Probabilistic Discovery of Time Series Motifs

Bill Chiu Eamonn Keogh Stefano Lonardi

Computer Science & Engineering Department
University of California - Riverside
Riverside, CA 92521

{bill, eamonn, stelo }@cs.ucr.edu

ABSTRACT

Several important time series data mining problems reduce to the core task of finding approximately repeated subsequences in a longer time series. In an earlier work, we formalized the idea of approximately repeated subsequences by introducing the notion of time series motifs. Two limitations of this work were the poor scalability of the motif discovery algorithm, and the inability to discover motifs in the presence of noise.

Here we address these limitations by introducing a novel algorithm inspired by recent advances in the problem of pattern discovery in biosequences. Our algorithm is probabilistic in nature, but as we show empirically and theoretically, it can find time series motifs with very high probability even in the presence of noise or "don't care" symbols. Not only is the algorithm fast, but it is an anytime algorithm, producing likely candidate motifs almost immediately, and gradually improving the quality of results over time.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Applications - Data Mining

Keywords

Time Series, Data Mining, Motifs, Randomized Algorithms.

1. INTRODUCTION

Several important time series data mining problems reduce to the core task of finding approximately repeated subsequences in a longer time series. In an earlier work, we formalized the idea of approximately repeated subsequences by introducing the notion of time series motifs [26]. We will define motifs more formally later in this work. In the meantime a simple graphic example will serve to develop the reader's intuition. Figure 1 illustrates an example of a motif discovered in a complex dataset.

Examples of algorithms that utilize *motifs* (typically under different names and with variants of definitions) include the following:

- Mining association rules in time series requires the discovery of motifs. These are referred to as *primitive shapes* in [7] and *frequent patterns* in [18].
- Several time series classification algorithms work by constructing typical *prototypes* of each class [22, 15]. These prototypes may be considered motifs.
- Many time series anomaly/interestingness detection algorithms essentially consist of modeling normal behavior with a set of *typical shapes* (which we see as motifs), and detecting future patterns that are dissimilar to all typical shapes [8].
- In robotics, Oates et al. [27], have introduced a method to allow an autonomous agent to generalize from a set of *qualitatively different experiences* gleaned from sensors. We see these "experiences" as motifs.
- Much of the work on finding *approximate periodic patterns* in time series can viewed as an attempt to discover motifs that occur at constrained intervals [14]. For example, the astute reader may have noticed that the motif in Figure 1 appears at approximately equal intervals, suggesting an unexpected regularity.

In addition to the application domains mentioned above, motif discovery can be very useful in its own right as an exploratory tool to allow hypothesis generation [11].

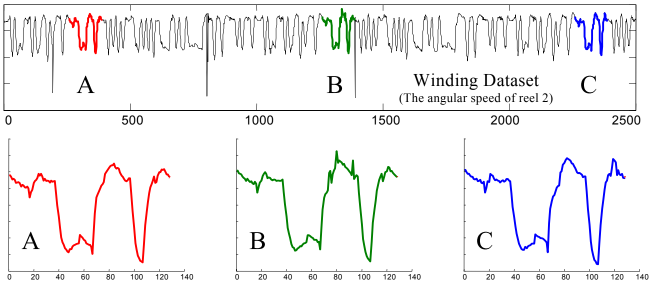

Figure 1: *Above*) An example of a motif that occurs three times in a complex and noisy industrial dataset. *Below*) a zoom-in reveals just how similar the three occurrences are to each other

There exists a vast body of work on efficiently locating *known* patterns in time series [1, 6, 12, 23, 35, 36, 37]. Here, however, we must be able to *discover* motifs without any prior knowledge about the regularities of the data under study.

The obvious, nested-loop, brute force approach to motif discovery would require a number of comparisons quadratic in the length of the database. Optimizations based on the triangular inequality can mitigate the time complexity by a large constant factor [26], but

SIGKDD '03, August 24-27, 2003, Washington, DC, USA

this approach is still untenable for large and massive datasets. All the works listed above introduce methods to discover some form of motifs, but the definitions are application specific, scalability is not addressed, and more importantly they completely disregard the problem of noise.

The importance of noise when attempting to discover motifs cannot be overstated. Consider the two sequences shown in Figure 2. While they are extremely similar, one of them has a downward spike at time 38.

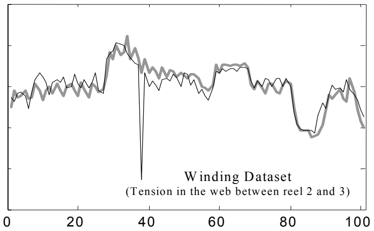

Figure 2: Two subsequences from an industrial dataset. Although they appear to very similar, the noisy downward spike at time period 38 in one of the sequences will make it difficult for algorithms to discover this potential motif

One could assume that one outlier in a sequence of length 100 data points would not make much difference. However even small amounts of noise can dominate distance measures, including the most commonly used data mining distance measures, such as the Euclidean distance [6, 7, 8, 21, 36]. Figure 3 shows that the spike can cause one of our candidate motifs to appear to be much more similar to an artificial sequence which just happens to have spike in the same place.

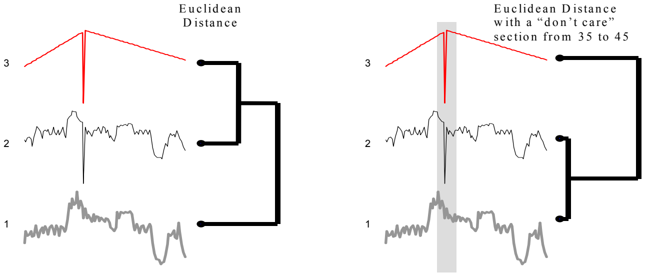

Figure 3: *Left*) The two sequences from Figure 2 clustered together with a synthetic sequence, using Euclidean distance. The synthetic sequence does not particularly resemble the two real sequences, but happens to have noise in the same place as sequence 2. This dendrogram demonstrates that a single piece of noise can dominate a distance function. *Right*) If we allow the distance function to have “*don't care*” sections (denoted by the gray bar), more intuitive results can be obtained

There is still hope for us if we wish to mine noisy datasets. Figure 3 also shows that allowing small *don't care* subsections (that is, sections which are ignored by the distance function), allows much more intuitive results to be obtained. We note that the utility of allowing *don't care* sections in time series has been documented before [1, 22], and it is a cornerstone of text and Biosequences data mining [3, 24, 25, 28, 30, 34].

The previous example illustrates the dangers of mining in the presence of noise. Indeed, this single spike might be best taken care of with a simple smoothing algorithm. More generally, however, we may have a potential motif if we are willing to overlook the fact that a small valley in one sequence is mirrored by a small peak in another, otherwise similar, sequence. Robustness to such situations is non-trivial [1].

Our contributions in this paper are twofold. We generalize the definition of time series motifs to allow for *don't-care* subsections, and we introduce a novel time- and space-efficient algorithm to discover motifs. Our method is based on a recent algorithm for pattern discovery in DNA sequences [34]. The intuition behind the algorithm is to project the data objects (in our case, time series), onto lower dimensional subspaces, based on a randomly chosen subset of the objects features. The lower dimensional space can be quickly post-processed to discover likely candidates for motifs, while the candidates can be quickly checked against the original data.

The rest of this paper is organized as follows. In Section 2 we formally define the time series motif problem. In Section 3 we briefly review related work in time series data mining, and in bioinformatics. Section 4 sees an extensive review of the work of Buhler and Tompa [34], upon which our algorithm is based. The section that gives a detailed explanation of our algorithm is omitted from this “poster” version of the paper. We urge the reader to consult the full version of the paper which is available from the second authors web page. In Section 5 we provide the results of a comprehensive experimental evaluation.

2. DEFINITIONS AND NOTATION

We made some initial progress in defining time series motifs in a previous paper [26], here we generalize the definition to allow for matching under the presence of noise, and to eliminate a special, degenerate case of a motif.

For concreteness, we begin with a definition of our data type of interest, time series:

Definition 1. *Time Series*: A time series $T = t_1,\ldots,t_m$ is an ordered set of m real-valued variables.

Time series can be very long, sometimes containing trillions of observations [12, 32]. We are typically not interested in any of the global properties of a time series; rather, we are interested in subsections of the time series, which are called subsequences.

Definition 2. *Subsequence*: Given a time series T of length m, a subsequence C of T is a sampling of length $n \leq m$ of contiguous position from T, that is, $C = t_p,\ldots,t_{p+n-1}$ for $1 \leq p \leq m - n + 1$.

Since all subsequences may be a potential motif, any motif discovery algorithm will eventually have to extract all of them, this can be achieved by use of a sliding window [7, 23, 36].

Definition 3. *Sliding Window*: Given a time series T of length m, and a user-defined subsequence length of n, a matrix S of all possible subsequences can be built by sliding a window of size n across T and placing subsequence C_p in the p^{th} row of S. The size of matrix S is $(m - n + 1)$ by n.

A task commonly associated with subsequences is to determine if a given subsequence is similar to other subsequences under some distance measure $D(C,M)$ [21]. This idea is formalized in the definition of a match.

Definition 4. *Match*: Given a positive real number R (called *range*) and a time series T containing a subsequence C beginning at position p and a subsequence M beginning at q, if $D(C, M) \leq R$, then M is called a *matching* subsequence of C.

The first three definitions are summarized in Figure 4, illustrating a time series of length 1,000, and two subsequences of length 128.

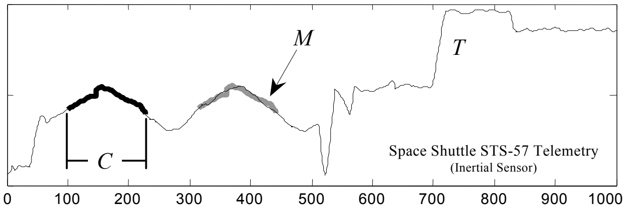

Figure 4: A visual intuition of a time series T (light line), a subsequence C (bold line) and a subsequence M that is a match to C (C is overlaid as a bold gray line)

Whereas the definition of a match is rather obvious and intuitive, we also need for the definition of a *trivial match*. One can observe that the best matches to a subsequence (apart from itself) tend to be located one or two points to the left or the right of the subsequence in question. Figure 5 shows the situation.

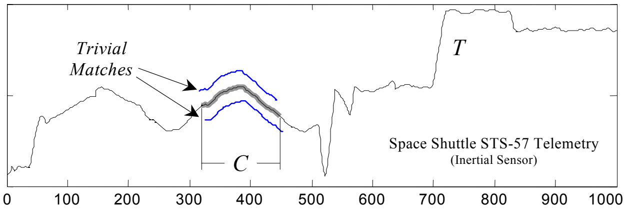

Figure 5: For almost any subsequence C in a time series, the best matches are the trivial subsequences immediately to the left and right of C

Intuitively, any definition of motif should exclude the possibility of over-counting these trivial matches, which we define more concretely below.

> **Definition 5**. *Trivial Match*: Given a time series T, containing a subsequence C beginning at position p and a matching subsequence M beginning at q, we say that M is a *trivial match* to C if either $p = q$ or there does not exist a subsequence M' beginning at q' such that $D(C, M') > R$, and either $q < q' < p$ or $p < q' < q$.

Each time series is normalized to have mean zero and a standard deviation of one before calling the distance function, because it is well understood that it is meaningless to compare time series with different offsets and amplitudes [6, 21, 35, 36].

We can now define the problem of enumerating the K most significant motifs in a time series.

> **Definition 6**. *K-Motif*(n,R): Given a time series T, a subsequence length n and a range R, the most significant motif in T (hereafter called the *1-Motif*(n,R)) is the subsequence C_1 that has highest count of non-trivial matches (ties are broken by choosing the motif whose matches have the lower variance). The K^{th} most significant motif in T (hereafter called the *K-Motif*(n,R)) is the subsequence C_K that has the highest count of non-trivial matches, and satisfies $D(C_K, C_i) > 2R$, for all $1 \leq i < K$.

Note that this definition forces the set of subsequences in each motif to be mutually exclusive. This is important because otherwise two motifs might share the majority of their elements, and thus be essentially the same. To gain more intuition for these definitions, Figure 1 shows the *1-Motif*(128,4) discovered in the Winding dataset.

Definition 6 does not allow for don't care subsections [1], but it can easily extended.

> **Definition 7**. *K-Motif*(n,R,d): The K^{th} most d-significant motif in T (hereafter called the *K-Motif*(n,R,d)) is the subsequence C_K that has the highest count of non-trivial matches, and satisfies $D(C_K, C_i) > 2R$ where d (possibly non-contiguous) datapoints can be ignored while calculating the distance between C_K, C_i, for all $1 \leq i < K$. In general we have $d < n$, and typically $d << n$.

Deciding which d datapoints to ignore is easy. Since we want to minimize the calculated distance, we can sort the indices i in increasing order of $|C_i - M_i|$, and ignore the first d.

We note that this definition has a close analogue in classic motif discovery in biosequences [28, 34]. In the bioinformatics community, the (w,d)-motif problem is to discover a reoccurring sequence of length w, where each occurrence may differ in d positions. Note that in the discrete case, there is no R parameter, since it is implicit the use of Hamming distance.

There is one final consideration we must address if we wish to have a meaningful definition of motif. The problem is best illustrated with a visual example.

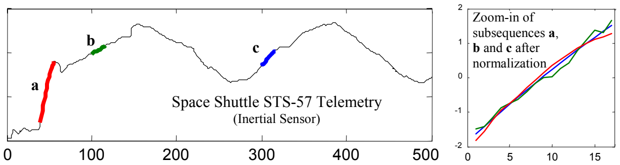

Figure 6: *Left*) Three subsequences of length 16 that can be modeled well by a straight line. *Right*) After normalization, all such subsequences become virtually indistinguishable

Figure 6 shows that the subsequences that can be well-approximated by a straight line with a positive slope will look almost identical after normalization. Since almost all time series can be modeled well by piecewise linear functions if the subsequences are short enough [12, 22], then the most common motifs will likely correspond to an upward trend or a downward trend of arbitrary angles. These "degenerate motifs" are unlikely to be of interest to anyone, and in any case, are trivial to enumerate with a simple algorithm [19]. We will therefore exclude them from further consideration. This can easily be achieved at the feature extraction stage, when using sliding windows to extract the subsequences. As the window is moved across the time series, the subsequences "straightness" can be measured by doing a least squares linear fit, and recording the residual error [21, 23]. Only those subsequences that have a residual error greater that some epsilon are extracted and passed to the motif discovery algorithm. With a careful implementation that reuses partial results from the previous windows, this can be achieved in amortized constant time per subsequence.

3. RELATED WORK

In order to frame our contribution in its proper context we will briefly consider related work.

To date the majority of work in time series data mining has focused indexing time series, the efficient discovery of *known* patterns in time series [1, 6, 12, 21, 22, 23, 31, 35, 36, 37].

The innovative work of Oates *et al.* considers the problem of learning "qualitatively different experiences" (which we see as motifs), but the authors are working with relatively small datasets, and thus did not address scalability issues [27].

Pattern discovery algorithms for biosequences have recently received increased attention from researchers, in particular after the challenge by Pevzner and Sze [28] (see below). We mention, in no particular order and without pretending to be exhaustive, TEIRESIAS [30], GIBBSSAMPLER [24], MEME [3], WINNOWER [28], VERBUMCULUS [2], PROJECTION [34], among others.

Of particular interest is the PROJECTION algorithm by Buhler and Tompa [34]. They applied random projection in their paper to find motif in nucleotide sequences. The most important contribution was in formulating the number of random trials to run in order to achieve some specific bucket richness. Since this work is the cornerstone of our contribution, we will discuss the contribution of Buhler and Tompa in more detail in the next section.

4. MOTIF DISCOVERY AND THE RANDOM PROJECTION ALGORITHM

The projection algorithm by Buhler and Tompa was designed to attack the planted *(w,d)*-motif problem, which was proposed by Pevzner and Sze [28].

> **Planted *(w,d)*-motif problem.** You are given t strings of length n, initially generated at random (i.e., each symbol generated i.i.d. with the equal probability). Each string is planted with exactly one approximate occurrence of an unknown motif y of length w, that is, an occurrence with exactly d substitutions. Find the unknown motif y.

The initial challenge by Pevzner and Sze was to solve the *(15,4)*-motif problem on $t=20$ sequences of $n=600$ symbols over the DNA alphabet (i.e., $|\Sigma|=4$). This problem turned out to be extremely hard to solve for commonly used pattern discovery algorithms. We need a few definitions to explain why.

> **Definition 8.** Given two strings y_1 and y_2, $|y_1|=|y_2|$, the Hamming distance $H(y_1,y_2)$ is given by the number of mismatches between y_1 and y_2.

> **Definition 9.** Given a string y, all strings at Hamming distance at most d from y are in its *d-neighborhood.*

Observe that if you consider two approximate occurrences of the unknown motif y, the Hamming distance between them may be as large as $2d$. In fact, it very likely that we will never observe y in the t sequences. Figure 7 illustrates the problem from a geometric perspective.

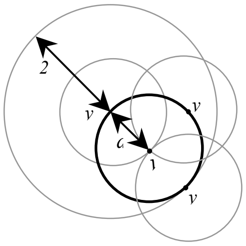

Figure 7: the string y is the (unknown) motif, d is the number of allowed mismatches, and y_1,y_2,y_3 belongs to the d-neighborhood of y. The problem is to find y from y_1,y_2,y_3

To make the problem even more difficult, even if we were able to determine exactly all the y_i in the d-neighborhood of y, there is no guarantee to find the unknown model y. Suppose $w=4$, $d=1$ and that we found the strings `{AAAA,TATA,CACA}`. The pair wise Hamming distance is 2 but there is no string at Hamming distance 1 to each of these.

The brute force strategy of building all possible substrings in the $2d$-neighboorshood of all the substrings of the sequence under analysis is doomed to fail. In fact, the size of the size $N(m,d)$ of the d-neighborhood of a string y is

$$N(|y|,d)=\sum_{j=0}^{d}\binom{|y|}{j}(|\Sigma|-1)^{j}\in O\left(|y|^{d}|\Sigma|^{d}\right), \tag{1}$$

and grows exponentially with d.

In order to reduce the huge search space, Buhler and Tompa used random projection to “guess” at least *some* of occurrences of the unknown planted motif.

Buhler and Tompa projection algorithm carries out i iterations, in each of which it chooses k distinct positions uniformly at random out the w possible. The k positions become a “mask” that is superimposed at all positions on the sequences under study. Each substring of size w in the sequence is therefore mapped to a string of size k, by reading the symbols though the mask.

The frequency of the projected strings is collected into a hash table. If k is chosen such that $k<w-d$ then it is likely that some of the occurrences of the planted motif will hash together in the same entry. Entries in the hash table whose count is higher than a specified threshold s are therefore selected, and they become the seed for a refinement process that uses expectation maximization (EM) [25].

Crucial factors in the success of PROJECTION are the choices of the projection size k, the number of iteration i, and the threshold s. The parameter k has to be chosen such that $k<w-d$ and $|\Sigma|^{k}>t(n-w+1)$ in order to sample from the non-varying positions (first constraint) and to filter out the noise (second constraint). The number of iteration i can be estimated from w, t, d, k, and s.

5. EXPERIMENTAL RESULTS

We begin with a simple demonstration of our algorithm.

5.1 A “Sanity Check”: Finding Planted Motifs

As a “sanity check” we attempt to recover two planted motifs, each with two occurrences, from a small dataset. The two planted motifs are shown in Figure 8. Note that they are by no means identical to each other. For example, consider the **AB** motif. During time period 30 to 40, subsequence **A** is mostly flat with a single dramatic upward spike. In contrast, subsequence **B** is characterized by a relatively smooth valley in this region. In addition, all the subsequences are noisy along their entire length.

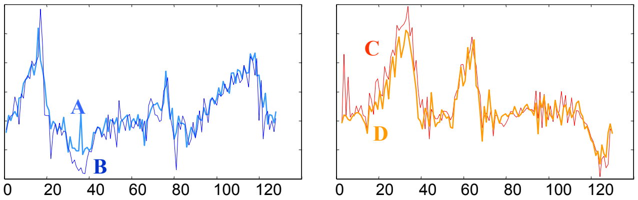

Figure 8: Two motifs which will be planted into a longer dataset as a simple test of our algorithm. *Left*) The **AB** motif, *Right*) the **CD** motif

We embedded the four subsequences into a random walk dataset of length 1128. The dataset was scaled such that the average standard deviation in any subsequence of length 128 was about the standard deviation of our embedded motifs. Figure 9 illustrates the dataset.

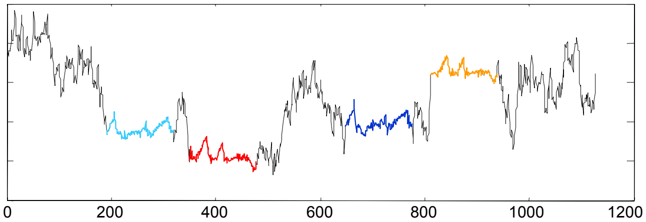

Figure 9: A random walk time series with implanted motifs. The subsequences were randomly imbedded in the following locations {**A**, 191}, {**B**, 649}, {**C**, 351} and {**D**, 812}

We ran our algorithm with n = 128, w = 16, a = 4 for 100 iterations. For ease of visualization we did *not* perform the numerosity reduction step discussed in Section 5.3. Because this is a relatively small dataset, we can visualize the collision matrix as a contour plot as in Figure 10.

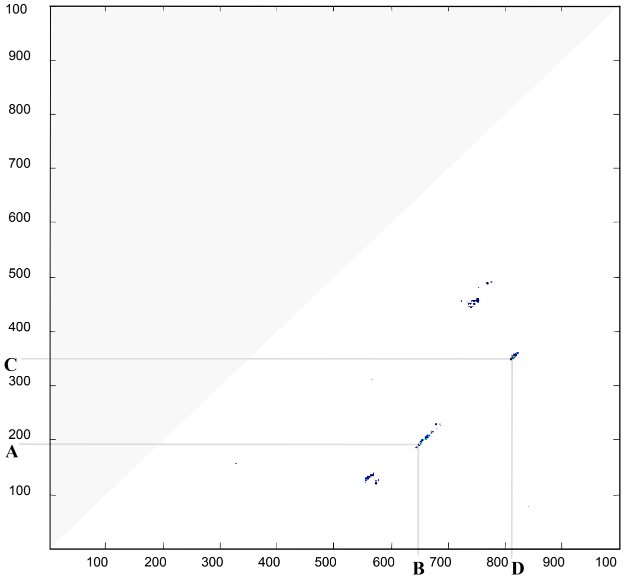

Figure 10: The collision matrix visualized as a contour plot. Only values which are at least 10 times the expected value (cf. 5.4) are shown

These preliminary results are extremely encouraging. The locations of the planted motifs are clearly seen as dark smudges. Note that the location of the planted motifs appears to be slightly "smeared" at a 45 degree angle. This is simply the result of not doing the numerosity reduction step, because if location (i, j) has strong motif, the locations (i +1, j +1) and (i -1, j -1) will have a slightly weaker one, etc.

Finally, we tested the sensitivity of the algorithm to the parameter n,w and a. In practice, one would like to be able to recover the motifs without knowing the *exact* length of the planted motifs. We discovered that we could make n much shorter than 128 and still trivially find (a subsection) of our planted motifs. This is not surprising since it is very likely that a portion of a motif is also a motif. A more satisfying result is the fact that we could set n to be larger that 128 (at least 150), and still easily recover the planted motif. Regarding parameters w and a, we found we could vary them greatly and still easily recover the planted motifs. The only significant difference was a slight change in the efficiency of the algorithm.

5.2 Sensitivity to Noise

The experiment in the previous section suggests that our algorithm is reasonably robust to noise. The planted motifs which were so easily recovered were actually quite noisy, as was the dataset into which they were imbedded. Nevertheless, it is natural to ask how sensitive to noise TIME SERIES PROJECTION is.

To answer the question we performed the following experiment. We took the dataset used in Section 6.1 and kept adding noise to it until the largest value in the collision matrix no longer corresponded to one of the planted motifs. We used Gaussian random noise, which was added to the entire length of the dataset. We began with noise which had a standard deviation that was a tiny fraction the standard deviation of the original data, and kept doubling the noise level until the average value of the planted motifs was no greater than the largest other value. Figure 11 shows a typical amount of noise that can be tolerated by our algorithm. If this noise level is doubled again, the planted motif is not anymore the 1-motif and 2-motif (although even when the noise level shown is quadruped, we still typically find the planted motifs in the first 4 or 5 motifs reported).

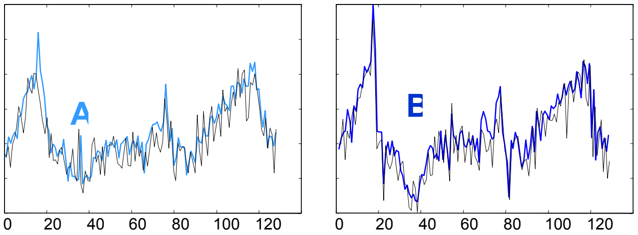

Figure 11: Even when noise is added to the test dataset introduced in Section 6.1, the TIME SERIES PROJECTION algorithm can still discover the planted motifs. Although noise is added to the entire dataset, here we only show the planted **AB** motif with an amount of noise that our algorithm can handle. If the amount of noise is doubled again, our algorithm fails to find this motif as the most promising candidate in the collision matrix

5.3 Efficiency of TIME SERIES PROJECTION

To test the scalability of our algorithm, we began by measuring the time taken for the experiment discussed in Section 6.1. We then repeatedly concatenated an additional 1,000 length of random walk data, and measured the increase in time required. We tested two variants of our algorithm. In the first, we ran the algorithm for 100 iterations. In the second, we stopped after the largest value in the collision matrix was at least ten times larger than expected by chance (as measured by Eq. 9). As a comparison we tested against the obvious brute force algorithm. We highly optimized the brute force algorithm (including removing the square root from the Euclidean distance function, "early abandonment", triangular inequality pruning, etc [21]). In contrast, as TIME SERIES PROJECTION is still in the development stage, we did not optimize it. The results are shown in Figure 12.

The results seems to confirm the theoretical analysis in Section 5.4, brute force is quadratic, TIME SERIES PROJECTION is linear, in the length of the time series. Note that for every experiment, we compared the result of both variants of our algorithm with the results from brute force. In *every* case the top 3 motifs were the same.

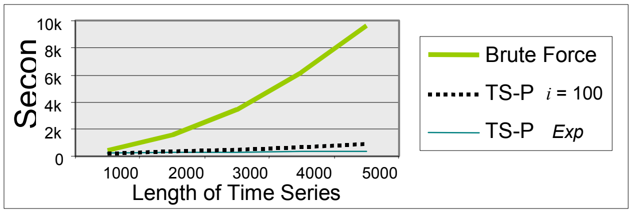

Figure 12: The scalability of various motif discovery algorithms

6. REFERENCES

[1] Agrawal, R., Lin, K. I., Sawhney, H. S. & Shim, K. (1995). Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *proceedings of the 21st Int'l Conference on Very Large Databases*. Zurich, Switzerland, Sept. pp 490-50.

[2] Apostolico, A., Bock, M. E. & Lonardi, S. (2002). Monotony of surprise and large-scale quest for unusual words. In *proceedings of the 6th Int'l Conference on Research in Computational Molecular Biology*. Washington, DC, April 18-21. pp 22-31.

[3] Bailey, T & Elkan, C. (1995). Unsupervised learning of multiple motifs in biopolymers using expectation maximization, Machine Learning, 21 (1/2), pp 51-80.

[4] Buhler, J. (2001). Efficient large-scale sequence comparison by locality-sensitive hashing, Bioinformatics 17: pp 419-428.

[5] Caraca-Valente., J.P. & Lopez-Chavarrias. I. (2000). Discovering similar patterns in time series. *In Proceedings of the Association for Computing Machinery 6th International Conference on Knowledge Discovery and Data Mining*, pp 497-505.

[6] Chan, K. & Fu, A. W. (1999). Efficient time series matching by wavelets. In *proceedings of the 15th IEEE Int'l Conference on Data Engineering*. Sydney, Australia, Mar 23-26. pp 126-133.

[7] Das, G., Lin, K., Mannila, H., Renganathan, G. & Smyth, P. (1998). Rule discovery from time series. In *proceedings of the 4th Int'l Conference on Knowledge Discovery and Data Mining*. New York, NY, Aug 27-31. pp 16-22.

[8] Dasgupta., D. & Forrest, S. (1999). Novelty detection in time series data using ideas from immunology. *In Proceedings of the 5th International Conference on Intelligent Systems* (1999).

[9] Daw, C. S., Finney, C. E. A. & Tracy, E. R. (2001). Symbolic analysis of experimental data. *Review of Scientific Instruments*.

[10] Durbin, R., Eddy, S., Krogh, A. & Mitchison, G. (1998). Biological sequence analysis: probabilistic models of proteins and nucleic acids. Cambridge University Press.

[11] Engelhardt, B., Chien, S. &Mutz, D. (2000). Hypothesis generation strategies for adaptive problem solving. In *Proceedings of the IEEE Aerospace Conference*, Big Sky, MT.

[12] Ge, X. & Smyth, P. (2000). Deformable Markov model templates for time-series pattern matching. In *proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Boston, MA, Aug 20-23. pp 81-90.

[13] Gionis, A., Indyk, P., Motwani, R. (1999). Similarity search in high dimensions via hashing. In *proceedings of 25th Int'l Conference on Very Large Databases*. Edinburgh, Scotland.

[14] Han, J. Dong, G. & Yin., Y. (1999). Efficient mining partial periodic patterns in time series database. In *Proceedings of the 15th International Conference on Data Engineering*, Sydney, Australia. pp 106-115.

[15] Hegland, M., Clarke, W. & Kahn, M. (2002). Mining the MACHO dataset, *Computer Physics Communications*, Vol 142(1-3), December 15. pp. 22-28.

[16] Hertz, G. & Stormo, G. (1999). Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, Vol. 15, pp 563-577.

[17] van Helden, J., Andre, B., & Collado-Vides, J. (1998) Extracting regulatory sites from the upstream region of the yeast genes by computational analysis of oligonucleotides. *J. Mol. Biol.*, Vol. 281, pp 827-842.

[18] Höppner, F. (2001). Discovery of temporal patterns -- learning rules about the qualitative behavior of time series. In *Proceedings of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Freiburg, Germany, pp 192-203.

[19] Indyk, P., Koudas, N. & Muthukrishnan, S. (2000). Identifying representative trends in massive time series data sets using sketches. In *proceedings of the 26th Int'l Conference on Very Large Data Bases*. Cairo, Egypt, Sept 10-14. pp 363-372.

[20] Indyk, P., and Motwani. R, Raghavan. R. & Vempala, S. (1997). Locality-preserving hashing in multidimensional spaces. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*. pp. 618-625.

[21] Keogh, E. and Kasetty, S. (2002). On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration. In the *8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* July 23 - 26, 2002. Edmonton, Alberta, Canada. pp 102-111.

[22] Keogh, E. and Pazzani, M. (1998). An enhanced representation of time series which allows fast and accurate classification clustering and relevance feedback. In *4th International Conference on Knowledge Discovery and Data Mining.* New York, NY, Aug 27-31. pp 239-243

[23] Keogh, E,. Chakrabarti, K,. Pazzani, M. & Mehrotra (2000). Dimensionality reduction for fast similarity search in large time series databases. *Journal of Knowledge and Information Systems.* pp 263-286.

[24] Lawrence, C.E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F. & Wootton, J. C. (1993). Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science*, Oct. Vol. 262, pp 208-214.

[25] Lawrence. C. &. Reilly. A. (1990). An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins*, Vol. 7, pp 41-51.

[26] Lin, J. Keogh, E. Patel, P. & Lonardi, S. (2002). Finding motifs in time series. *In the 2nd Workshop on Temporal Data Mining, at the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Edmonton, Alberta, Canada.

[27] Oates, T., Schmill, M. & Cohen, P. (2000). A Method for Clustering the Experiences of a Mobile Robot that Accords with Human Judgments. In *Proceedings of the 17th National Conference on Artificial Intelligence*. pp 846-851.

[28] Pevzner, P. A. & Sze, S. H. (2000). Combinatorial approaches to finding subtle signals in DNA sequences. In *proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*. La Jolla, CA, Aug 19-23. pp 269-278.

[29] Reinert, G., Schbath, S. & Waterman, M. S. (2000). Probabilistic and statistical properties of words: An overview. J. *Comput. Bio*., Vol. 7, pp 1-46.

[30] Rigoutsos, I. & Floratos, A. (1998) Combinatorial pattern discovery in biological sequences: The Teiresias algorithm, Bioinformatics, 14(1), pp. 55-67.

[31] Roddick, J. F., Hornsby, K. & Spiliopoulou, M. (2001). An Updated Bibliography of Temporal, Spatial and Spatio-Temporal Data Mining Research. Lecture Notes in Artificial Intelligence. 2007. pp 147-163.

[32] Scargle, J., (2000). Bayesian Blocks, A new method to analyze structure in photon counting data, Astrophysical Journal, 504, pp 405-418.

[33] Staden, R. (1989). Methods for discovering novel motifs in nucleic acid sequences. *Comput. Appl. Biosci*., Vol. 5(5). pp 293-298.

[34] Tompa, M. & Buhler, J. (2001). Finding motifs using random projections. In *proceedings of the 5th Int'l Conference on Computational Molecular Biology*. Montreal, Canada, Apr 22-25. pp 67-74.

[35] Vlachos, M., Kollios, G. & Gunopulos, G. (2002). Discovering similar multidimensional trajectories. In *proceedings 18th International Conference on Data Engineering*. pp 673-684.

[36] Yi, B, K., & Faloutsos, C. (2000). Fast time sequence indexing for arbitrary L*p* norms. In *proceedings of the 26st Intl Conference on Very Large Databases*. pp 385-394.

[37] Yi, B,K., Jagadish, H., & Faloutsos, C. (1998). Efficient retrieval of similar time sequences under time warping. IEEE International Conference on Data Engineering. pp 201-208.