Variable-Length Latent Motif Discovery

Master Thesis Exposé

Leonard Clauß
Humboldt University of Berlin

August 9, 2021

Supervisors: Dr. rer. nat. Patrick Schäfer
Prof. Dr.-Ing. Ulf Leser

Contents

1 Introduction

A time series is a sequence of real valued numbers, e.g. recorded from a sensor, ordered in time. As the amount of available time series data increased drastically over the last few years, time series analysis gained a lot of attention in recent research. A typical challenge is motif discovery, i.e. the problem of finding frequently occurring patterns of given size within a time series (Figure 1). It is formulated as an unsupervised learning problem. Motif discovery is used as an exploratory task across a multitude of domains, e.g. medicine [1], biology [3], meteorology [15] and robotics [18].

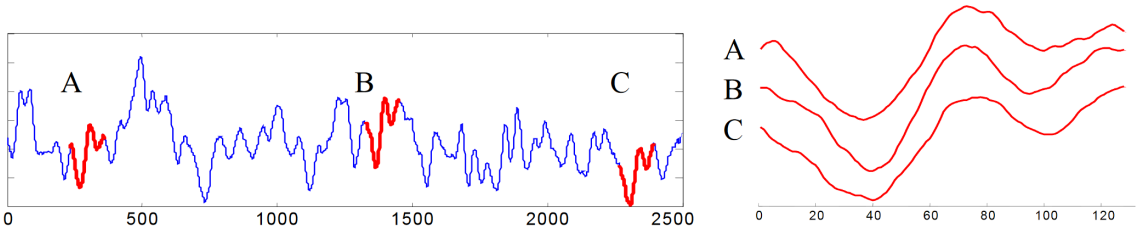

Figure 1: A time series contains a pattern that approximately repeats three times (left). The direct comparison (right) shows that, except for the offset, the marked subsequences are very similar to each other.[1]

In literature, definitions for motifs are found in three different variations, which we will define precisely in the Background section. All these definitions assume that a similarity measure, e.g. Euclidean distance, and a subsequence (motif) size are given.

- *Pair Motifs*, as defined by Mueen et al. [17], are the pairs of subsequences in a time series that are most similar.
- *Set Motifs* are those subsequences in a time series that occur frequently, whereas only non-overlapping subsequences within a given similarity threshold are counted. These form a hypersphere. [11]
- *Latent Motifs* again are real valued sequences that occur frequently in a time series, given a predefined similarity threshold. Two different definitions exist:
  - *Latent Learning Motifs* are similar in definition to set motifs and form a hypersphere. However, the sequence itself, i.e. the center of the motif, must not necessarily be a subsequence of the analyzed time series. [7]
  - *Latent Range Motifs* are sets of subsequences in a time series that are all pairwise within the similarity threshold and thus form a Reuleaux polygon. [17]

Although many motif discovery methods were proposed, most of them address scalability of the pair motif problem. Only few search for set or latent motifs (see [5] for an overview). Farther there exists no exact discovery algorithm for latent motifs, as shown by Moczalla [16]. Thus, in our last study project "Latent Motif Discovery using Maximum Clique algorithms" [5], we proposed a new approach for exact latent motif discovery called CliqueMotif. This algorithm converts the pairwise subsequence distances (distance matrix) of a time series into a so-called distance graph. Given a motif size l, this graph

[1] figure from [19], page 1

contains a node for each subsequence of length l. Two nodes are connected by an unweighted edge if their respective subsequences do not overlap and are within the given similarity threshold. Then, the maximum clique is found which exactly matches the definition of the top latent range motif. Our evaluation showed that the algorithm performs well on problem instances with short time series and tight motif similarity thresholds. On the other hand, it does not scale well on longer time series and looser similarity thresholds, as the problem of finding the maximum clique in a graph is NP-hard.

However, as discussed in [13], the motif length l is a user-defined parameter that is not trivial to set. In this thesis, we plan to extend the CliqueMotif algorithm for variable-length motif discovery, creating the first exact algorithm for this problem in literature. For further usability, we will also create a graphical user interface.

In the next section 2 we will first of all give definitions of the previously mentioned terms. Section 3 then gives an overview of the related work. The last two sections present the objectives (4) and planned methods (5) for this work.

2 Background

In the following we formally define the aforementioned variations of the time series motif discovery problem in literature.

2.1 Basic terms

We begin by defining the basic terms of time series and subsequences.

Definition 2.1 (Time Series)**.** A time series $T = (t_1, t_2, ..., t_n)$ of length n is an ordered sequence of n real-valued numbers.

Definition 2.2 (Subsequence)**.** A subsequence $S_{i,l}$ in a time series $T = (t_1, t_2, ..., t_n)$ with $1 \leq i \leq n$ and $1 \leq l \leq n - i + 1$ is itself a time series of length l and defined as $S_{i,l} = (t_i, t_{i+1}, ..., t_{i+l-1})$, i.e. l consecutive values starting at offset i.

Definition 2.3 (Overlapping subsequences)**.** Two subsequences $S_{i,l}$ and $S_{j,l}$ in a time series T overlap if and only if $i \leq j < i + l$ or $j \leq i < j + l$, i.e. they share at least one index of T.

2.2 Distance measures

Next we introduce the distance measures commonly used for motif discovery.

Definition 2.4 (Euclidean Distance)**.** Given two points $x, y \in \mathbb{R}^n$, their Euclidean distance $d(x, y)$ is defined as:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Definition 2.5 (Z-normalized Euclidean Distance)**.** Given two points $x, y \in \mathbb{R}^n$, their z-normalized Euclidean distance $d_{norm}(x, y)$ is defined as:

$$d_{norm}(x, y) = \sqrt{\sum_{i=1}^{n} (\hat{x}_i - \hat{y}_i)^2} \ ,$$

where $\hat{x}$ and $\hat{y}$ are the z-normalized points of x and y, respectively:

$$\hat{x}_i = \frac{x_i - \mu_x}{\sigma_x} \quad \text{with} \quad \mu_x = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{and} \quad \sigma_x = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_x)^2}$$

The Euclidean distance d is a metric. The z-normalized Euclidean distance is a pseudometric, i.e., distinct points can have a distance of 0. Most motif discovery algorithms use the z-normalized Euclidean distance as the distance measure to gain robustness against horizontal stretching and displacements of subsequences in the time series.

Definition 2.6 (Pearson correlation coefficient)**.** Given two points $x, y \in \mathbb{R}^n$, the Pearson correlation coefficient $\rho(x, y)$ is defined as:

$$\rho(x, y) = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{n\sigma_x\sigma_y} = \frac{1}{n}\sum_{i=1}^{n}\hat{x}_i\hat{y}_i$$

Lemma 2.1. *Pearson's correlation coefficient and the z-normalized Euclidean distance can be converted into each other using the following formulas:*

$$\rho(x, y) = 1 - \frac{d_{norm}(x, y)^2}{2n} \quad \textit{and} \quad d_{norm}(x, y) = \sqrt{2n(1 - \rho(x, y))}$$

Proof. see [4] □

This correlation coefficient is useful for variable-length motif discovery, which we will show in Section 5.

2.3 Motif definitions

For the following definitions assume a length l and a pseudometric $d(\cdot, \cdot)$ on time series of length l as given. We now define the first type of motifs.

Definition 2.7 (Top Pair Motif [17])**.** The top pair motif $P = \{S_{i,l}, S_{j,l}\}$ of a time series T is the set of two non-overlapping subsequences in T that have the minimal distance $d(S_{i,l}, S_{j,l})$ among all pairs of non-overlapping subsequences.

Note that there might be multiple different top pair motifs if they share the same (minimum) distance. Next we define set and latent motifs that describe approximately repeating patterns in a time series.

Definition 2.8 (r-matching time series)**.** Two time series (or subsequences) T_1 and T_2 of length l are r-matching if and only if $d(T_1, T_2) \leq r$.

Definition 2.9 (Top Set Motif [11])**.** Given a radius r. The top set motif T_M of a time series T is the subsequence of length l in T that has the highest number of occurrences. That means, it induces the largest set M of pairwise non-overlapping subsequences of length l in T that are r-matching to T_M.

The idea of latent motifs is to maximize the number of occurrences by relaxing the constraint that T_M has to be a subsequence of T.

Definition 2.10 (Top Latent Range Motif [17])**.** Given a radius r. The top latent range motif R of a time series T is the largest set of subsequences in T with length l that are pairwise non-overlapping and pairwise $2r$-matching.

Definition 2.11 (Top Latent Learning Motif [7])**.** Given a radius r. The top latent learning motif T_L of a time series T is a time series of length l, not necessarily a subsequence of T, that has the highest number of occurrences in T. That means, it induces the largest set L of pairwise non-overlapping subsequences of length l in T that are r-matching to T_L.

Geometrically, the top set motif and top latent learning motif each form a hypersphere with radius r around T_M and T_L, respectively. The top latent range motif forms a Reuleaux polygon.

Again, there might be multiple different top set, top latent range and top latent learning motifs for a single time series. Finally, the pair, set, latent range and latent learning motif discovery problems are redefined as finding their respective top motif.

3 Related Work

3.1 Matrix Profile

Given a time series T and a subsequence length l, the distance profile of a subsequence of length l is a vector that contains its z-normalized Euclidean distance to each other non-overlapping subsequence in T. The matrix profile of the time series, as presented in [23], is a single vector that for each subsequence in T contains the minimal z-normalized Euclidean distance to any other non-overlapping subsequence in T (the 1-NN distance). An entry in the matrix profile thus corresponds to the global minimum of the respective distance profile, i.e. the distance to the nearest neighbor subsequence. The overall minimum in the matrix profile occurs twice, as the z-normalized Euclidean distance is symmetric. The two corresponding subsequences form the top pair motif. Farther, the distance matrix given T is a matrix that contains the distance between each pair of non-overlapping subsequences in T. It therefore consists exactly of all distance profiles.

The whole matrix profile can be computed very fast using the SCAMP algorithm [25], which is an optimized version of STOMP [24]. It only needs $O(n^2)$ time as opposed to $O(n^2 l)$ for the naive approach, where n is the length of the time series, and thus is independent of the subsequence length l. STOMP exploits the fast Fourier transform to compute the sliding dot product of the first subsequence (offset 0), which contains the dot product with every other query subsequence. Given the sliding dot product of a subsequence at offset i, the sliding dot product for the next subsequence at offset $i+1$ is calculated in linear time, as a single dot product is computed in constant time. Each distance profile can easily be calculated from the respective sliding dot product, also in linear time. For mathematical details, we refer to the original work [24]. The final matrix profile is formed by finding the minimum of each distance profile. CliqueMotif uses the distance matrix, i.e. all distance profiles, an intermediate result of SCAMP (see Section 3.4).

3.2 Variable-length Motif Discovery

Linardi et al. proposed an interesting algorithm, named VALMOD, for exact variable-length pair motif discovery [13]. VALMOD uses a simple variable-length motif definition:

Definition 3.1 (Variable-Length Top Pair Motifs [13])**.** The variable-length top pair motifs of a time series T are the set of top pair motifs of T for each length in a given range $[l_{min}, ..., l_{max}]$.

The idea is that distances between subsequences do not change much when the subsequence length gets increased slightly. Thus, the start index of the nearest neighbor of a

subsequence probably does not change as well with increasing motif length. This property is exploited by first computing the matrix profile for the minimal length l_{min}. For increased lengths, the algorithm does not compute the whole matrix profile. Instead, it only finds the global minimum, i.e. the top pair motif. This is done by providing a lower bound of the z-normalized Euclidean distance for increased lengths. Many calculations of exact distances then get pruned if the lower bound distance is already higher than the best motif so far. In the best case, only $O(np)$ exact distances need to be computed for each length, where p is a parameter. The higher p is set, the more likely the best case is. For $l_{max} - l_{min} \in O(n)$, this results in an overall best-case complexity of $O(n^2p)$. The worst-case complexity, however, is $O(n^3 logp)$. For the derivation of the runtime complexity, see the original work [13].

Another method that is based on the matrix profile was presented in [14]. The SKIMP algorithm calculates the matrix profile for all motif lengths. This set of matrix profiles is called the pan matrix profile (PMP) and is calculated by simply running a state-of-the-art fixed-length algorithm, like SCAMP, for each motif length. However, SKIMP is designed as an any-time algorithm and thus used as an approximate method. The order of motif lengths is chosen by iterating through a balanced binary search tree with breadth-first search. At any time, matrix profiles of motif lengths that were not calculated yet can thus get interpolated. In their experiments, the PMP was approximated with less than 10% error after just 3% of all needed calculations for the exact result.

A recent method that is used for variable-length motif discovery is GrammarViz [22]. The approximate algorithm first discretizes a time series and its subsequences into words using SAX [12]. It then uses a compression algorithm to find rules of a context-free grammar for these words. The most used rules represent frequently occurring discretized subsequences and therefore motifs. As a rule can represent subsequences of any length, the algorithm finds variable-length motifs but ignores similarity radius as input parameter.

The HIME algorithm [6] also uses SAX representations to find similar subsequences. It is therefore also an approximate algorithm. However, it does not compute SAX words for a small subsequence length and then compares their concatenation to find longer similar subsequences. Instead, a new SAX word is calculated when the length is increased. Thus, longer motifs can be found, because there are not multiple words representing a single subsequence that have to match. Based on this, HIME tries to find long motifs by expanding shorter ones that were already found. If a longer motif is found, whose occurrences overlap with those of a shorter motif, the latter is removed.

Note that there is no exact algorithm in literature that discovers variable-length latent motifs. The only non-trivial exact algorithm for variable-length pair motif discovery is VALMOD, as it is based on the matrix profile.

3.3 Exact maximum clique algorithms

The problem of finding the maximum clique in an arbitrary graph is NP-hard. This results from the fact that the clique decision problem, i.e. deciding whether a given graph G contains a clique of size k, is already NP-complete (it is one of Karp's 21 original NP-complete problems) [9]. However, efficient exact algorithms exist for sparse graphs that have best-case polynomial runtime. State-of-the-art methods for this problem are BBMCSP [21], PMC [20] and LMC [8]. CliqueMotif uses LMC, as it showed the most promising results and the source code is available (see Section 5).

LMC preprocesses the graph by searching a large initial clique and removing all nodes that can not be a part of a larger clique. Afterwards it employs a branch-and-bound strategy. A simple upper bound for the maximum clique is the number of colors in any

greedy graph coloring. For improvement of this coloring-based upper bound, the algorithm uses incremental MaxSAT reasoning. Therefore, the graph gets colored and implicitly converted into a partial MaxSAT instance which contains hard clauses that have to be satisfied and soft clauses that should be satisfied as far as possible. Then, if conflicting soft clauses are detected, a tighter upper bound can be derived.

3.4 CliqueMotif (Study Project)

CliqueMotif was presented in our previous study project [5]. The algorithm finds the top latent range motif and has two defined parameters: motif length l (also called window) and radius r. Its processing steps are shown in Figure 2, which is only for visualization purposes. The implementation differs slightly, as noted below.

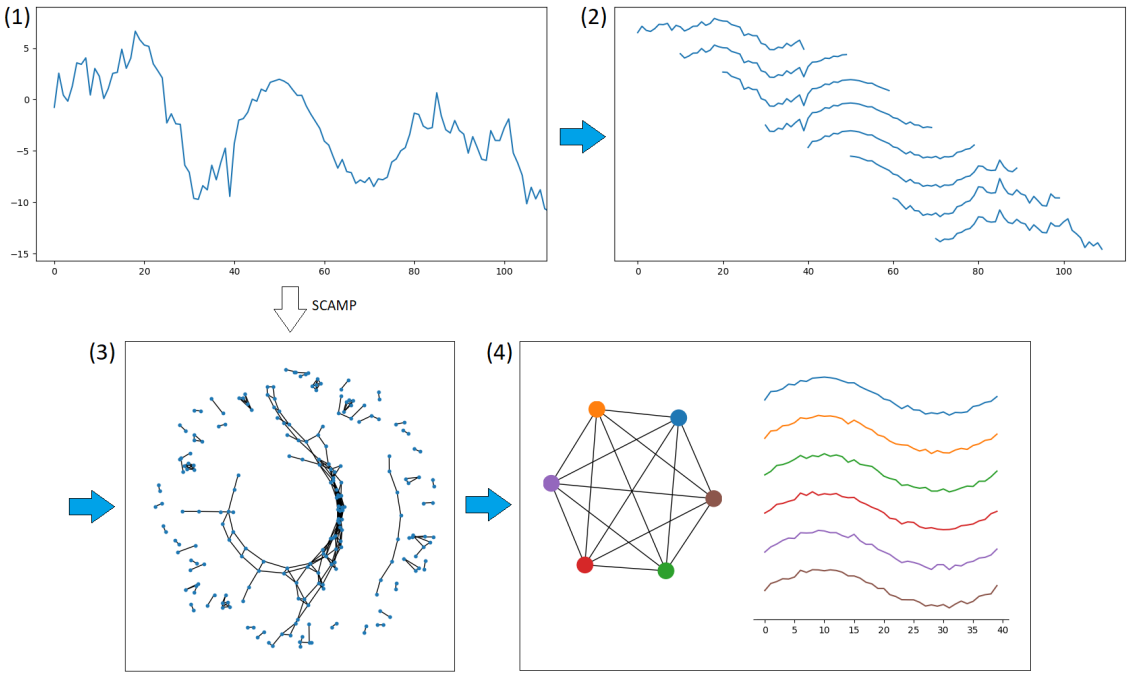

Figure 2: Processing steps of the CliqueMotif algorithm: extracting z-normalized subsequences using a sliding window (1 to 2), calculating the pairwise Euclidean distances and creating the distance graph (2 to 3), then finding the maximum clique (3 to 4)

First, all subsequences are extracted using a sliding window of given length l and z-normalized. Then, the Euclidean distance is calculated for each pair of subsequences. This corresponds to the z-normalized Euclidean distance as the subsequences were normalized beforehand. Note that the z-normalized subsequences are not calculated explicitly in the actual implementation. This is done using SCAMP, which directly computes the correlation between each pair of subsequences from the time series. Afterwards the so-called distance graph is constructed by creating a node for each subsequence. Two nodes get connected by an unweighted edge if and only if the two corresponding subsequences are non-overlapping and have a z-normalized Euclidean distance below $2r$ where r is the given motif radius. Finally, the maximum clique in that graph is found using LMC. The

corresponding subsequences form the top latent range motif as it exactly matches the definition.

Our evaluation showed that CliqueMotif does not scale well with increasing time series length and motif radius due to the NP-hard problem of finding the maximum clique in the distance graph. For short time series (up to length 8000) or very low motif radii (corresponding to a minimal correlation of at least 0.95 between two subsequences), CliqueMotif still has a competitive runtime of few seconds. The theoretical complexity of the latent range motif discovery problem, particularly whether it is NP-hard, is not yet known. Hence, it is unclear whether CliqueMotif is the most efficient algorithm for the problem.

4 Objectives

The goal of this thesis is to improve the CliqueMotif algorithm [5] for practical usage by extending it for variable-length motif discovery. Therefore, we identified the following seven work packages (WP). Afterwards, Section 5 will give a more detailed overview of how we plan to address these issues.

WP1 Define variable-length latent range motifs. A naive definition in analogy to Definition 3.1 would be to find the top latent range motif for each motif length given a range of lengths and a fixed motif radius. However, this favors short motifs, as shown in Section 5.1. We suggest to use a length-invariant correlation threshold instead of a fixed radius, which effectively increases the radius with increasing length.

WP2 Compare an efficient implementation for the creation of all distance graphs to the naive approach. The latter is to run SCAMP once for each motif length. The former might utilize a faster implementation using VALMOD, as outlined in Section 5.2.

WP3 Quickly find the maximum clique for incremental motif lengths l and $l + 1$, i.e. starting with short lengths l. LMC prunes the search space by using a heuristic to find an initial large clique and then using its size as a lower bound. As we expect that the size of the maximum clique will not change much when increasing the motif length slightly, we could improve this heuristic by searching for the largest clique in the graph for length $l + 1$ near the maximum clique for the shorter motif length l (see Section 5.3). Afterwards, we will use the remaining part of LMC unchanged to find the exact maximum clique.

WP4 Find and remove similar motif sets. A problem, that also occurs in VALMOD, is that there are many overlapping top motif sets between different lengths that should be filtered. Therefore, we plan to define a similarity function between two motif sets. Then, the top motifs get clustered and only one representative is output for each cluster (see Section 5.4).

WP5 Evaluate the algorithm on real data and compare it to other state-of-the-art variable-length pair and set motif discovery algorithms: GrammarViz [22], HIME [6] and VALMOD [13]. As the proposed method is an exact algorithm, we will mainly focus on runtime and memory consumption. However, we will also compare the motifs found on some datasets in order to discuss the benefit of latent motifs. We plan to use the data from a current anomaly detection competition [10], which consists of 250 time series with periodic data.

WP6 Optionally: create a graphical user interface for intuitive usage of the algorithm and visualization of the motif discovery results. In order for it to be easily accessible, this will be a stand-alone application.

WP7 Additionally, we believe that the latent range motif discovery problem is NP-complete. Thus, we will try to proof that its corresponding decision problem is NP-complete, by reducing *3-SAT* to it. This would show that no algorithm can exist that has a lower complexity than CliqueMotif for the latent range motif discovery problem, unless $P = NP$.

5 Methods

5.1 Motif Definition and Ranking (WP1)

When defining variable-length latent range motifs, we plan to use a definition similar to VALMOD's for variable-length pair motifs (Def. 3.1):

Definition 5.1 (Variable-Length Top Latent Range Motifs)**.** The top variable-length latent range motifs of a time series T are the set of top latent range motifs of T for each length l in a given range $l \in [l_{min}, ..., l_{max}]$ using a motif radius $r(l) : \mathbb{N} \rightarrow \mathbb{R}^+$ as a function of l.

For the motif radius function $r(l)$ there are two intuitive options: (1) a fixed (constant) radius or (2) a function that grows with increasing length. The former has the drawback that it punishes long motif lengths. In [13] the authors divided the z-normalized Euclidean distance by the square root of the subsequence length. Using this measure, two subsequences still approximately have the same distance when scaled to different lengths. Lemma 2.1 shows that this measure is equal to the correlation, besides some constant transformation. Therefore, it would be natural to choose a fixed correlation and set $r(l)$ accordingly. This effectively increases r with increasing length, i.e. equal to option (2).

To further explore the difference between both options of using a fixed radius and a fixed correlation, we ran the following experiment. Using CliqueMotif, we found the top latent range motif of the 'nyc_taxi' time series [2] for each length. The dataset contains the taxi passenger count in New York City every 30 minutes. First, we set a fixed correlation (0.95). Then, we set the radius to a constant value of 1.095. These two values result in the same motif radius at length 48, which is the period length of the time series (one day).

The sizes of the top motif for each length are shown in Figure 3. At length 48, both methods obviously find the same top motif. It is also clear that a fixed radius favors short motifs. Interestingly, when using a fixed correlation there are large drops in motif size at lengths $48, 96, 336$ (1 day, 2 days, 1 week), which are intuitively appropriate motif lengths for this dataset.

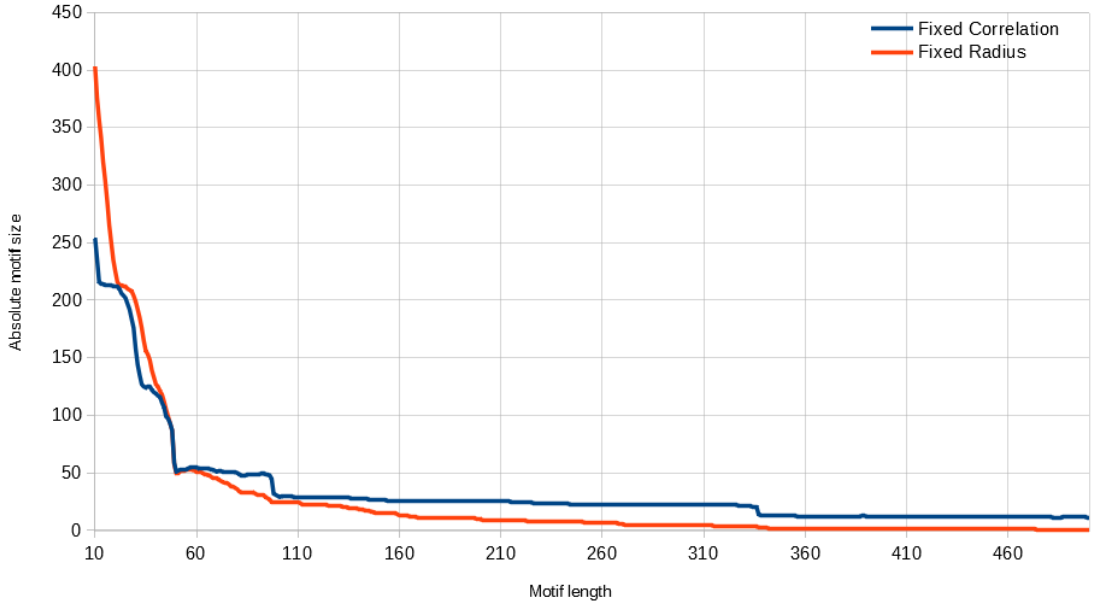

Figure 3: Top motif size (number of occurrences) per length in the 'nyc_taxi' dataset [2] using a fixed correlation threshold of 0.95 (blue) and a fixed motif radius of 1.095 (red)

We also compared the so-called relative motif sizes, which we define as the number of occurrences of a latent range motif R divided by the theoretical maximum number of occurrences for the specific motif length l in time series T:

$$rel\_size(R) = \frac{|R|}{\lfloor |T|/l \rfloor}$$

Figure 4 shows that using relative motif size, these drops in frequency are even more visible. Thus, we think that a fixed correlation is the better choice for defining variable-length latent range motifs.

5.2 Distance Graph Creation (WP2)

For the creation of the distance graphs for different motif lengths, we plan to adapt VALMOD for efficient distance computations. The algorithm finds the top pair motif for each length. As described in Section 3.2, the complete matrix profile is only calculated for the smallest motif length. For longer motif lengths, a lower bound for the z-normalized Euclidean distance between two subsequences is computed. It is used to prune exact distance calculations if the lower bound is already higher than the distance of the best pair motif so far. We can use the algorithm by setting this pruning threshold to $2r$, where r is the motif radius. Thus, only those distances need to get calculated that are possibly within $2r$, as these result in an edge in the distance graph. Note that instead of a distance threshold we will use the correlation threshold, because it is independent of the motif length. This method creates the distance graphs successively. After a graph is created and its maximum clique has been found, the graph is discarded. Thus, only one graph needs to be kept in memory at any time with at most n nodes and n^2 edges, where n is the time series length.

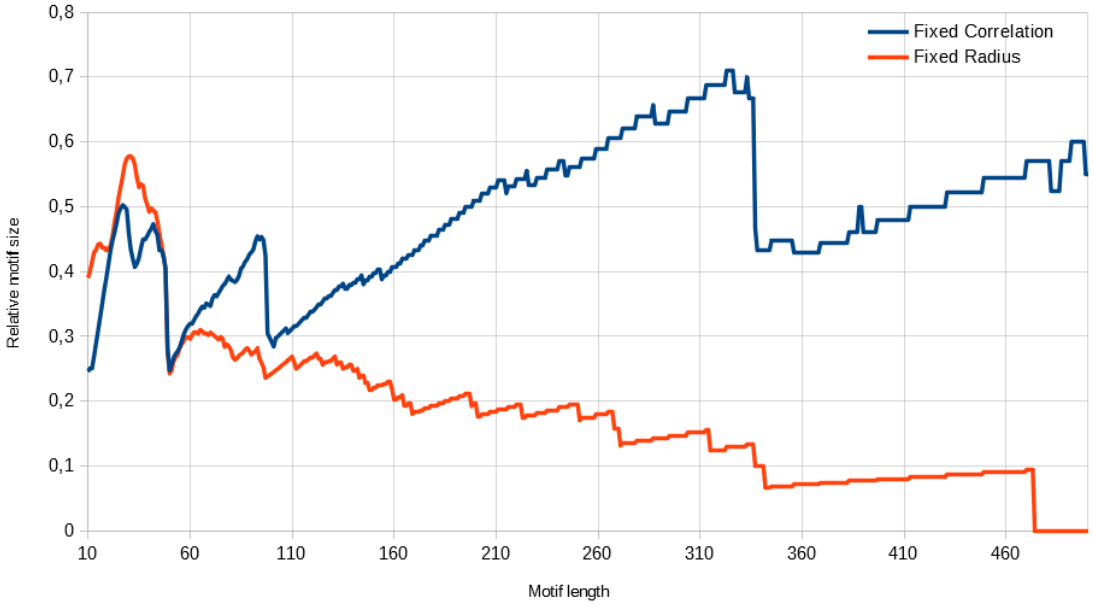

Figure 4: Relative top motif size (number of occurrences divided by maximum number of occurrences) per length in the 'nyc_taxi' dataset [2] using a fixed correlation threshold of 0.95 (blue) and a fixed motif radius of 1.095 (red)

5.3 Maximum Clique Search (WP3)

Finding the maximum clique in all distance graphs still is a hard problem. However, for variable-length motif discovery we can exploit the similarity between graphs of consecutive motif lengths. As mentioned in Section 4, we plan to achieve a speedup by using the following idea of a heuristic to find an initial lower bound and prune the search space. When increasing the motif length by 1, the top motif subsequences will likely still be in the same positions. Given the subsequence indices $I_l = \{i_1, i_2, ..., i_k\}$ of the top motif for length l and the distance graph G_{l+1} for size $l+1$. We then calculate the sub-graph that only contains nodes for subsequences which completely cover the subsequences represented by I_l:

$$V = \{i_1 - 1, i_1, i_2 - 1, i_2, ..., i_k - 1, i_k\}$$

The graph thus has $|V| = 2|I_l|$ nodes, i.e. twice the size of the top motif for size l. Due to this small size, the maximum clique of size s can be found very fast in the sub-graph. As explained above, we expect s to be a tight lower bound for the maximum clique size in G_{l+1}. Using this, we can improve the heuristic in LMC that is used to find an initial lower bound.

5.4 Motif Clustering (WP4)

There are many overlapping motif sets between different lengths, especially if the lengths differ only slightly. A simple idea is to remove covered motifs, which are short motifs that completely overlap with long motifs, similar to HIME [6]. However, as short motifs generally have more occurrences than longer motifs, they are often not completely overlapping. Thus, we plan to cluster similar motifs and then output a single representative for each cluster. Based on the observation of appropriate motif lengths being at a local peak in

the relative motif size graph (see Section 5.1), this representative should be the motif with the highest relative size.

For clustering, a similarity function between two motif sets R_1 and R_2 is needed. Therefore, we first calculate the mean sequence $\overline{r}_1, \overline{r}_2$ for both motifs, which is the mean of all z-normalized subsequences in the respective motif set:

$$\overline{r}_a(i) = \frac{1}{|R_a|} \sum_{s \in R_a} \hat{s}(i)$$

Intuitively, two motifs are similar if their mean sequences are similar. However, the mean sequences for two motifs of different lengths also have different lengths. Dynamic Time Warping (DTW) is not an appropriate similarity measure, as the sequences are not warped and are derived from the same time series. In order to use correlation or Euclidean distance, we thus need to pad the shorter sequence with zeros.

The number of clusters, i.e. the number of unique motifs, is not known beforehand. Simple algorithms like K-Means thus can not be used. Instead, we plan to use hierarchical clustering, as it does not have additional parameters and the number of clusters can be adjusted afterwards.

5.5 Implementation (all WP)

The programming language used for the proposed method will depend on the part of the algorithm. For SCAMP, we will use an existing implementation in C++[2]. We will modify the original implementation of VALMOD, written in C[3]. To find the maximum clique we use and adapt an existing implementation of the LMC algorithm[4]. The communication between these subprocesses will be written in Python. The graphical user interface will also be written in Python using the PyQt framework[5].

Finally, the project code will be uploaded to a public GitHub repository[6] in order for it to be available for public usage.

References

[1] H. Abe and T. Yamaguchi. Implementing an Integrated Time-Series Data Mining Environment - A Case Study of Medical KDD on Chronic Hepatitis. In *1st international conference on complex medical engineering (CME2005)*, 2005.

[2] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017.

[3] I. P. Androulakis, J. Vitolo, and C. Roth. Selecting maximally informative genes to enable temporal expression profiling analysis. *Proc. of Foundations of Systems Biology in Engineering*, page 23, 2005.

[4] Michael R. Berthold and Frank Höppner. On clustering time series using euclidean distance and pearson correlation. *arXiv preprint arXiv:1601.02213*, 2016.

[2] `https://github.com/zpzim/SCAMP`
[3] `https://helios.mi.parisdescartes.fr/~mlinardi/VALMOD.html`
[4] `https://home.mis.u-picardie.fr/~cli/EnglishPage.html`
[5] `https://riverbankcomputing.com/software/pyqt/download`
[6] `https://github.com/leclauss/clique-motif`

[5] Leonard Clauss. Latent Motif Discovery using Maximum Clique algorithms. *Humboldt University of Berlin*, 2021.

[6] Yifeng Gao and Jessica Lin. HIME: discovering variable-length motifs in large-scale time series. *Knowledge and Information Systems*, 61(1):513–542, 2019.

[7] Josif Grabocka, Nicolas Schilling, and Lars Schmidt-Thieme. Latent Time-Series Motifs. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 11(1):1–20, 2016.

[8] Hua Jiang, Chu-Min Li, and Felip Manyà. Combining Efficient Preprocessing and Incremental MaxSAT Reasoning for MaxClique in Large Graphs. In *Proceedings of the twenty-second European conference on artificial intelligence*, pages 939–947, 2016.

[9] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[10] E. Keogh, T. Dutta Roy, U. Naik, and A. Agrawal. Multi-dataset Time-Series Anomaly Detection Competition. *SIGKDD*, 2021. `https://compete.hexagon-ml.com/practice/competition/39/`.

[11] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Pranav Patel. Finding Motifs in Time Series. In *Proc. of the 2nd Workshop on Temporal Data Mining*, pages 53–68, 2002.

[12] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing SAX: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2):107–144, 2007.

[13] Michele Linardi, Yan Zhu, Themis Palpanas, and Eamonn Keogh. Matrix profile X: VALMOD-scalable discovery of variable-length motifs in data series. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1053–1066, 2018.

[14] Frank Madrid, Shima Imani, Ryan Mercer, Zachary Zimmerman, Nader Shakibay, and Eamonn Keogh. Matrix Profile XX: Finding and Visualizing Time Series Motifs of All Lengths using the Matrix Profile. In *2019 IEEE International Conference on Big Knowledge (ICBK)*, pages 175–182. IEEE, 2019.

[15] Amy McGovern, Derek H. Rosendahl, Adrianna Kruger, Meredith G. Beaton, Rodger A. Brown, and Kelvin K. Droegemeier. Anticipating the formation of tornadoes through data mining. 2007.

[16] Rafael Moczalla. A Synthetic Motif Generator. Master's thesis, Humboldt University of Berlin, 2020.

[17] Abdullah Mueen, Eamonn Keogh, Qiang Zhu, Sydney Cash, and Brandon Westover. Exact Discovery of Time Series Motifs. In *Proceedings of the 2009 SIAM international conference on data mining*, pages 473–484. SIAM, 2009.

[18] Tim Oates, Matthew D. Schmill, and Paul R. Cohen. A Method for Clustering the Experiences of a Mobile Robot that Accords with Human Judgments. In *AAAI/IAAI*, pages 846–851, 2000.

[19] Pranav Patel, Eamonn Keogh, Jessica Lin, and Stefano Lonardi. Mining Motifs in Massive Time Series Databases. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pages 370–377. IEEE, 2002.

[20] Ryan A. Rossi, David F. Gleich, Assefaw H. Gebremedhin, and Md. Mostofa Ali Patwary. Parallel Maximum Clique Algorithms with Applications to Network Analysis and Storage. *arXiv preprint arXiv:1302.6256*, 2013.

[21] Pablo San Segundo, Alvaro Lopez, and Panos M Pardalos. A new exact maximum clique algorithm for large and massive sparse graphs. *Computers & Operations Research*, 66:81–94, 2016.

[22] Pavel Senin, Jessica Lin, Xing Wang, Tim Oates, Sunil Gandhi, Arnold P. Boedihardjo, Crystal Chen, and Susan Frankenstein. GrammarViz 3.0: Interactive Discovery of Variable-Length Time Series Patterns. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(1):1–28, 2018.

[23] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets. In *2016 IEEE 16th international conference on data mining (ICDM)*, pages 1317–1322. Ieee, 2016.

[24] Yan Zhu, Zachary Zimmerman, Nader Shakibay Senobari, Chin-Chia Michael Yeh, Gareth Funning, Abdullah Mueen, Philip Brisk, and Eamonn Keogh. 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 (ICDM)*, pages 739–748. IEEE, 2016.

[25] Zachary Zimmerman, Kaveh Kamgar, Nader Shakibay Senobari, Brian Crites, Gareth Funning, Philip Brisk, and Eamonn Keogh. Matrix profile XIV: scaling time series motif discovery with GPUs to break a quintillion pairwise comparisons a day and beyond. In *Proceedings of the ACM Symposium on Cloud Computing*, pages 74–86, 2019.