Dear Reader: This is an expanded version of the six-page paper that appears in ICDM 2019.

Matrix Profile XV: Exploiting Time Series Consensus Motifs to Find Structure in Time Series Sets

Kaveh Kamgar, Shaghayegh Gharghabi, Eamonn Keogh
Department of Computer Science and Engineering
University of California, Riverside, CA, US
kkamg003@ucr.edu, sghar003@ucr.edu, eamonn@cs.ucr.edu

***Abstract*—In recent years the data mining community has largely coalesced around the idea that many problems in time series analytics essentially reduce to finding and then reasoning about repeated structure in time series. Existing tools can find conserved structure within a single time series (*motifs*) and between pairs of time series (*joins*). However, to date there are no tools to find repeated structure in *sets* of time series, an idea we call *time series consensus motifs* in recognition of their similarity to their discrete analogs in DNA strings. In this work we introduce a definition of time series consensus motifs and a scalable algorithm to discover them in large data collections. We further show that given this new primitive, we can solve multiple higher-level problems in time series data mining. We demonstrate the utility of our ideas with case studies in domains as diverse as animal motion studies, human behavior, medicine and energy disaggregation.**

Keywords*—*time series, motif discovery, conserved patterns

I. INTRODUCTION

There is a growing consensus that many problems in time series analytics essentially reduce to finding and reasoning about repeated structure in time series. For example, segmentation [8], summarization [20], and anomaly detection, can all be framed as algorithms that exploit repeated structure. There are existing tools to find repeated structure within a single time series (*motifs*) and between a pair of time series (*joins*) [19][21]. However, to the best of our knowledge, there are no tools to find repeated structure among *sets* of time series. The analog of this task in discrete strings such as DNA is called *consensus* or *conserved motifs*. These approximately repeated strings are at the heart of much of molecular genetics and are an active area of research [15]. We call repeated structure in sets of time series data, *time series consensus motifs*, acknowledging their similarity to their discrete analogs in DNA strings [2][18]. We can best illustrate the task at hand by considering the analog problem in text strings, using Hamming distance as a proxy for the Euclidean Distance. Consider the following strings:

ooogmisxeturingjkeatanankgokeyomtegooooolotyshemfgsa
lotoogmishejrecytygorcheturingxpoutporstyim
eloterdocegtpogindauryheblaiteturingyoorloungmekeqpbd
itxpeeturingyougrinatecrsthedinarupoooougwcuing

Is there any conserved pattern of length six, that appears in each string? Even in this tiny dataset, the answer, `turing`, is not immediately apparent. If we restrict ourselves to exact matches, this problem can be solved in linear time with a suffix tree. However, if we wish to be invariant to even a single character mismatch, say one occurrence of `turing` is misspelled as `tu`**`n`**`ing`, then the problem explodes in complexity.

Fig. 1 shows an example of the real-valued version of this problem in a dataset of electrooculograph (eye movement) data, created by a volunteer modeling communication by a disabled individual with Locked In Syndrome [7].

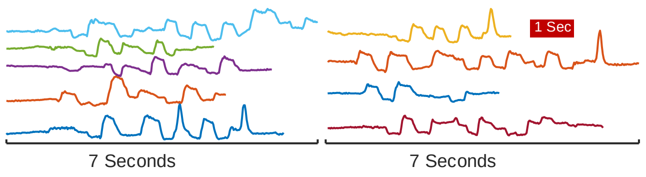

Fig. 1: Nine time series corresponding to different sentences (in Japanese) spelled out by the eye movements of an individual modeling Locked In Syndrome. Just the vertical axis is shown. Is there a well conserved one second long pattern in all nine traces?

We now ask the corresponding question, *is there any conserved pattern, covering a one second interval, which appears in each time series?* The optimal answer, under the definition we propose in this work, is shown in Fig. 2.

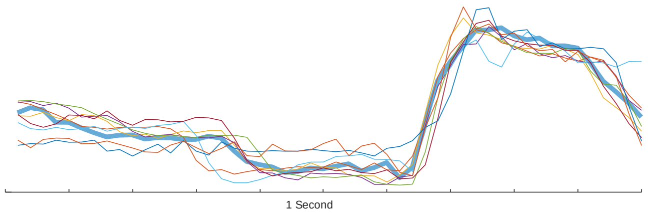

Fig. 2: The one second long consensus motif in the set of time series is shown in Fig. 1 (colors are preserved between plots). One time series is shown with a thicker and bolder line because it is the most *central* or *seed* time series, as will be explained in detail later.

The pattern is unexpectedly well conserved and suggests an underlying mechanism that created faithful conservation. In fact, if we examine the annotation that accompanies this data, we find that the pattern shown in Fig. 2 corresponds to the Japanese Katakana character ア. Moreover, ア is the only character that is common to all nine sentences.

In some cases, finding the most conserved pattern might be an end in itself. However, in other cases, it might just be the beginning of further analytics. Consider the similar experiment conducted with a different individual shown in Fig. 3.

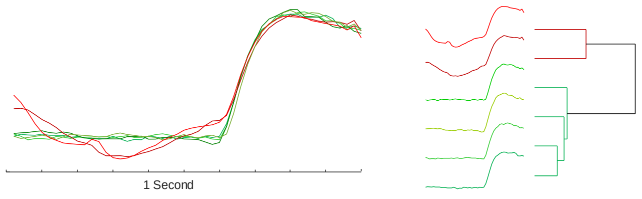

Fig. 3: (*left*) The consensus motif from an individual (different than the person in Fig. 1) modeling Locked In Syndrome. While there is clearly high conservation, our subjective color coding, and the hierarchical clustering (*right*) suggest that there is evidence of hierarchical structure here.

Once again there is a well conserved pattern, and once again we find that it corresponds to a single Katakana character, this time ガ. However, this time there is a strong suggestion that there are two distinct ways in which this individual produced this word. Perhaps this is an example of *orthography* for eye movement communication, just as the written word "read" can be pronounced either *reed* or *red*, depending on whether it refers to the present or the past tense. Or perhaps it simply reflects two sessions, one in which the participant is *communicating* more rapidly and fluidly, the other in which she was tired or distracted. In either case, this example shows that finding such patterns can hint at unexpected regularities.

In this work we introduce *time series consensus motifs* as a novel primitive for time series data mining. The rest of this paper is organized as follows. In Section II we introduce all necessary definitions and notation. We introduce a brute force algorithm to find consensus motifs in Section III, and then in Section IV we address its poor scalability with our novel algorithm, *Ostinato*. We conduct an extensive empirical analysis of our ideas in Section V. We review and contrast with related work in Section VI. Conclusions and directions for future work are presented in Section VII.

II. DEFINITIONS AND NOTATION

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

Definition 1: A *time series* T is a sequence of real-valued numbers $T = t_1, t_2, ..., t_n$ where n is the length of the time series. In a sequence of k time series, the i^{th} time series will be referred to as T^i.

In shape-based time series analysis [8][19], we are typically only interested in local similarity between small sections of the data known as *subsequences*:

Definition 2: A *subsequence* of length m is a sequence of m contiguous elements in a time series T. A subsequence in time series T which starts from position i and ends at position $i + m - 1$ is referred to as $T_{i,m}$.

If we envision all subsequences in the set of k time series as points in m-dimensional space, then there exists an m-ball with minimum distance surrounding each subsequence in each of the k time series which encompasses at least one subsequence from each of the remaining $k - 1$ time series. We call this distance the *radius*:

Definition 3: The *radius* r of a subsequence $T^i_{j,m}$ of time series T^i with respect to a sequence of time series $T^1 \ldots T^k$ is the maximum distance between $T^i_{j,m}$ and its nearest neighbor in each of $T^1 \ldots T^k$.

Fig. 4 offers a visual intuition of this notation.

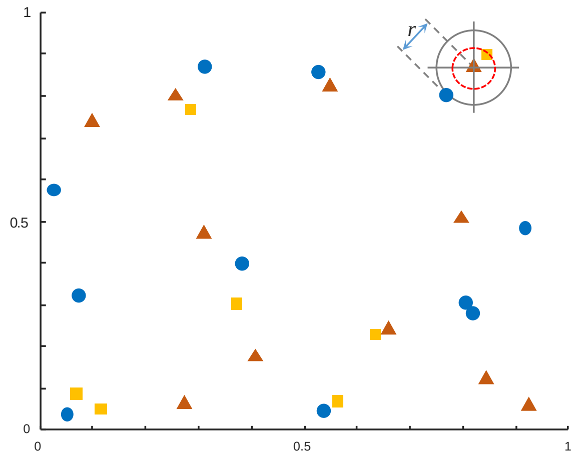

Fig. 4: The subsequences of three time series, A ●, B ▲ and C ■ exist as points in a m-dimensional space. A hypersphere can be centered at each of the subsequences and have its radius r expanded until it includes at least one of each of the time series. The subsequence that has the smallest such r is the consensus motif.

In any set of time series, we expect to see highly varying degrees of conservation across different subsequences as demonstrated by their induced radii. We refer to the time series subsequence with the smallest such radius as the *time series consensus motif*:

Definition 4: The *time series consensus motif* is the subsequence taken from one of a sequence of k time series $T^1 \ldots T^k$, which possesses the smallest *radius* of any subsequence appearing in any of time series $T^1 \ldots T^k$.

As Fig. 2 and Fig. 3 suggest, this definition assumes that some highly conserved structure is present in all k time series considered, but this assumption may not always hold true. This is especially the case for our initial forays into a data collection when doing exploratory data mining. For clarity, suppose that we augmented the four discrete strings shown in the introduction with a fifth *outlier* string:

atgcatgcatgcatgcatgcatgcatgcatgcatgcatgcatgcatgc

Clearly this string does not contain the otherwise conserved string `turing`, however we may still wish to find this pattern, which appears in most of the strings. Returning to the same problem in time series, up until this point, we have assumed that the number of time series in our set is equal to the number that must be considered to compute a radius. Definition 5 generalizes Definition 4 to allow the case where the k time series, which determine a subsequence radius, may be chosen as any subset of a set of P time series. This notion allows us to omit consideration of P - k potential *outlier* time series.

Definition 5: The *k of P* consensus motif is the consensus motif with the smallest radius that can be found using any subset of k time series from a sequence of P time series.

Choosing k time series from P allows us to simultaneously exclude $P – k$ *outlier* time series.

As we will show, when computing the radius of each subsequence, we can prune the search space by computing a lower bound. We accomplish this using an *A*-to-*B* join or *ABJoin*.

Definition 6: An *ABJoin* is a meta time series annotating each overlapping subsequence $A_{i,m}$ in A, with the distance to its nearest neighbor in B.

ABjoins were introduced in [19], using the notation J_{AB}. Below we will detail the algorithms for time series consensus motifs. The extension to the discovery of the *k of P* case is obvious. We provide code for the *k of P* version on the website [22].

III. BRUTE FORCE CONSENSUS MOTIF DISCOVERY

In order to find a consensus motif as defined in Definition 4, we need to find which subsequence from the k time series induces the smallest radius. To help explain our proposed solution, we begin by considering a simple brute force solution.

We begin by concatenating all k time series into a single time series T, with *null* markers in between them to mark the transitions from one time series to the next. Let us denote the length of this long time series as N. We can use this long time series to compute a distance matrix D containing every pairwise distance between all z-normalized subsequences of length m in T. Here each candidate subsequence represents a possible consensus motif.

We can then search for the consensus motif by keeping track of a best-so-far candidate motif for O(N) steps. At step j, we find the minimum row value of D in column j and compare that to our best-so-far. We require O(N^2) distance calculations to populate D, followed by O(N^2) comparison step. Naively, each distance calculation requires O(m) operations, and the distance matrix as shown in Fig. 5 requires O(N^2) space.

Fig. 5 depicts this for a toy example with k = 3, the time series T^1, T^2 and T^3. Note that the constituent time series can be of different lengths, so long as each has a length greater or equal to the length m, which is the user's choice of motif length.

We can use the methods of computing ABJoins developed in [19][21] to make our first improvement to time complexity. We note that given a particular choice of j, we can find the radius of all subsequences in time series T^j by finding the elementwise maximum of ABJoin(T^j, T^1)…ABJoin(T^j,T^k) [19]. This reduces our time and space requirements to O(N^2) operations and O(N) memory.

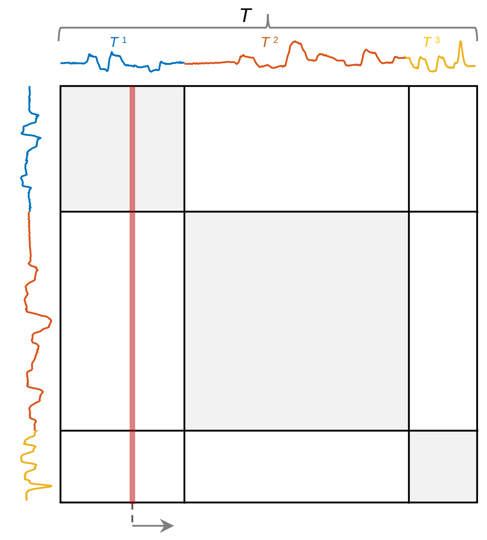

Fig. 5: We can search for the consensus motif by sweeping across all columns of the pairwise distance matrix (visualized by the red line) and finding the minimum value in each of the k (here, k = 3) regions. The maximum of these k values is the radius of the corresponding subsequence, and the smallest such radius is found at the consensus motif.

Referring back to Fig. 4, the brute force algorithm can be seen as visiting each point in the m-dimensional space, and finding its nearest neighbor in each of the other "colors", recording the maximum such distance as the radius r. Each ABJoin represents a reduction over the columns spanning a single tile in Fig. 5. For completeness, and because we will use it later as a starting point to explain our new algorithm, we take the time to outline the brute force algorithm in TABLE 1.

The description of our brute force algorithm uses a distance matrix representation to describe our search space, as visualized in Fig. 5. We interpret T as a time series formed by concatenation of time series $T^1…T^n$. Here D is a distance matrix containing the distance between every pair of subsequences of length m in time series T.

In TABLE 1, line 1 initializes the *best-so-far* radius, time series index, and subsequence index. The time series and subsequence indices indicate the location of the subsequence with minimum known radius. In line 2, we loop over each time series $T^1…T^k$. Line 3 initializes an array for the radius of each subsequence in time series T^j. Line 4 begins a loop over $T^1 \ldots T^k$ excluding T^j. Line 5 updates the radius calculations for T^j using an ABJoin algorithm. In Line 6, we find the subsequence with the minimum radius in T^j. Lines 7 and 8 compare the minimum radius found in any subsequence in T^j to the minimum radius found overall and updates our *best-so-far* candidate. In line 9, we return the radius, time series index, and subsequence index of the subsequence with the smallest radius in $T^1 \ldots T^k$.

TABLE 1: THE BRUTE FORCE CONSENSUS MOTIF ALGORITHM

```
Function: BruteForceConsensusSearch($T^1 \ldots T^k$, $D$, $m$)
Input: $T^1 \ldots T^k$ - Sequence of Time series
       $m$ - Subsequence Length
Output:
       bsfRad - best so far radius which is found
       tsIndex - index of time series with best so far radius
       ssIndex - index of subsequence with best so far radius
1 {bsfRad, tsIndex, ssIndex} ← {inf, 0, 0}
2 for $j$ ← 1 to $k$
3   Radii ← zeros(length($T^j$) − $m$ + 1)
4   for $i$ ← 1 to $k$ except $j$
5     Radii ← elementwise_max(Radii, ABJoin($T^j$, $T^i$))
6   {minRadius, minRadIndex} ← {min, argmin} (Radii)
7   if minRadius < bsfRad
8     {bsfRad, tsIndex, ssIndex} ← minRadius, $i$, minRadIndex}
9   return {bsfRad, tsIndex, ssIndex}
```

IV.OSTINATO: CONSENSUS MOTIF DISCOVERY

Having shown the brute force algorithm for consensus motif discovery, we are now in a position to show how to speed it up.

A. An Exploitable Observation

Consider Fig. 6, which shows the same toy dataset shown in Fig. 4, but before the consensus motif was discovered. Recall that in TABLE 1 line 1, the value of *best-so-far-r* is initialized to *inf*. In line 2, we begin to process each subsequence. In Fig. 6.*top* we show that B_1 is the first candidate subsequence to be completely evaluated, and thus updated for *best-so-far-r* from infinity to a finite number.

In Fig. 6.*center* we show that B_2 is the next point to be evaluated. We begin by computing the distance from the subsequence B to its nearest neighbor in A (red dashed circle), which happens to be A_{17}. Naively we would then compute the distance from the subsequence B to its nearest neighbor in C. However, we can easily see here that this will be fruitless. Because $B_1.r$ is less than the distance between B_2 and its nearest neighbor in A, the distance to C is now inconsequential. Even if it was zero, we know we still have $B_1.r < B_2.r$. Thus we can admissibly prune $B_1.r$ from further consideration.

B. Ostinato: Fast Consensus Motif Search

We call our consensus motif search algorithm *Ostinato*. Ostinato first computes a lower bound for each candidate subsequence considered by the brute force method. This sequence of lower bounds is used to both order our search, and to admissibly prune unpromising candidates.

Ostinato begins by computing a lower bound on the radius of each subsequence in each time series in the form of an ABJoin. This covers all comparisons over a single block diagonal section as displayed in Fig. 5. Using a fast ABJoin method [19], this requires $O(N^2/k)$ operations to cover the k time series.

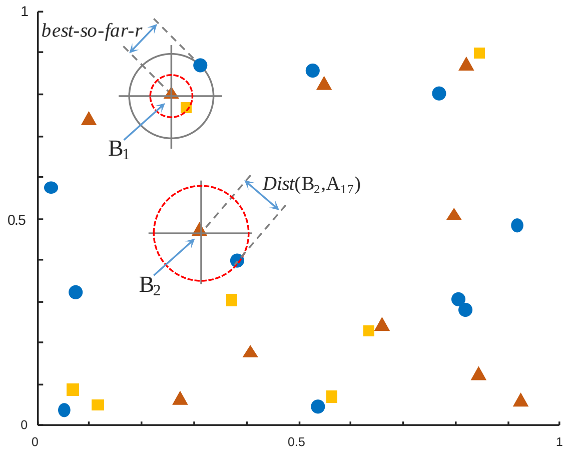

Fig. 6: (cf. Fig. 5) A visual intuition of the time series consensus motif discovery process. Key: A ●, B ▲, C ■.

This first step is followed by k searches corresponding to the k time series. In each search, candidate subsequences from a single time series are ordered by increasing the lower bound on their radii. We find the best possible candidate from within that time series by evaluating candidates until the known lower bound on the radius of the next candidate exceeds *bsfRad*, our best current candidate. At that point we admissibly prune the remaining candidates in that time series from our search space. The algorithm is outlined in TABLE 2 below.

TABLE 2: THE OSTINATO ALGORITHM

```
Function: Ostinato($T^1 \ldots T^k$, $m$)
Input: $T^1 \ldots T^k$ - Sequence of Time series
       $m$ - Subsequence Length
Output: bsfRad - best so far radius which is found
        tsIndex - index of time series with best so far radius
        ssIndex - index of subsequence with best so far radius
1  {bsfRad, tsIndex, ssIndex} ← {inf, 0, 0}
2  for $j$ ← 1 to $k$
3    $h$ ← ($j$ + 1) if ($j$ < $k$), else 1
4    MP ← ABJoin($T^j$, $T^h$)
5    SI ← sortAndIndex(MP)
6    for each value $q$ in SI
7      radius ← $MP_q$
8      if radius ≥ bsfRad
9        break loop
10     for $i$ ← 1 to $k$ except $j$ and $h$
11       radius ← max(radius, min(EuclideanDist($T^i$, $T^j_{q,m}$)))
12       if radius ≥ bsfRad
13         break loop
14     if radius < bsfRad
15       {bsfRad, tsIndex, ssIndex} ← {radius, $j$, $q$}
16 return {bsfRad, tsIndex, ssIndex}
```

In line 1, we initialize the radius, time series index, and subsequence index of our consensus motif using sentinel values. Line 4 computes a lower bound on the radius of every subsequence in time series T^j using an ABJoin, and line 5 produces an index map of T^j in order to increase radius lower bound. We initialize the radius of each subsequence to its greatest known lower bound in line 7. If the lower bound on the radius of any subsequence exceeds that of any already computed radius, we admissibly abandon the search over remaining candidates in T^j in line 9. Line 10 loops over each time series, excluding the one which contains our current candidate. In lines 12 and 13, we abandon our current candidate if its maximum known lower bound exceeds the radius of any other previously evaluated candidate. Line 11 updates the radius of our current candidate subsequence using the distance between it and its nearest neighbor in the next time series, with the last wrapping around to the first.

Lines 14 and 15 update the current *best-so-far*. Finally, line 16 returns the candidate with the smallest radius as a triple consisting of its radius, the index of the time series where it was found, and its subsequence index within the time series.

For simplicity, we have shown just the top-1 algorithm. The algorithm can be generalized to compute the top-*k* motifs by creating a top-*k* *best-so-far* list in Line 1, adjusting the comparison and update methods in lines 8, 12, 14, and 15, and returning the full list in line 17.

V. EXPERIMENTAL EVALUATION

To ensure that our experiments are reproducible, we provide a website containing all data, code, and spreadsheets for the results [22]. This commitment to reproducibility extends to all the examples in the prior sections.

Before showing formal experimental evaluations of the scalability and robustness of our ideas, we will introduce three examples of consensus motifs in very diverse domains to show the generality of our ideas.

A. *Demonstrations of Generality*

V.A.1 Electrical Power Demand

Data mining of household electrical energy demand is an active research area [14]. The fundamental problem is that the meters measure the aggregate energy consumption of the entire building. However, appliance-by-appliance consumption information is much more valuable than aggregate data for a variety of tasks, including reducing energy demand and improving load forecasting for the electrical grid. Thus, much of the research in this area focuses on the task of *disaggregation*, teasing out the demand patterns of individual devices. In Fig. 7 we show a random two-day trace from this data type. The reader will appreciate that it is noisy and complex domain.

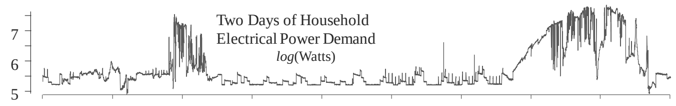

Fig. 7: Two days of electrical power demand from the REFIT Electrical Load Measurements dataset, H-1 [14].

To demonstrate that we can find semantically meaningful consensus motifs even in the face of such noisy data, we conducted the following experiment. We extracted seven sample time series with the length of 20,000 data points. The slightly variable sampling rate of this dataset is roughly eight to ten seconds. Thus, 20,000 data points is about two days. We searched for the consensus motif with a length of 800, or approximately two hours. In Fig. 8 we show the consensus motif discovered.

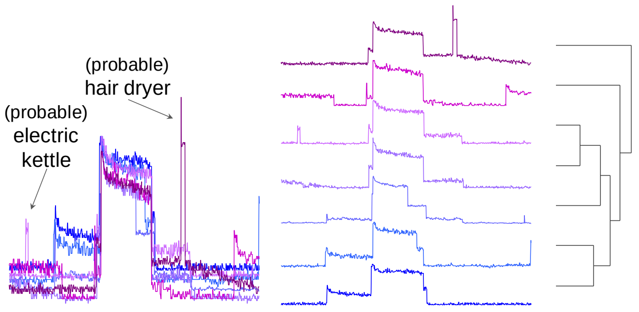

Fig. 8: *left*) The consensus motif discovered in the power demand data set. The overall shape is conserved, in spite of noise and spikes caused by short-lived high-power demands (hair dryer, kettle). *right*) A clustering of the patterns makes their similarity more evident.

Could we do something similar with classic motif discovery? Classic motif discovery asks, *what pattern is conserved anywhere in this data[19]?* In contrast, our query asks, *what pattern happens at least once every two days?* This latter query maps better to most disaggregation algorithms, as it finds quotidian behaviors rather than well-conserved but possibly rare behaviors, such as putting a dishwasher into its self-cleaning cycle every week.

V.A.2 Mitochondrial DNA

It is well understood that we can often produce better clusters by only clustering the subset of the data that is most amenable to clustering. There are several ways that we can attempt to resolve this *chicken-and-egg* paradox, and here we show that consensus motifs offer such a possibility.

To see this, let us consider a domain for which we can obtain unambiguous ground truth. DNA is normally processed using string comparisons. However, it can sometimes be fruitful to convert it into a real-valued time series. Here we use the simple conversion algorithm outlined in Appendix A to convert the Mitochondrial DNA (mtDNA) of four randomly chosen animals to time series. The mtDNA of all species are just over 16,000 data points. We truncate them to exactly 16,000 to allow the clustering under Euclidean distance in Fig. 9.*left*.

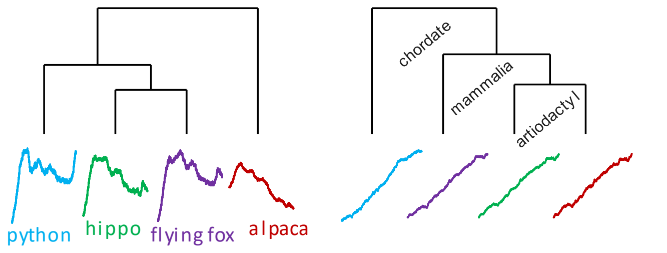

Fig. 9: The clustering of the mtDNA of *Python bivittatus*, *Hippopotamus amphibius*, *Pteropus scapulatus*, *Lama pacos*. *left*) using all the mtDNA gives incorrect results. However, using just the consensus motif of length 1,000 (*right*) gives the correct taxonomic relationship.

This clustering is clearly incorrect. For example, it suggests that the hippo is more closely related to the python than to the alpaca, which is a fellow even-toed ungulate (*Artiodactyla*). To address this problem, we can simply compute the consensus motifs of a short subset of the data. Here we arbitrarily choose a subsequence length of 1,000 and use only the *discovered* subsequences to cluster the data. As Fig. 9.*right* shows, this produces reasonable results.

V.A.3 Insect EPG Telemetry

The Asian citrus psyllid (*Diaphorina citri*) is an insect vector of the pathogen that causes citrus greening disease, causing billions of dollars of loss to the citrus industry in the last decade. To understand the behavior of this insect, entomologists use a device called an electrical penetration graph (EPG) to collect data reflecting the insect's interaction with plants [11][12]. As shown in Fig. 10.*left* such data is typically complex and noisy.

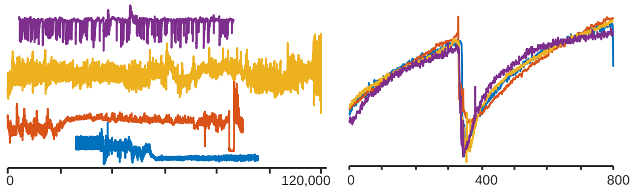

Fig. 10: *left*) Four time series of insect telemetry. *right*) The top eight-second-long consensus motif is well conserved and corresponds to "phloem salivation" behavior.

One basic question entomologists are interested in is, *what insect behaviors are conserved when something in the environment has been changed [12]?* For example, it was recently discovered, by manual inspection of EPGs, that the feeding behavior of the white-backed planthopper (*Sogatella furcifera*) could be changed by infecting rice plants with a special virus [11]. Such findings are useful because they sometimes suggest a control mechanism [12]. As shown in Fig. 10.*right*, our algorithm can recover conserved behaviors in this data in spite of how noisy and large it is.

To demonstrate a scenario in which consensus motif discovery could be useful to a research entomologist, let us consider EPG from a different insect, a silverleaf whitefly (*Bemisia tabaci*). As Fig. 11 shows, this is a tiny insect, about the size of the period at the end of this sentence. In spite of its small size, this insect is an important agricultural pest. While the silverleaf whitefly had been known in the United States since 1896, in the mid-1980s a virulent strain (strain B) appeared in poinsettia crops in Florida. Less than a year after its initial identification, strain B was found in tomatoes and other fruit and vegetable crops. The silverleaf whitefly caused over one hundred million dollars in damage to Texas and California agricultural industries within five years [6]. More recent estimates suggest that this insect may cause over a billion dollars of crop damage annually.

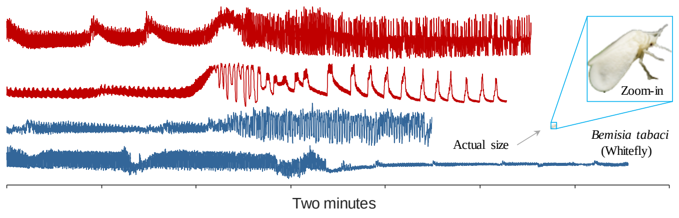

Fig. 11: *right*) A life-sized image of a silverleaf whitefly shows how incredibly small it is. In spite of its size, skilled entomologists are still able to attach it to an EPG apparatus (using a solid gold wire just 12.5 µm in diameter), and record time series data under various conditions (*left*).

There is an active worldwide community attempting to understand and ultimately control this insect. As part of this effort, they often record data under two or more conditions, e.g. light and dark, hot and cold, and humid and dry environments.

One question that remains the subject of some controversy is whether the behavior of the insects is changed by the presence of oils mixed with *citronellal*, the chemical that gives citronella oil its distinctive lemon scent. To test this, we obtained seven recordings of the insect interact with a plant coated with a thin film of citronellal-infused oil and ten control recordings of the insect on untreated plants. We found the one-second consensus motif of each group independently. Fig. 12 shows the results.

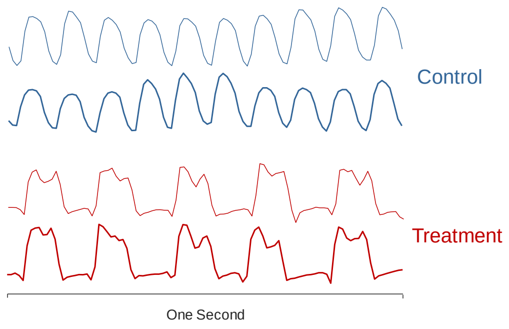

Fig. 12: The consensus motif (bold) and one additional sample from the motif set for the two classes.

The results are highly suggestive. For the control, the consensus motif seems to be classic `passive phloem ingestion`. In contrast, the treatment data produced a consensus motif that does not seem to have been observed in the literature [13]. It is important to disclaim that we are not making any biological claims here. Our datasets are too small to support statistical significance testing. Our point is simply to show how our tools may be used to investigate important problems and produce potentially actionable information.

B. Robustness of Consensus Motifs

The experiments above (especially Fig. 8 and Fig. 10) suggest that consensus motifs can be discovered in noisy and complex datasets. However, here we stress test the definition to see how much noise our approach can handle.

We took random instances from class one of the Mallat dataset [5], each of which is of length 1,024, and embedded each of them within a random walk of length 65,536. Fig. 13 shows an example. Given ten such time series, can we recover the embedded pattern? We count any extracted pattern that overlaps with an embedded Mallat pattern by at least fifty percent as a true positive. The possibility of this happening by chance, i.e. the *default rate*, is only 3.1 percent.

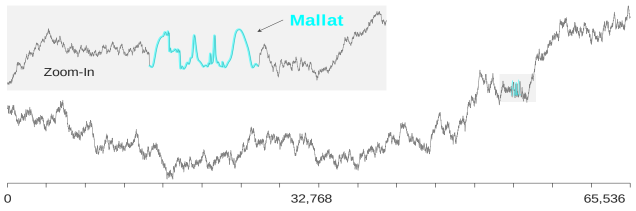

Fig. 13: A random walk of length 2^{16} with a single instance from the UCR archive Mallat dataset embedded. (inset) A zoom-in of the embedded pattern.

We find that Ostinato can easily recover the embedded pattern. In order to stress test it, we repeated the experiment, adding increasing amounts of Gaussian noise, until we failed to recover the embedded pattern. Fig. 14 shows the results.

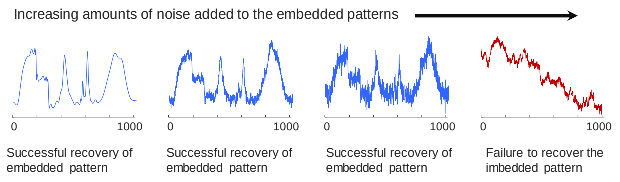

Fig. 14: The recovered most central or *seed* subsequence as we iteratively add increasing amounts of Gaussian noise to the data. The rightmost blue pattern shows the *most* amount of noise our definition can tolerate.

This result suggests that our definition is robust to significant amounts of noise.

It is also natural to ask how sensitive the definition is to its only parameter, the length of the subsequences, m. We repeated the basic experiment described above, attempting to recover the embedded pattern from ten randomly generated instances of the time series shown in Fig. 13. Instead of just testing with the correct pattern length of m = 1,024, we searched with increasingly shorter and longer lengths. We discovered that we can recover the embedded pattern for any value of m from 74 to 1,076, suggesting that the definition is not too sensitive to its only parameter. Note that this flexibility is asymmetric. If our domain knowledge or experience suggests that there might be conserved structure of length L, we can be very conservative and choose a much shorter length, say $m = L/2$ or even $m = L/4$, and still expect to find the conserved structure. However, if we are too liberal and choose a longer length, we may have a fruitless search. As we shall show in the next section, Ostinato is so fast that we can quickly search over multiple lengths and choose the best motifs by eyeballing an objective score function.

C. Scalability of Consensus Motif Discovery

To measure the scalability of our proposed algorithm, we performed the following experiments. We created ten time series like the one shown in Fig. 13, but with varying lengths, and we then measured how long it takes to find the top-1 consensus motif of length 1,024. We compared three approaches:

- **Brute Force Consensus Search**: As described in TABLE 1. While just a strawman, we made significant effort to produce a highly optimized implementation.
- **STAMP**: We adapted this algorithm from [19] to solve the task at hand. STAMP makes the algorithm's performance independent of the length to the motifs. We created an optimized version of STAMP for this task to match the implementation details of Ostinato wherever possible. Thus our version of STAMP is significantly faster than the original code[1].
- **Ostinato**: Our proposed algorithm.

Fig. 15 shows the results.

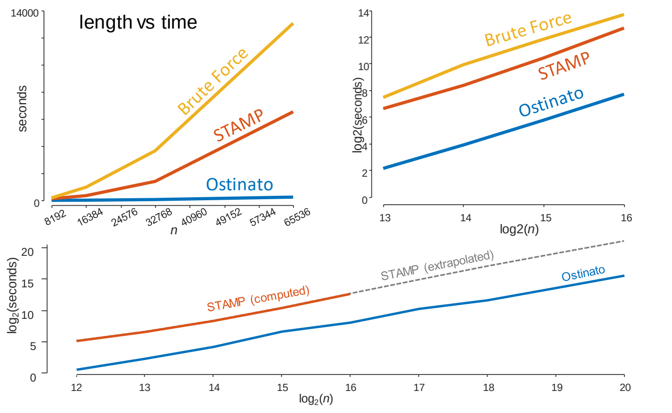

Fig. 15: A comparison of three algorithms for consensus motif discovery. *left*) The time taken to find the top-1 motif in increasing longer datasets. *right*) The differences in performance can be better appreciated in a loglog plot. *bottom*) Only Ostinato can realistically consider datasets with millions of data points (STAMP experiments that required more than 24 hours are carefully extrapolated).

We deliberately chose the best possible case for STAMP, with the length of each time series (n) being a power of two.

[1] The original STAMP algorithm is only directly comparable to Ostinato in the special case of $k = 2$. We both optimized STAMP and generalized it to arbitrary k.

Nevertheless, by the time we consider n = 65,036 we are more than sixty-three times faster, and the gap continues to grow as we see larger and larger datasets.

We demonstrate the scalability of Ostinato with a variable number of time series and with variable subsequence length in Fig. 16.

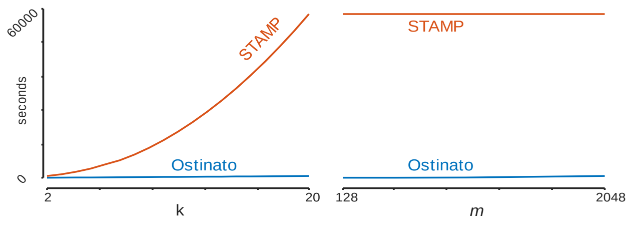

Fig. 16: *left*) Scalability with the cardinality of the set of time series, *k*. Beyond *k* = 10, the time for STAMP was extrapolated, *right*) Scalability with the motif length, *m*.

In Fig. 16.*left*, with n fixed at 65,036 and m at 1,024, we measure how the algorithms scale as k increases from 2 to 20. In Fig. 16.*right*, with n fixed at 65,036 and k = 10, we measure how the algorithm scales as m goes from 128 to 2,048.

Note that Fig. 16.*right* reaffirms that the time taken is independent of the length of the motifs, a highly desirable property, which we inherit from our use of a STOMP-like algorithm as a core subroutine [21].

D. The Utility of *k of P* Consensus Motifs

To test the utility of the *k of P* variant of consensus motifs, we performed the following experiment. We created a dataset with exemplars like the one shown in Fig. 13. For both the *k of P* variant of consensus motifs and the regular (i.e. *P* of *P*) version, we tested if it could recover the embedded motif from these five time series. We then added an increasing number of random walks without embedded patterns, to test how the presence of spurious data affects the algorithms. For the *k of P* variant, *P* grows from five to ten, and *k* remains fixed at five. We repeated the process twenty times, reporting the results in TABLE 3, which are averaged over twenty runs.

TABLE 3: SENSITIVITY TO SPURIOUS DATA

P =	5	6	7	8	9	10
(k hardcoded to 5) *k of P*	19/20	19/20	18/20	16/20	16/20	16/20
P of *P*	19/20	0/20	0/20	0/20	0/20	0/20

In fairness, we could improve the results for the *P* of *P* algorithm in several ways. Smoothing the data may help, as would choosing a smaller value for *m*. Nevertheless, the results support the utility of the *k of P* variant of consensus motifs.

The experiment above shows that the *k of P* variant can be more robust, but it assumes we know the correct value for *k*. What if we do not? To consider this issue, we revisit the data type considered in Fig. 1 and Fig. 2. For another volunteer's session, she created seven time series with a maximum length of 350. We created an additional twenty random walk time series of that length, for a total *P* of 27. We then fixed *P* to 27, and computed *k of P* for every *k* from 2 to 27, recording the radius at each step Fig. 17 shows the result.

The inflection point in Fig. 17 strongly suggests that the correct value for *k* is at seven. Moreover, we confirmed that the seven subsequences all come from the real electrooculographs, and that they correspond to the Katakana character モ.

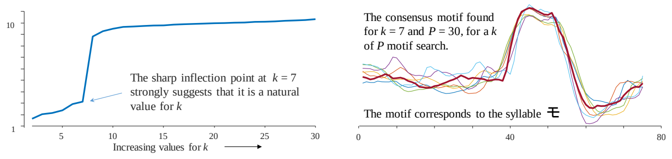

Fig. 17: *left*) The radius of the *k of P* consensus motif for every *k* from 2 to 27. The plot suggests the best *k* is seven. *right*) The suggested *k* = 7 motif corresponds to the syllable モ, the only syllable that appears in all seven traces.

Case Study: Quantifying Parkinson's Disease

Parkinson's Disease (PD) is a neurodegenerative disease which affects gait and mobility. As hinted at in Fig. 18, one tool that clinicians use to assess the severity of the disease is telemetry from a vertical ground reaction force device [10].

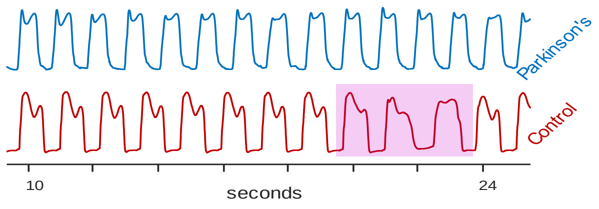

Fig. 18: Time series snippets from vertical ground reaction force device recordings, left foot only. *top*) An individual with moderate Parkinson's exhibits some variability in her signal. *bottom*) A healthy individual has a more regular gait, *except* at the highlighted location.

Clinicians can visually inspect the data and use the degree of variability to quantify the progression of the disease. The severity is typically qualified with the Hoehn and Yahr (HY) scale [1], with 0 indicating "no functional disability" and 3 indicating "mild to moderate disability".

Objectively quantifying the gait can be difficult. One can envision many ways to score variability, however as shown in Fig. 18.*bottom*, when we examine the full traces of the healthy individuals, we typically find some irregular regions. The explanation for such regions is prosaic: the device used to measure the ground force is limited in length, and the patient must turn around when she reaches the end of the apparatus. The attending physician can annotate and ignore such regions, but large-scale retroactive studies typically do not have access to these annotations.

Thus, we should measure the variability of only the least variable regions. The reader will appreciate that consensus motifs offer a direct way to do this. Our idea is to divide the time series into equal sized chunks of length L and measure the radius of the top-1 consensus motif of length m. Because we will set $m \ll L$, our measure ignores the spurious irregularities caused by the patient turning around at end of the apparatus.

To test our intuition, we consider the Parkinson Disease dataset provided by Hausdorff group [10], which is publicly available in PhysioBank [9]. This dataset consists of gait force profiles of 23 patients with idiopathic PD (HY 3 scale) and 92 healthy controls. The data was recorded at a sampling rate of 100 Hz. Because the shortest length of subject walking is 3,000, we truncate all the other subject's data to that length. We concatenate the left and right foot force profiles, then we divide them into six parts, each with L = 1,000. We find the radius of the consensus motif with length m = 300 in these partitions. Note that we choose 300, because it is round number, which approximately matches the length of an average step Fig. 19 summarizes the results.

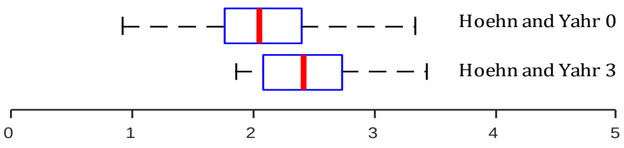

Fig. 19: Quiver plots that summarize the distribution of radii of consensus motifs of the gait from both healthy (HY-0) and sick (HY-3) individuals.

The mean for patients with a HR score of 3 is slightly greater than the upper quartile for the patients with a HY score of 0. While we made no attempt to tune L or m here, some domain knowledge could potentially further improve our results. We must disclaim that we are making no medical claims here. This example just serves as an illustration of the type of higher-level problems that consensus motifs can be applied to.

E. Case Study: Ostinato for Segmentation

We conclude the experimental section with an additional example of a higher-level algorithm that exploits the consensus motifs as a subroutine. As before, our goal is not to exhaustively tackle a new problem, but to show the potential utility of considering the consensus motifs as a primitive. The task we address is segmentation [8]. Suppose that we have a long time series that at some point reflects a change in the dynamics of a system being measured. Can we detect when the change happened? We propose the simple algorithm sketched out below. Full code is available at [22].

Divide the time series into R equal-size regions, S_1, S_2,…,S_R. For example, in Fig. 20.*top* the time series T is of length 40,000, and we can divide it into R = 40 subsequences, each of length 1,000. Let us consider a value for m that is much less than the region of 1,000, say 200.

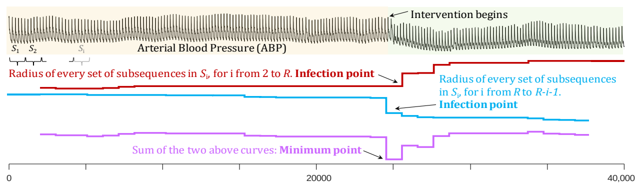

Fig. 20: *top*) A time series showing the APB of a patient. At time 25,000, a clinician changed the patient orientation. *bottom*) By examining the change in radius as we consider increasingly large subsets of all the regions, working both left-to-right and right-to-left, we can get clues as to the location of the change of system behavior.

Suppose that we measure the radius produced by every growing set of subsequences in {S_1 to S_i}, for i from 2 to R. What would we expect to see? We should expect that this curve grows slowly as each newly added subsequence includes data from the same behavior. However, the curve will grow rapidly, producing an inflection point as we encounter the first subsequence that includes data from the new and different behavior. As the red curve in Fig. 20 shows, this is exactly what we see.

In this example, the sum of this red curve with its "mirror image" (blue) curve correctly minimizes at the location where the system changes. In [22] we show that this simple algorithm is effective on dozens of datasets from diverse domains.

VI. RELATED WORK

We have relegated the discussion of related work to the end of this paper, so the reader has a better appreciation of the issues involved.

The bioinformatics literature offers several definitions for consensus sequences/motifs [15][18], as a nucleotide sequence of DNA/RNA, or an amino acid sequence of proteins. However, these definitions do not generalize to real-valued data. For example, the discrete definitions have the property that the error the analog to our *radius*, cannot decrease as the length increases. In contrast, in the real-valued case, because of z-normalization, a longer motif could have a smaller radius than a shorter motif length. Apart from anything else, consensus sequences are typically only 6 to 8 base-pairs long, whereas we have shown the need to find time series consensus motifs at least two orders of magnitude longer.

In [15] the authors use the term "Consensus Sequence Motifs" in a time series context. However, they are working with a domain-dependent transformation of the time series, a high-level abstraction of the data (i.e. low/medium/high), and their method discovers motifs of intervals. The work is completely orthogonal to our domain independent motifs discovered in the raw data.

There is significant work on finding or creating representative patterns in sets of time series, using averaging [16]. However, these works attempt to explain all the data, not discover just the conserved data.

Thus, to the best of our knowledge, there is no work on finding conserved structure in arbitrary sets of more than two real-valued time series.

VII. CONCLUSIONS

We motivated the need for, and then introduced, the first known algorithm for finding repeated structure within sets of time series. With multiple case studies and experiments on real datasets from diverse domains, we demonstrated that our definitions and algorithms are robust enough to recover conserved data, even in the presence of significant amounts of noise or spurious data.

Moreover, we have shown that our algorithm is surprisingly tractable, and we can find conserved data in datasets with tens of millions of datapoints in reasonable time. In particular, for all our experiments with electrooculography, power demand, insect EPG, gait and ABP, our experiments ran much faster than real-time. That is to say, if the data represents *X* seconds, Ostinato took less than *X* seconds to find the motifs.

There are several directions for future work. Ostinato is currently a batch algorithm, but [19] has shown that it can be fruitful to produce *anytime* algorithms for motif discovery problems. In addition, Ostinato always produces some motif, even in random data. It would be useful to have an objective score or significance test that reflects the quality or significance of the motifs [17].

Finally, as our examples *Case Study: Quantifying Parkinson's Disease* and *Case Study: Ostinato for Segmentation* hint, we believe that there are many possibilities for novel higher-level algorithms that use consensus motifs as a primitive. The fact that there are hundreds of algorithms that exploit consensus motifs/consensus sequences for discrete strings such as nucleotides or amino acids [2][3][15][18], suggest that our real-valued version may find many unexpected uses in the time series data mining community.

ACKNOWLEDGEMENTS: We gratefully acknowledge NSF awards DGE 1631776, CNS 1544969, in addition to Google, Mitsubishi and NetApp.

REFERENCES

[1] O. Afsar, U. Tirnakli, and N. Marwan, "Recurrence Quantification Analysis at work: Quasi-periodicity based interpretation of gait force profiles for patients with Parkinson disease", Scientific Reports 8.1 (2018): 9102.

[2] A. Apostolico, "Monotony of Surprise and Large-Scale Quest for Unusual Words", Journal of Computational Biology V10, 3-4, 2003 pp. 283-311.

[3] V. Boeva. "Analysis of Genomic Sequence Motifs for Deciphering Transcription Factor Binding and Transcriptional Regulation in Eukaryotic Cells". Frontiers in Genetics. Vol 7, 2016.

[4] A. Camerra, et al. "Beyond one billion time series: indexing and mining very large time series collections with *i*SAX2+". Knowl. Inf. Syst. 39(1): 123-151 (2014)

[5] Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A. and Batista, G., "The UCR time series classification archive", URL www.cs.ucr. edu/~eamonn/time_series_data.

[6] Y. F. Fan and F. Petitt, (1998). "Dispersal of the broad mite, Polyphagotarsonemus latus (Acari: *Tarsonemidae)* on *Bemisia tabaci* (Homoptera: *Aleyrodidae*)". Experimental and Applied Acarology. 22 (7): 411–415

[7] F. Fang and T. Shinozaki, "Electrooculography-based continuous eye-writing recognition system for efficient assistive communication systems", PloS one, 13(2).

[8] S. Gharghabi, Y. Ding, C.C.M. Yeh, K. Kamgar, L. Ulanova, and E. Keogh, "Matrix Profile VIII: Domain Agnostic Online Semantic Segmentation at Superhuman Performance Levels", ICDM 2017 IEEE International Conference on Data Mining, pp. 117-126

[9] A.L. Goldberger, et al. "PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals" Circulation 101.23 (2000): e215-e220

[10] J. Hausdorff, Z. Ladin, and J. Wei, "Footswitch system for measurement of the temporal parameters of gait", Journal of biomechanics 28.3 (1995): pp. 347–51.

[11] W. Lei, P. Li, Y. Han, S. Gong, L. Yang, and M. Hou, "EPG recordings reveal differential feeding behaviors", *S. furcifera* in response to plant virus infection and transmission success. Scientific reports, 6, p.30240.

[12] B. Liu, et al., 2012, "Difference in feeding behaviors of two invasive whiteflies on host plants with different suitability: implication for competitive displacement", Journal of Biological Sciences, 8(5), p.697.

[13] M. Milenovic, E. Wosula, Carmelo, R. and J. P. Legg. "Impact of Host Plant Species and Whitefly Species on Feeding Behavior of *Bemisia tabaci*." Frontiers in Plant Science, Vol 10, 2019.

[14] D. Murray, et al., "A data management platform for personalised real-time energy feedback", Proc. 8th Int. Conf. Energy Efficiency Domestic Appl. Lighting (EEDAL), Horw, Switzerland, Aug. 2015, pp. 1–15

[15] R. Pathinarupothi and E. Rangan, "Consensus motifs as adaptive and efficient predictors for acute hypotensive episodes", EMBC 2017: 1688-91.

[16] F. Petitjean and P. Gancarski, "Summarizing a set of time series by averaging: From Steiner sequence to compact multiple alignment", Theoretical Computer Science, 414(1): pp. 76–91, 2012.

[17] J. Serrà, I. Serra, A. Corral and J. Lluís Arcos: Ranking and significance of variable-length similarity-based time series motifs. Expert Syst. Appl. 55: 452-460 (2016)

[18] G. Stormo, "DNA binding sites: representation and discovery", BIOINFORMATICS Vol. 16, no. 1 2000, pp. 16-23

[19] C.C.M. Yeh et. al., "Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View that Includes Motifs, Discords and Shapelets", IEEE ICDM 2016, pp. 1317-1322

[20] C.C.M. Yeh, H. Van Herle, and E. Keogh, "Matrix Profile III: The Matrix Profile Allows Visualization of Salient Subsequences in Massive Time Series", Data Mining (ICDM), 2016 IEEE 16th International Conference, pp. 579-588

[21] Y. Zhu, et al, "Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins", IEEE ICDM 2016

[22] E. Keogh: *sites.google.com/site/consensusmotifs/*

VIII. APPENDIX A: ALGORTHM TO CONVERT DNA TO TIME SERIES

This is the algorithm we used to convert DNA strings to time series, it is slightly adapted from an algorithm appearing in [4]. Echoing [4], we do not claim any biological significance for our results on such data. We are merely exploit the availability of ground truth available in evolutionary biology.

```
$T_1 = 0$,    **for** i = 1 **to** length(DNAstring)
                  **if** DNAstringi = **A**, then $T_i+1 = T_i + 2$
                  **if** DNAstringi = **G**, then $T_i+1 = T_i + 1$
                  **if** DNAstringi = **C**, then $T_i+1 = T_i - 1$
                  **if** DNAstringi = **T**, then $T_i+1 = T_i – 2$
```