

Learning Significant Alignments: An Alternative to Normalized Local Alignment

Eric Breimer and Mark Goldberg

Computer Science Department
Rensselaer Polytechnic Institute
110 Eight Street, Troy NY 12180
breime@cs.rpi.edu

Abstract. We describe a supervised learning approach to resolve difficulties in finding biologically significant local alignments, an essential technique in computational biology. It was noticed that the $O(n^2)$ -algorithm by Smith-Waterman, the prevalent tool for computing local sequence alignment, often outputs long, meaningless alignments while ignoring shorter, biologically significant alignments. Arslan *et. al.* proposed a $O(n^2 \log n)$ time algorithm which outputs a *normalized local alignment* that maximizes the degree of similarity rather than the total similarity score. Given a properly selected *normalization parameter*, the new algorithm can discover useful alignments that are not found by the Smith-Waterman algorithm. However, since the relationship between the *normalization parameter* and the input is not clear, computing useful alignments may require a repeated execution of the algorithm with different parameter values. Furthermore, the results may require an expert's feedback to determine their usefulness. We propose a learning approach to the problem. Our system, which implements this approach, uses existing biologically significant alignments to produce an algorithm that intelligently processes sub-optimal alignments. The algorithm runs in $O(n^2)$ -time and the post-processing does not require an expert's feedback to produce meaningful results. Our experiments show that the algorithm is superior to the Smith-Waterman algorithm in its ability to extract and align biologically similar regions.

1 Background

Local sequence alignment is an essential technique for identifying similarity between biological sequences [7] [9] [12]. The Smith-Waterman algorithm [17] is considered the standard tool for computing local sequence alignment. However, it was noticed (see, [2], [3], [5]) that the algorithm has two essential flaws. First, since the algorithm's objective is to find alignments with the maximal score, it often finds long alignments with a high score and misses shorter ones with a higher degree of similarity (alignment score divided by alignment length). Figure 1 (i) illustrates this flaw, called the *shadow effect*, where biologically significant "short" alignments are shadowed by "longer" alignments that may be irrelevant. Second, the algorithm often combines two or more segments of high similarity

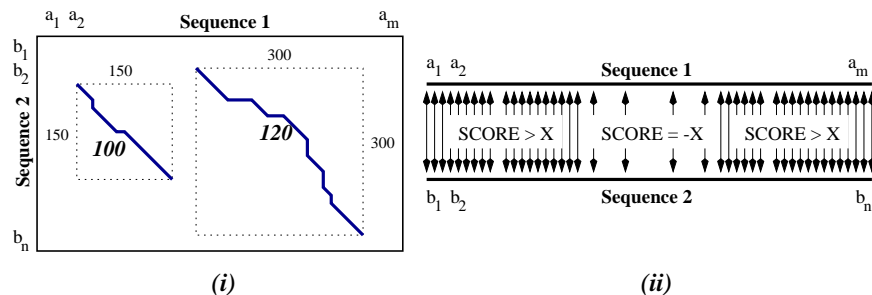


Fig. 1. Flaws of Smith-Waterman (i) Shadow effect: Sub-optimal alignments with high degree of similarity are ignored in favor of longer alignments. (ii) Mosaic effect: Two high scoring segments are joined into a single alignment that may not be biologically significant.

with intermediate low similarity segments. Figure 1 (ii) illustrates this flaw, called the *mosaic effect*, where poor scoring internal fragments are included in between otherwise meaningful alignments.

A number of attempts have been made to correct the flaws of Smith-Waterman, including unsuccessful approaches that were abandoned [13] [16], approaches that are computationally expensive [14] [19], and approaches that require sensitive heuristics [20] [4]. Further attempts were made to consider length in the computation of alignments [6] [11] [15] [18]. The most recent and successful approach was proposed by Arslan et. al. [5]. This approach seeks to maximize the *normalized score* ($SCORE_L$)

$$SCORE_L = \frac{SCORE}{LENGTH + L}$$

where $SCORE$ is the conventional alignment score and $LENGTH$ is the sum of the lengths of the two aligned segments. The value of L , called the *normalization parameter*, is used to control the degree of normalization. Arslan et. al. [5] discovered an $O(n^2 \log n)$ algorithm for computing *normalized local alignment* (*nla*) that uses the fractional programming technique developed in [10]. The algorithm, although practical and effective, is very sensitive to the value of L . For $L = 0$, a small biologically insignificant segment of exact matches will always score higher than any alignment with at least one gap or mismatch. For L larger than the length of the inputs, the algorithm's output is that of Smith-Waterman's. In general, if L is too small, the algorithm is indistinguishable from an exact matching algorithm, whereas if L is too large, the algorithm suffers from the same negative side effects of Smith-Waterman. The useful range for L is input-dependent and the relationship between the input and the appropriate value of L is generally unclear [5]. Thus, applying the algorithm requires *guessing* a preliminary value for L , and obtaining meaningful alignments may require repeated execution of the algorithm with a varying value of L . Furthermore, the results may require expert feedback to determine their usefulness. Repeatedly executing the

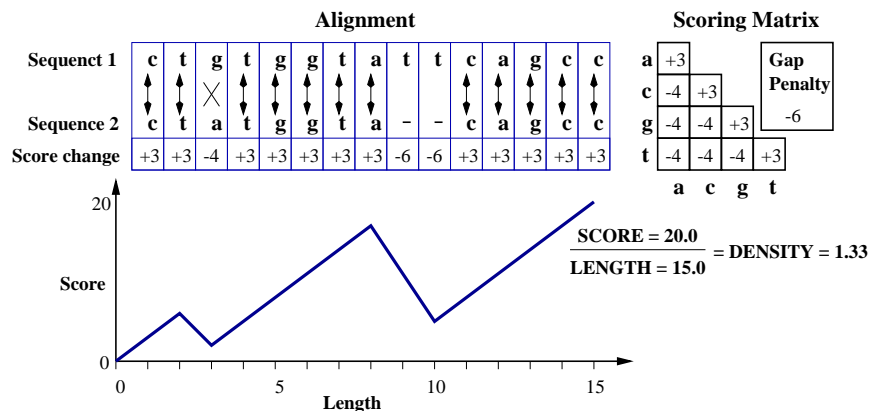


Fig. 2. The degree of similarity (density) of an alignment is defined as the alignment's score divided by the alignment's length.

$O(n^2 \log n)$ *nla* algorithm and manually evaluating the results are tedious and time consuming. For large scale applications, a more automated and an efficient process is still needed.

We propose a supervised learning approach to address the difficulties in finding biologically significant local alignments.

For the remainder of this paper, an alignment that captures biologically significant similarity will be called a *motif*. In practice, *motifs* are identified by expert biologists. Coincidental alignments without biological significance will be called *padding*. The training data will include pairs of sequence segments where all *motifs* are known. Thus, given any alignment, we can identify the specific segments that are *motifs* and those that are *padding*s. Degree of similarity, also called density, refers to the alignment's score divided by the alignment's length (see figure 2). Alignment density, although similar to *normalized score*, does not include a *normalization parameter* and defines length to be the number of aligned symbol pairs.

2 Learning Approach

Our learning approach is to use input sequences with known *motifs* to train an algorithm to align and extract these *motifs* with high probability. Our expectation is that the *motifs* possess certain characteristics that can be generalized such that the trained algorithm will perform well on similar inputs. This expectation is not unreasonable for biological sequences given that the *motifs* arise from underlying processes of mutation and conservation that are common among all inputs. We propose a learning system that outputs a specialized $O(n^2)$ local alignment algorithm. Specifically, the system learns (i) a strategy for exploring sub-optimal Smith-Waterman alignments and (ii) learns parameters for processing these alignments to prune *padding*s and highlight *motifs*. Our algorithm is

```

while  $i < m_A$  and  $j < m_B$ 
  if  $A_i^x == B_j^x$ 
    if  $A_i^y == B_j^y$ 
      Mark that  $A_i$  and  $B_j$  overlap
     $i = i + 1$ 
     $j = j + 1$ 
  else if  $A_i^x > B_j^x$ 
     $j = j + 1$ 
  else
     $i = i + 1$ 

```

Fig. 3. Algorithm for computing the overlap between two alignments A and B of lengths m_A and m_B respectively.

evaluated experimentally. The main goal is to train an algorithm that is superior to the Smith-Waterman algorithm in terms of its ability to extract and align biologically similar regions and improves upon the $O(n^2 \log n)$ runtime of the *nla* algorithm.

2.1 Sub-optimal Alignments

It is well known ([2], [3]) that the alignment's degree of similarity is an important measure in identifying biologically significant similarity. The alignments with the highest degree of similarity may not be captured by the optimal (maximum scoring) Smith-Waterman alignment, but are often captured by sub-optimal alignments [21].

Given training data, we can use Barton's algorithm [8] to efficiently output all non-overlapping maximal scoring alignments to see if a *motif* is discovered (e.g., contained within a sub-optimal alignment). We can determine the precise locations of overlap between any two alignments in linear time using the algorithm in figure 3. The sub-optimal alignments and *motifs* are both represented as a sequence of index pairs

$$A = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$$

where m is the length of alignment A and (x_i, y_i) indicates that the symbols a_{x_i} and b_{y_i} from input sequences $a = a_1 \dots a_n$ and $b = b_1 \dots b_n$ are aligned.

Once we've searched all the alignments, we can consider the following question: How many top scoring alignments must we output in order to guarantee that a *motif* will not be missed? Preliminary experiments show that if a *motif* is discovered by Smith-Waterman, it will typically be among the top k scoring alignments, where k is relatively small (e.g, $k < 20$). Figure 4 shows the percentage of *motifs* discovered by computing the top k scoring Smith-Waterman alignments. The data was generated using training samples with up to 20 *motif*'s of various size. The percentage equals the total number of overlapping pairs between the *motifs* and sub-optimal alignments divided by the total number of

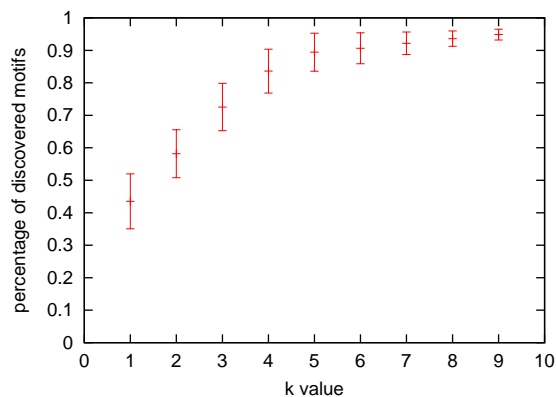


Fig. 4. The mean percentage of *motifs* discovered by computing the top k scoring Smith-Waterman alignments: The error bars indicate standard deviation. Results based on 100 training trails each containing 1 to 20 artificially generated *motifs*

pairs in the *motifs*. After a sufficient amount of training, we can determine if the discovery percentage reaches an asymptote and we can use this information to limit the number of sub-optimal alignments computed in the future.

Minimum k -value (K)

The minimum number K such that computing more than K sub-optimal alignments will not increase the probability of discovering a *motif*

Other measures can be used to determine when the discovery percentage reaches an adequate value. However, preliminary experiments indicate that the simple definition above is sufficient for producing an efficient and useful algorithm.

2.2 Alignment Density

Due to the *mosaic effect*, high scoring alignments may often contain two or more *motifs* separated by *padding*. For practical input, the degree of similarity in the *motifs* is significantly greater than the degree in the *padding* and density thresholds can be used to discriminate the two. However, this density difference is not evident if the sampling interval length is very small. Very small segments of the *padding* often possess high density. Similarly, small segments of the *motif* may possess low density. Using a large sampling interval is also problematic because a large interval will most frequently contain both *motif* and *padding* and may even encompass two or more *motifs*. Thus, pinpointing the start and end of a *motif* is prone to error. To use density thresholds effectively, one must carefully choose the interval length.

By sampling and plotting segment densities using different interval lengths we can simultaneously detect the minimum interval length and density thresh-

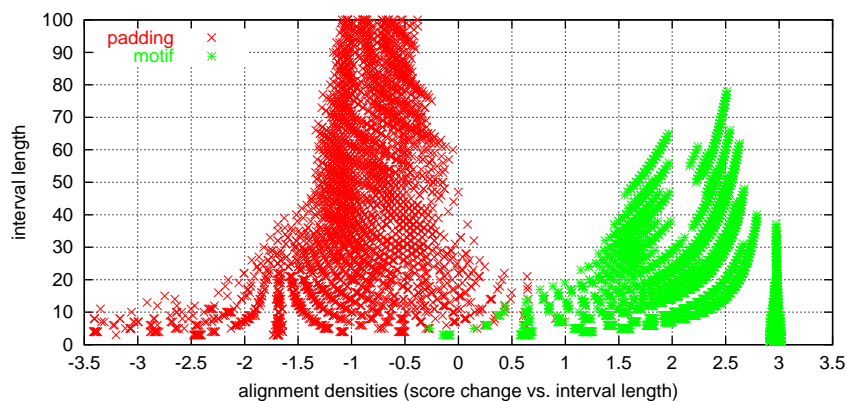


Fig. 5. Alignment densities: Given a training sample, we examine the top k Smith-Waterman alignments using varying interval lengths. The left and right clusters represent interval contained within *padding* and *motif* respectively.

olds that adequately discriminates *motif* from *padding*. Figure 5 shows alignment density plotted over interval length. The right cluster represents intervals entirely contained within a *motif*, while the left cluster represents intervals entirely contained within the *padding*. Intervals that overlap *motifs* and *padding*s are omitted from the plot. As the interval length increases the disparity between *motif* density and *padding* density increases.

To sample density, we define an interval of length w , scan each contiguously labeled segment from left to right, compute the score change s over length w and output a data point $(w, \frac{s}{w})$. Intuitively, every data point is label either *motif* or *padding* depending upon the labeling of the contiguous segment. Our goal is to find the minimal interval length L that adequately discriminates *motif* from *padding* and since very long intervals are undesirable, we can limit the length of the sampling intervals. In practice, $w = 0, 1, \dots, 100$ is sufficient and the running time of this process is negligible compared to the execution time of Barton’s algorithm. From this data, we can efficiently compute the following thresholds.

Minimum interval length (L)

The minimum length L such that all *motif* points above L fall to the right of some density threshold and all *padding* points fall to the left of that threshold.

Maximum motif threshold (M)

The maximum density M such that all *motif* points above L fall to the right of M .

Minimum padding threshold (P)

The minimum density P such that all *padding* points above L fall to the left of P .

The thresholds above can be parameterized to add flexibility. For instance, $L(r)$ would be defined as the minimum length such that r percent of all *motif*

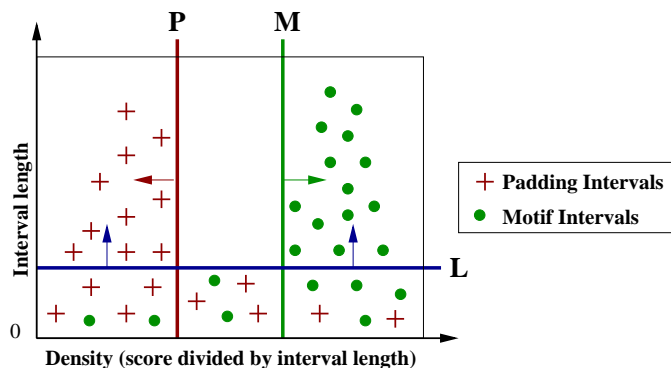


Fig. 6. L is the minimum interval length. P and M are the density thresholds for identifying *padding* and *motif* segments, respectively.

```

length = 1
Sort m in ascending order according to density
Sort p in descending order according to density
while  $m_j^d < p_k^d$ 
  length++;
  For all  $m_i$ , if  $m_i^l < length$  then remove  $m_i$ 
  For all  $p_i$ , if  $p_i^l < length$  then remove  $p_i$ 
   $j = sizeof(m) * (1 - r)$ 
   $k = sizeof(p) * (1 - r)$ 
L = length
M =  $m_j^d$ 
P =  $p_k^d$ 

```

Fig. 7. Algorithm for computing L , M , and P .

points above $L(r)$ fall to the right of some density threshold and r percent all *padding* points above $L(r)$ fall to the left of some threshold. The parameterized thresholds may be necessary to minimize the effect of a few uncharacteristic data points.

Figure 7 shows the parameterized algorithm for computing L , P , and M where r the percentage of points that must achieve the thresholds, m_i^l and m_i^d are the length and density of the i th *motif* data point, and p_i^l and p_i^d are the length and density of the i th *padding* data point.

Preliminary experiments indicate that the thresholds accurately identify the start and end of the *motifs*. Moreover, these thresholds do not significantly change as the *motif* lengths and *padding* lengths of the training data are varied, or if the number of *motifs* or their relative positioning in the input sequences vary.

2.3 Learning

Our learning system uses the observations above to generate a local alignment algorithm. To generate an algorithm, the system requires at least one training sample and implements the following steps.

Learning Stage

1. **Obtain sub-optimal alignments:** Given an input pair where all *motifs* are known, use Barton’s algorithm to output all maximal scoring Smith-Waterman alignments in order of total alignment score.
2. **Obtain k -value statistics:** Search each alignment for the known *motifs* and label each alignment accordingly. For each k , where $k = 1, \dots, m$, record the total percentage of *motifs* discovered within the top k alignments.
3. **Obtain density statistics:** From the labeled alignments, scan each *motif* segment with varying intervals and obtain the *motif* density statistics. Likewise, scan each *padding* segment and obtain the *padding* density statistics.
4. **Compute thresholds:** Repeat steps 1-3 until the training data is exhausted or until some time or computation limit has exceeded. After training is complete Compute K , L , M , and P accordingly.

Typically, Barton’s algorithm requires a parameter that determines the number of top scoring alignments to output. Barton’s algorithm requires $O(n^2)$ time and $O(n + k)$ space to output the score and the endpoints of k alignments. Afterwards the out-putted alignments can be sorted and each alignment can be generated using $O(n_i^2)$ time and $O(n_i)$ space, where n_i is the individual length of each alignment. If k is reasonably small (e.g., $k < 200$), the total time to compute and sort all k alignments is only 2 or 3 times longer than the basic Smith-Waterman algorithm.

After Barton’s algorithm is complete the next step is to search the top k alignments to see if any of the known *motifs* overlap a sub-optimal alignment. Contiguous segments of overlap are labeled as *motif*. This entire process is linear with respect to the sum of the lengths of the alignments.

After labeling the sub-optimal alignments, the system can sample the scoring densities of all the contiguously labeled segments and compute the thresholds. Training may not yield enough *motif* points to adequately determine M . This can occur if few *motifs* were embedded within the sub-optimal alignments. To compute a useful value for M , we can directly sample the *motifs* rather than sample their representations within the sub-optimal alignments. It is still necessary to compute the sub-optimal alignments to obtain the *padding* points. L must be computed in order to sample and label future alignments and selecting an appropriate L requires both *motif* and *padding* points.

Once obtained, the thresholds define a customized algorithm for computing and processing sub-optimal alignments. Assuming that the learned thresholds generalize well over a broad set of input, the evolved algorithm is expected to discover alignments that exhibit the same degree of similarity as the *motifs* used for training.

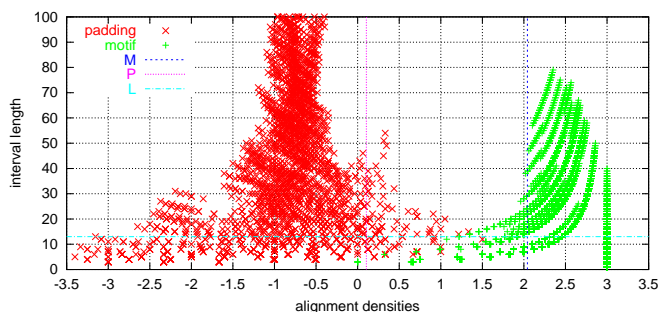


Fig. 8. Density distribution for training input. For $r = 0.995$, $L(r) = 13$, $M(r) = 2.042$, $P(r) = 0.105$ and $K = 6$.

Evolved Algorithm

1. **Obtain top K alignments:** Use Barton's algorithm to output the top K scoring alignments.
2. **Identify the *padding* segments:** Scan each alignment with a sampling interval window of length L . If the scoring density is less than P mark the segment as *padding*.
3. **Identify the *motif* segments:** Scan each alignment with a sampling interval window of length L . If the scoring density is greater than M mark the segment as *motif*.
4. **Prune and sort each segments:** After each sub-optimal alignment has been labeled, the user can choose to remove all alignment segments marked at *padding* or extract and sort all segments marked as *motif*. The *motif* segments can be sorted by density, length, or total score.

3 Experiments

We experimented on a specific region of the human ¹ and mouse ² DNA where there are highly conserved segments (called exons) that have been identified as significant by biologists. We extracted a pair of sub-regions, one from the human and one from the mouse, which shared 10 exons ranging in length from 66 to 164. Each exon possessed at least 80% similarity (e.g, 80% of the symbols in the mouse exactly match the human). The regions each contain approximately 5000 base pairs (symbols) and the total length of exons are approximately 1200 base pairs.

Figure 8 shows the results of training. Using the thresholds, we evolve an algorithm and apply it to the training data itself. We extract all potential *motifs* from the sub-optimal alignments and sort them by density. The result is that the algorithm successfully identified 8 of the 10 motifs. The algorithm reported

¹ Homo sapiens ATP-binding cassette, sub-family B (MDR/TAP), member 11

² Mus musculus ATP-binding cassette, sub-family B (MDR/TAP), member 11

9 potential motifs the shortest of which proved to be *padding*. To identify a *motif* the algorithm must compute a contiguous alignment that contains 90% of a particular *motif* (called *accuracy requirement*). Furthermore, the length of overlap between the *motif* and the computed alignment must be greater than 90% of the length of the computed alignment (called *precision requirement*). Using the same criteria, 3 of the 10 *motifs* were identified by Barton’s algorithm without performing pruning and *motif* extraction.

We applied the evolved algorithm to another sub-region, which contains 10 exons (length 64 to 172) with at least 80% similarity. The evolved algorithm was able to identify 9 out of the 10 motifs, whereas, Barton’s identified 5. Large scale experiments will be presented in the final paper.

4 Discussion

When the *normalization parameter* (L) is carefully selected, the *nla* algorithm can potentially output significantly long alignments with high score density that would not be contained in any of the sub-optimal alignments produced by Barton’s algorithm. This occurs when significant alignments are embedded in the intermediate *padding* of two adjoined *motifs* (a result of the *mosaic effect*) or when a significant alignment overlaps the beginning of a high scoring alignment (a result of the *shadow effect*). Since our approach is based on Smith-Waterman, our evolved algorithms can not eliminate these effects no matter how well it is trained. However, by isolating intermediate *padding* and recomputing the alignment, it is possible to discover these embedded alignments. Thus, it may be useful add another stage which identifies and isolates intermediate *padding* for further *motif* discovery. It is important to point out that such a stage would be impossible without an approach for identifying intermediate *padding* among sub-optimal alignments. Investigating the effectiveness of this post-processing stage is important. In general, we believe that our approach is a practical starting point for discovering significant alignments that would otherwise be hidden to due to the *mosaic effect*.

Many significant alignments are hidden from Barton’s algorithm due to the *shadow effect*. Its important to consider the likelihood of such alignments in practical data. Furthermore, its important to consider how one would determine the proper range and granularity of L -values to uncover alignments hidden by the *shadow effect*. Perhaps the most effective approach might be to apply supervised learning to automatically determine an appropriate range for L . However, training on one sample would required the repeated execution the $O(n^2 \log n)$ *nla* algorithm. Whereas, in our approach, training on one sample only requires one execution of Barton’s algorithm followed by post-processing, which together requires $O(n^2)$ time in practice. Another possible extension of our algorithm would be to apply the density thresholds during the execution of Barton’s algorithm. An edit path would be terminated if it resulted in a sufficiently long alignment that did not meet the density threshold. Such modification may allow the discovery of significant alignments that would otherwise be shadowed by excessively

long high scoring alignments. However, such modifications can adversely effect the integrity of the algorithm and may prove to be unsuccessful. Before attempting this modification, it was important to show that carefully selected length and density thresholds are useful in the post-processing of sub-optimal alignments.

It is certainly clear that our approach does not eliminate all the flaws of Smith-Waterman, but it certainly improves its discovery capabilities without adding severe computational costs. Although there have been other attempts to improve Smith-Waterman's effectiveness, our approach fundamentally differs from these previous attempts. We apply supervised learning to automatically generate thresholds to post-process sub-optimal alignments. Thus, we provide a system where existing knowledge of biological similarities can be used to evolve a customized Smith-Waterman based approach. Rather than applying human-generated heuristics at the post-processing stage, we have provided a general framework for learning and applying heuristics automatically.

References

1. Alexandrov, N., Solovyev, V.: Statistical significance of ungapped alignments. Pacific Symp. on Biocomputing (1998) 463-472
2. Altschul, S., Erickson, B.: Locally optimal sub-alignments using nonlinear similarity functions. Bulletin of Mathematical Biology **48** (1986) 633-660
3. Altschul, S., Erickson, B.: Significance levels for biological sequence comparison using nonlinear similarity functions. Bulletin of Mathematical Biology **50** (1988) 77-92
4. Altschul, S., Madden, T., Schaffer, A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.: Gapped Blast and Psi-Blast: a new generation of protein database search programs. Nucleic Acids Research **25** (1997) 3389-3402
5. Arslan, A., Egecioglu, Ö., Pevzner, P.: A new approach to sequence comparison: normalized sequence alignment. Proceeding of the Fifth Annual International Conference on Molecular Biology (2001) 2-11
6. Arslan, A., Egecioglu, Ö.: An efficient uniform-cost normalized edit distance algorithm. 6th Symp. on String Processing and Info. Retrieval (1999) 8-15
7. Bafna, V., Huson, D.: The conserved exon method of gene finding. Proc. of the 8th Int. Conf. on Intelligent Systems for Molecular Bio. (2000) 3-12
8. Barton, G.: An efficient algorithm to locate all locally optimal alignments between two sequences allowing for gaps. Computer Applications in the Biosciences **9** (1993) 729-734
9. Batzoglou, S., Pachter, L., Mesirov, J., Berger, B., Lander, E.: Comparative analysis of mouse and human DNA and application to exon prediction. Proc. of the 4th Annual Int. Conf. on Computational Molecular Biology (2000) 46-53
10. Dinkelbach, W.: On nonlinear fractional programming. Management Science **13** (1967) 492-498
11. Egecioglu, Ö., Ibel, M.: Parallel algorithms for fast computation of normalized edit distances. Proc. of the 8th IEEE Symp. on Par. and Dist. Proc. (1996) 496-503
12. Gelfand, M., Mironov, A., Pevzner P.: Gene recognition via spliced sequence alignment. Proc. Natl. Acad. Sci. USA **93** (1996) 9061-9066
13. Goad, W., Kanehisa, M.: Pattern recognition in nucleic acid sequences: a general method for finding local homologies and symmetries Nucleic Acids Research **10** (1982) 247-263

14. Huang, X., Pevzner, P., Miller, W.: Parametric recomputing in alignment graph. Proc. of the 5th Annual Symp. on Comb. Pat. Matching (1994) 87-101
15. Oommen, B., Zhang, K.: The normalized string editing problem revisited. IEEE Trans. on PAMI **18** (1996) 669-672
16. Seller, P.: Pattern recognition in genetic sequences by mismatch density. Bull. of Math. Bio. **46** (1984) 501-504
17. Smith, T., Waterman, M.: Identification of common molecular subsequences. Journal of Molecular Biology **147** (1981) 195-197
18. Vidal, E., Marzal, A., Aibar, P.: Fast computation of normalized edit distances. IEEE Trans. on PAMI **17** (1995) 899-902
19. Zhang, Z., Berman, P., Miller, W.: Alignments without low-scoring regions. J. Comput. Biol. **5** (1998) 197-200
20. Zhang, Z., Berman, P., Wiehe, T., Miller, W.: Post-processing long pairwise alignments. Bioinformatics **15** (1999) 1012-1019
21. Zuker, M.: Suboptimal sequence alignment in molecular biology: alignment with error analysis. Journal of Molecular Biology **221** (1991) 403-420