came from the walking section, our scoring function reports *failure*. However, it is possible that the walking section actually includes two distinct behaviors, say, `walking upstairs` and then `walking downstairs`, and those labels are not available to us. In such a case, a snippet from each of the two latent walking behaviors may actually represent a *success*.

B. Robustness Tests

The experiments in the previous section could justifiably be criticized for being too contrived. All of the data belongs to exactly one of two behaviors, with no "distracting" subsequences. However, most real data has such sections. For example, the PAMAP dataset is comprised mostly of `walk`, `run`, `cycle`, etc. However, there are short regions of ill-defined and less structured data, such as when the user pauses at a stop light, or transitions from walking to cycling and spends a few ill-defined moments unlocking the bicycle and carrying it to the side of the road. It is natural to ask how our method fares when presented with such distracting regions.

To test this, we repeated the experiment above, this time after concatenating regions of distracting data to the original time series. To ensure that the distracting data itself does not have patterns worthy of being summarized by a snippet, we used both random data, and random walk data. We considered the cases where distracting data of 10% and 20% of the original time series length was appended to the original time series. Table IV shows the result for the scoring function for three different algorithms in these more challenging scenarios.

Table IV. The performance of three algorithms in snippet discovery

Dataset	*K*-means	Random Sampling	Snippet-Finder
$\text{MixedBag}_{\text{RandWalk10}}$	51	39.1*	84
$\text{MixedBag}_{\text{RandWalk20}}$	50	32.8*	76
$\text{MixedBag}_{\text{Rand10}}$	52	39.1*	82
$\text{MixedBag}_{\text{Rand20}}$	50	32.8*	82

* Calculated exactly, not computed experimentally.

We observed that our proposed algorithm is largely invariant to the reasonable amount of spurious distractor data. The performance *does* begin to suffer as we see large amounts (>10%) of random walk data, as this data has the property that any two randomly selected subsequences tend to be closer than any random subsequences of structured data (i.e. gait, heartbeats etc.). In [2] there is an explanation of this phenomenon, and a possible solution (a correction factor for the Euclidean distance). However, given the high performance of our simple algorithm, we leave such considerations for future work.

We have established that our snippet discovery algorithm can robustly find typical time series in diverse settings. In the following sections we show some case studies that demonstrate the *utility* of snippet discovery in real world applications.

C. Case Study: Human Behavior

We consider a dataset collected as part of a large-scale five-year NIH-funded project at the University of Tennessee. More than one hundred youths are being monitored during a semi-structured free-living period (development group) while another one hundred youths are being monitored during true free-living activities during an after-school program and at home (validation group). Fig. 15 shows an example of the former data type. Note that for *some* data we have video, thus ground truth.

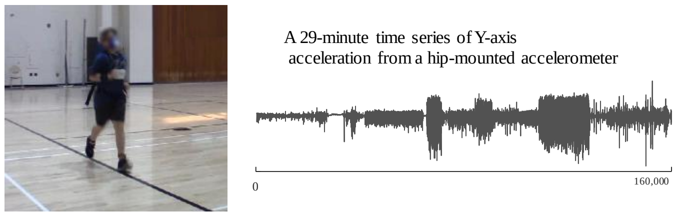

A 29-minute time series of Y-axis acceleration from a hip-mounted accelerometer
0
160,000

Fig. 15. *left*) A participant in the Tennessee study. He is wearing a portable ergospirometer, and sensors on each limb. *right*) A small sample of the data collected as the participant runs around a basketball court[2].

Collecting and analyzing this massive dataset is a difficult task. Even beyond the scientific goals of the study, there are basic issues of compliance (*did the participant perform his prescribed 30 minutes of exercise at some point today*?).

[2] The procedures were reviewed and approved by The University of Tennessee Institutional Review Board before the start of the study. A parent/legal guardian of each participant signed a written informed consent and filled out a health history questionnaire, and each child signed a written assent prior to participation in the study.

For this experiment, we consider sensor data from a hip-mounted accelerometer, which is collected at 90 Hz. In Fig. 16 we show the snippets discovered.

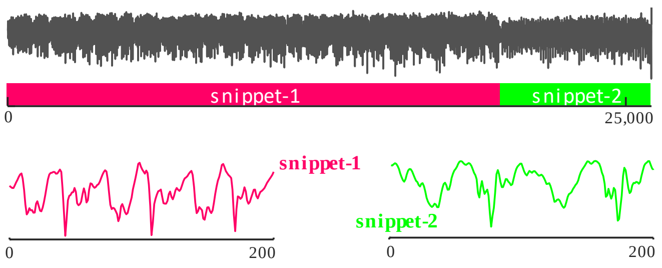


Fig. 16. *top*) A five-minute region of behavior for the study participant. While it is only a tiny fraction of the full day, it still defies quick visual interpretation. *bottom*) the top-2 snippets discovered in this dataset.

At a quick glance at the output of Fig. 16, the study organizer can immediately tell that this data represents about four minutes of running followed about a minute of very slow “cool-down” walking. Note that this example was used in Fig. 1 as an example of the thumbnail icon summarization tool.

In Fig. 17 we show the profiles that were used to compute the top two snippets. Note that this is not a view that would be shown to an end-user; it is just an internal representation used by the snippet discovery algorithm. Here we show it to give context to the figures shown in Section III as we describe our algorithm, and to explain how we compute the “regime bar.”

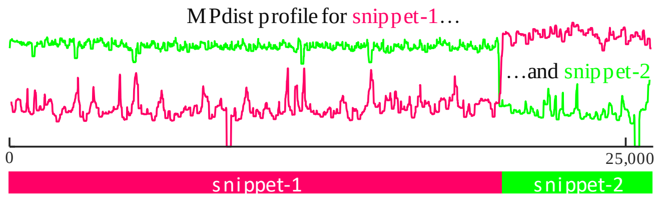


Fig. 17. *top*) The profiles that were used to compute the top two snippets. *bottom*) It is from these curves that we obtain the “regime bar,” which tells us which snippet explains which region of data.

In this example, it happens that only one example of each regime is in the regime bar. However, that does not have to be the case. To see this, as shown in Fig. 18, we repeated the experiment with similar data from the PAMAP dataset, a widely used benchmark [18].

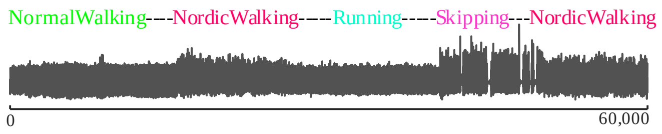


Fig. 18. A ten-minute region of behavior from the PAMAP dataset [18]. The data is the Y-axis acceleration from a chest-worn sensor. In this dataset the ground truth is available from careful annotations made at the time.

As shown in Fig. 19, we extracted the top-4 snippets from this dataset. Note that one of the snippets, snippet-3, reflect two non-contiguous regions.

Note that the skipping (rope-jumping) section has several discontinuities, presumably because the participant caught her foot in the rope and had to restart. In spite of these unstructured regions, the top-4 snippets perfectly summarize this data set.

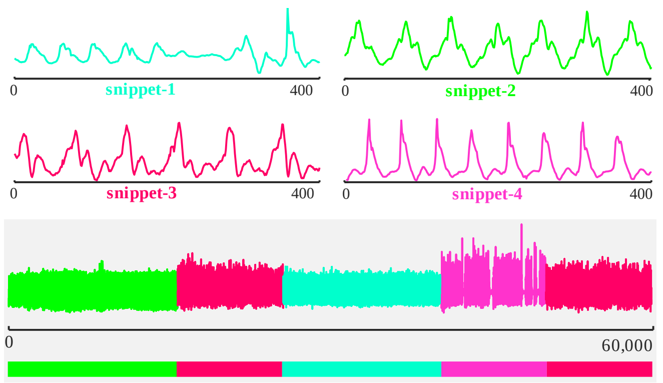


Fig. 19. *top*) The top-4 snippets from the PAMAP dataset. To someone with familiarly with this dataset/domain, it is easy to label the four behaviors. *bottom*) As an alternative to creating a regime bar, we can simply brush the colors of each snippet onto the original time series.

D. Case Study: Medicine

Working with clinicians of the David Geffen School of Medicine at UCLA, we are creating a tool to summarize ICU (Intensive Care Unit) telemetry. In an ICU setting, for a Level-3 patient, a typical protocol requires a nurse or doctor to go physically bedside once an hour and examine both the patient and their vital signs displayed on the bedside monitor [3][5]. The latter typically contains the most recent data in the sliding window for the last twelve seconds. However, it may be that this most recent data is *atypical* of the last hour. For example:

- If the patient recently sneezed, this will change her intrathoracic pressure, decreasing the flow of blood to the heart, which then is forced to compensate. This can change the ECG, arterial blood pressure (ABP), and respiration.
- The mere presence of medical staff may induce stress in a conscious patient and change their physiological readings.

Thus, we argue instead of, or in addition to, presenting the last twelve seconds, we should present the top-*k* snippets over the last hour.

We asked ICU clinicians to create datasets for which they know (or at least, strongly suspect) what a correct answer should look like, and to examine these datasets with Snippet-Finder. Fig. 20 shows one such example.

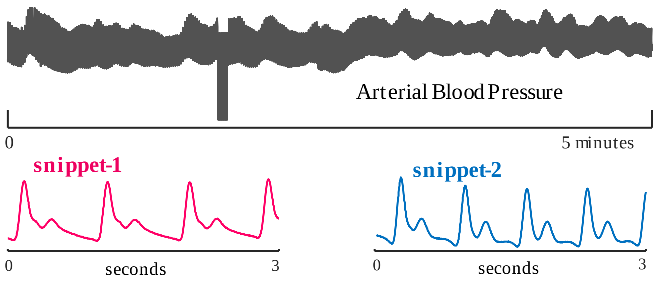


Fig. 20. *top*) The ABP of a 54-year old female. *bottom*) The top-2 four-second snippets discovered in the APB data.

Note that the two snippets reflect different heart rates, at about 70 and 84 BPM respectively. However, this is *not* the reason why they were reported as the top-2 snippets. If we rescale the data to make each beat the same length, we would get similar snippets. To make the difference between the two snippets clearer, in Fig. 21.*left* we show zoom-in sections of the individual beats.

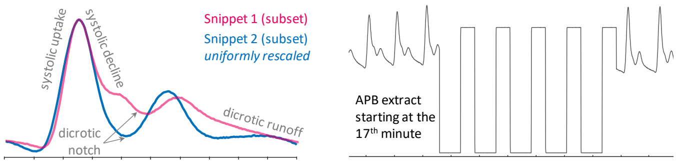


Fig. 21. *left*) A zoom-in to individual beats discovered in the top-2 snippets. *right*) A sensor failure in the original ABP dataset (see also Fig. 20).

Note that beyond differences in periodicity, these beats have very different shapes. We can now reveal the ground truth. This dataset was created by researchers interested in *hyperemia*, and the difference in the two regions was actually *induced* by clinicians who manipulated the patients by changing their orientation on a special bed. Thus, in this case, the researchers can confirm that the results of Snippet-Finder are objectively correct and medically meaningful.

Before moving on, there are two interesting observations for this case study. As shown in Fig. 21.*right*, the original data contains a small region in which the sensor failed to record physiological data, and instead reported a square-wave calibration signal [22]. As noted in Section II.*A*, any *motif*-based definition of a typical pattern would surely report this section. However, our definition rewards both coverage and fidelity, and while this section has perfect fidelity, it has very low coverage.

The second interesting observation is that most of the original dataset has the significant wandering baseline, while our top-2 snippets do not. This is a very desirable and sensible property. A snippet on a rising trend is a better fit to other rising trends, but a poorer fit to falling trends (and vice-versa). However, if we are to have only one snippet per "behavior," a neutral trend is clearly the best compromise and most representative.

E. Case Study: Electrical Power Demand

To demonstrate the versatility of snippets, we consider a dataset that is four orders of magnitude longer than the datasets considered in the previous examples. As shown in Fig. 22, the Italian power demand dataset represents the hourly electrical power demand of a small Italian city for 1,220 days, beginning on Jan 1st 1995.

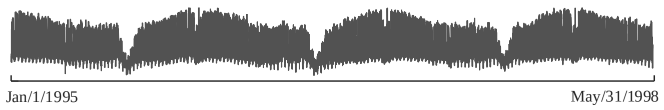


Fig. 22. The Italian power demand dataset contains a little over three years of electrical demand for a small Italian city.

To avoid "over polishing" our query with exact values query length, we search for the top-2 snippets of length 200. This was our quick "eyeballing" guess as to the length of a week, but it is actually about 8.3 days. As Fig. 23 shows, the returned snippets are not constrained by calendar conventions to start on a particular day. An HVAC engineer we consulted suggests that these profiles demonstrate that the respective households have high ownership rates and use of HVAC systems for cooling, but a low use of electrical equipment for heating in the winter.

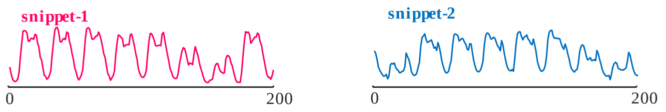


Fig. 23. The top-2 snippets from the Italian power demand dataset. *left*) snippet-1 runs from Monday to Monday (inclusive). *right*) snippet-2 runs from Sunday to Sunday (inclusive).

However, in this case, the real power of these snippets comes from examining the regions they explain, as shown in Fig. 24. In this figure, it is clear that the snippets represent summer and winter regimes respectively.

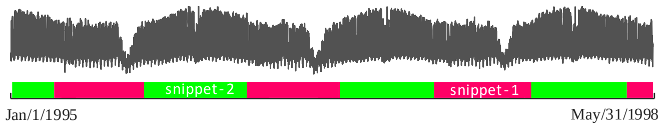


Fig. 24. The Italian power demand dataset. The horizontal bar shows the colors of each snippet onto the original time series.

Before leaving this section, we will take this opportunity to reiterate our discussion of what snippet discovery is *not*. As explained in Section II.*A*, snippet discovery is a completely distinct task from *clustering*. Nevertheless, as a side effect, here we have produced an implicit clustering into seasons. Moreover, a brief survey of the research efforts to explicitly cluster the electrical profile into seasons (see [19] and the references therein) suggests that the algorithms specialized for this task are more complex, and require more domain knowledge; for example, the subsequences must be *exactly* one-week long.

Likewise, snippet discovery is a completely distinct task from *segmentation* [13]. Once again, it happens that in this case, snippet discovery incidentally solves the segmentation problem if we consider the boundaries between the snippets. However, recall that the segmentation problem, as it is typically defined, is tasked *only* with finding boundary locations, not in explaining them, or producing representative patterns [13].

F. Case Study: Biology

As a final example of the utility of snippets, we consider a problem for which our help was solicited by the team of biologists at UCR who recorded the feeding behaviors of sap-sucking insects with an electrical penetration graph (EPG). In Fig. 25 we show three examples of the data in question.

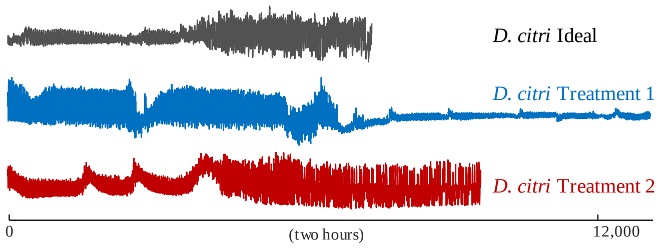


Fig. 25. Three examples of telemetry collected from the three individual Asian citrus psyllids (*Diaphorina citri*). Assuming that the top example is an ideal specimen, and that the bottom two are alternative treatments. How could we quantify the differences (if any) caused by the treatments?

If they define the time series shown in the top of Fig. 25 as being *ideal*, and collect the two other examples under different environmental conditions, then they can ask "*How do T1 and T2 differ from the ideal specimen, if at all?*" The answer does not appear to lie in any single number such as *mean*, *max*, *variance*, *entropy* etc. We hypothesize that changes in behavior may show up in differences in local regions as changes in *shape*. However, given that the approximately five hours of data shown in Fig. 25 represents less than one-thousandth of their archive, visual inspection of the full data archive is clearly intractable.

The reader will readily appreciate that snippets are a potential way to answer this question. As shown in Fig. 26, we can discover the top-2 snippets for each time series, and visually compare them.

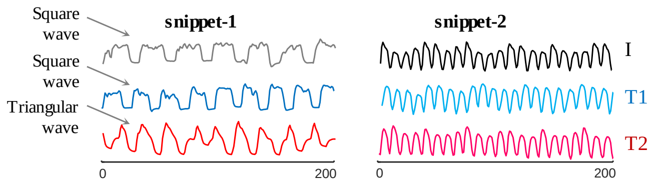


Fig. 26. The top-2 snippets extracted from the datasets presented in Fig. 25. Note that the snippet-2s are effectively identical, but while the snippet-1 for I and T1 are very similar, T2's snippet-1 appears unique.

For all three examples, the 2nd snippets are near identical and represent (to the biologists) very familiar examples of "*derailed stylet mechanics*." The 1st snippet from T1 and I appear to be "*Xylem ingestion*," but the 1st snippet from T2 is a mystery. Its period is similar to that for Xylem ingestion, but the shape of the peaks is unfamiliarly sharp. Is this a biologically significant finding, or is there a more pedestrian explanation, such as a malfunctioning apparatus? This is currently under investigation. However, this experiment shows the utility of snippets in helping to explore and compare large datasets that would otherwise defy human inspection.

V. Conclusions And Future Work

We have introduced a novel primitive called *top-k time series snippets*. We have further shown an algorithm that can robustly find snippets in large datasets, even when corrupted by noise, dropouts, wandering baseline, etc. We have made all our data and code publicly available for the community to confirm and extend our work [24], including a large archive of benchmark datasets that will allow the community to compare new approaches and gauge progress on this task. In future work we plan to generalize our algorithm to streaming settings, in order to maintain snippets over fast-moving streams.

Acknowledgments

We gratefully acknowledge funding from NIH 1R01HD083431-01A1 and from NSF 1544969 and 1510741.

References

[1] Alvarez-Estevez, D. and Moret-Bonillo, V., 2015. Computer-assisted diagnosis of the sleep Apnea-Hypopnea syndrome: A review. *Sleep disorders*, *2015*.

[2] Batista, G.E., Keogh, E.J., Tataw, O.M. and De Souza, V.M., 2014. CID: an efficient complexity-invariant distance for time series. *Data Mining and Knowledge Discovery*, *28*(3), pp.634-669.

[3] Drews, F.A., 2008. Patient monitors in critical care: Lessons for improvement.

[4] Elhamifar, E., Sapiro, G. and Vidal, R., 2012, June. See all by looking at a few: Sparse modeling for finding representative objects. In *CVPR, 2012 IEEE Conference on* (pp. 1600-1607). IEEE.

[5] Forde-Johnston, C., 2014. Intentional rounding: a review of the literature. *Nursing Standard (2014+)*, *28*(32), p.37.

[6] Gharghabi, S., Imani, S., Bagnall, A. and Keogh, E. An ultra-fast time series distance measure to allow data mining in more complex real-world deployments. UCR Tech Report 2018009.

[7] Heldt, T., Oefinger, M.B., Hoshiyama, M. and Mark, R.G., 2003, September. Circulatory response to passive and active changes in posture. In *Computers in Cardiology, 2003* (pp. 263-266). IEEE.

[8] Hendryx, E.P., Rivière, B.M., Sorensen, D.C. and Rusin, C.G., 2018. Finding representative electrocardiogram beat morphologies with CUR. *Journal of biomedical informatics*, *77*, pp.97-110.

[9] Indyk, P., Koudas, N. and Muthukrishnan, S., 2000, September. Identifying representative trends in massive time series data sets using sketches. In *VLDB* (pp. 363-372).

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

[11] Kolhoff, P., Preuß, J. and Loviscach, J., 2008. Content-based icons for music files. *Computers & Graphics*, *32*(5), pp.550-560.

[12] Langohr, L. and Toivonen, H., 2012. Finding representative nodes in probabilistic graphs. In *Bisociative Knowledge Discovery* (pp. 218-229). Springer, Berlin, Heidelberg.

[13] Lin, J.F.S., Karg, M. and Kulic, D., 2016. Movement Primitive Segmentation for Human Motion Modeling. IEEE Transactions on Human-Machine Systems, 46(3), pp.325-339.

[14] Linnarsson, D., et al., 1996. Blood pressure and heart rate responses to sudden changes of gravity during exercise. *American Journal of Physiology-Heart and Circulatory Physiology*, *270*(6), pp.H2132-42.

[15] Lu, L. and Zhang, H.J., 2003, November. Automated extraction of music snippets. In *Proceedings of the eleventh ACM international conference on Multimedia* (pp. 140-147). ACM.

[16] Pan, F., Wang, W., Tung, A.K. and Yang, J., 2005, November. Finding representative set from massive data. In *Data Mining, Fifth IEEE International Conference on* (pp. 8-pp). IEEE.

[17] Papadimitriou, S. and Yu, P., 2006, June. Optimal multi-scale patterns in time series streams. In *Proc' of the 2006 ACM SIGMOD* (pp. 647-658).

[18] Reiss, A. and Stricker, D., 2012, June. Introducing a new benchmarked dataset for activity monitoring. In *Wearable Computers (ISWC), 2012 16th International Symposium on*(pp. 108-109). IEEE.

[19] Rhodes, J.D., Cole, W.J., Upshaw, C.R., Edgar, T.F. and Webber, M.E., 2014. Clustering analysis of residential electricity demand profiles. *Applied Energy*, *135*, pp.461-471.

[20] Rosa, K.D., Shah, R., Lin, B., Gershman, A. and Frederking, R., 2011. Topical clustering of tweets. *Proceedings of the ACM SIGIR*.

[21] Salmenkivi, M., 2006. Finding representative sets of dialect words for geographical regions. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*.

[22] Samaniego, N.C., Morris, F. and Brady, W.J., 2003. Electrocardiographic artefact mimicking arrhythmic change on the ECG. *Emergency medicine journal*, *20*(4), pp.356-357.

[23] Schneider, T.D., 2002. Consensus sequence zen. *Applied bioinformatics*, *1*(3), p.111.

[24] Snippet-Finder project https://sites.google.com/site/snippetfinderinfo/

[25] Wang, X., Xu, Z., Zhang, L., Liu, C. and Rui, Y., 2012, October. Towards indexing representative images on the web. In *Proc' of the 20th ACM international conference on Multimedia* (pp. 1229-38).

[26] Yeh, C.C.M., et al. 2016, December. Matrix profile I: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on* (pp. 1317-1322). IEEE.

[27] Yu, J., Reiter, E., Hunter, J. and Mellish, C., 2007. Choosing the content of textual summaries of large time-series data sets. *Natural Language Engineering*, *13*(1), pp.25-49.

[28] Zhu, Y., et al., 2016, December. Matrix profile II: Exploiting a novel algorithm and gpus to break the one hundred million barrier for time series motifs and joins. In *2016 IEEE 16th International Conference on Data Mining* (pp. 739-748).