Combinatorial Approaches to Finding Subtle Signals in DNA Sequences

Pavel A. Pevzner[1,2,3] and **Sing-Hoi Sze**[3]
Departments of Mathematics[1], Molecular Biology[2], and Computer Science[3],
University of Southern California, Los Angeles, CA 90089-1113.
ppevzner@hto.usc.edu, ssze@hto.usc.edu

Abstract

Signal finding (pattern discovery in unaligned DNA sequences) is a fundamental problem in both computer science and molecular biology with important applications in locating regulatory sites and drug target identification. Despite many studies, this problem is far from being solved: most signals in DNA sequences are so complicated that we don't yet have good models or reliable algorithms for their recognition. We complement existing statistical and machine learning approaches to this problem by combinatorial approaches that proved to be successful in identifying very subtle signals.

Keywords: pattern discovery, multiple alignment

Introduction

Perhaps the first signal in DNA was found in 1970 by Hamilton Smith after the discovery of the Hind II restriction enzyme. Finding the palindromic site of this signal was not a simple problem in 1970; in fact, Hamilton Smith published two consecutive papers on Hind II, one on enzyme purification and the other one on finding the enzyme's recognition signal (Kelly & Smith 1970).

Looking back to the early 1970s, we realize that Hamilton Smith was lucky: restriction sites are the simplest signals in DNA. Thirty years later, despite many studies, the signal finding problem is far from being solved. Most signals in DNA sequences (promoters, splicing sites, etc.) are so complicated that we don't yet have good models or reliable algorithms for their recognition. In particular, we are unaware of any algorithm that solves the following "simple" problem:

Challenge Problem. Find a signal in a sample of sequences, each 600 nucleotides long and each containing an unknown signal (pattern) of length 15 with 4 mismatches.

Such a $(15, 4)$-signal is rather strong, stronger than the $(6, 0)$-signal of Hind II restriction site (it appears approximately once per 10,000 nucleotides in an i.i.d. sample). However, some of the best currently available algorithms, like CONSENSUS (Hertz & Stormo 1999), Gibbs sampler (Lawrence *et al.* 1993; Neuwald, Liu, & Lawrence 1995), and MEME (Bailey & Elkan 1995) fail to find such a signal even in i.i.d. samples with probabilities of each nucleotide 1/4 (Table 1). This paper explains why the problem above presents a bottleneck for the current algorithms and proposes new combinatorial approaches to signal finding that solves the Challenge Problem and other problems that are beyond the possibilities of current algorithms.

In its simplest form, the signal finding problem can be formulated as follows: given a sample of sequences and an unknown pattern (signal) that appears at different unknown positions in each sequence, can we find the unknown pattern? If an l-letter pattern appears exactly in each sequence, one can find the signal by a straightforward enumeration of all l-letter substrings that appear in the sample. However, biological signals are subject to mutations and usually don't appear exactly. A natural model is to allow the unknown pattern to appear with some number of mismatches, insertions, and deletions in sample sequences.

Why is finding a rather strong $(15, 4)$-signal so difficult? The problem is that any two instances of the mutated (l, d)-signal may differ in as many as $2d$ positions. As a result, any two instances of the signal in the Challenge Problem may differ by as many as 8 mutations, a rather large number. The numerous spurious similarities with 8 mutations in 15 positions disguise the real signal and lead to difficulties in signal finding.

We will approach the problem from the combinatorial viewpoint. In the first *winnowing* approach, we, in contrast to many signal finding algorithms, concentrate on non-signals (spurious similarities) rather than the signal itself. Given a sample of sequences, the WINNOWER algorithm constructs a graph with vertices corresponding to substrings from the sample sequences and edges corresponding to similar substrings. Some edges in this graph connect the instances of the signal in the sample sequences (*signal edges*), while others correspond to spurious similarities (*spurious edges*). The spurious edges disguise the signal edges and make signal finding difficult. For example, in the Challenge Problem, there are $\approx 20,000$ spurious edges for each signal edge. WINNOWER deletes the spurious edges from this possibly very large graph and guarantees that all signal edges

sequence length	100	200	300	400	500	600	700	800	900	1000
CONSENSUS	0.92	0.94	0.53	0.31	0.29	0.07	0.15	0.09	0.01	0.04
GibbsDNA	0.93	0.96	0.51	0.46	0.29	0.12	0.09	0.34	0.00	0.12
MEME	0.91	0.78	0.59	0.37	0.17	0.10	0.02	0.03	0.00	0.00
WINNOWER ($k = 2$)	0.98	0.98	0.97	0.95	0.97	0.92	0.58	0.02	0.02	0.02
WINNOWER ($k = 3$)	0.98	0.98	0.97	0.94	0.97	0.92	0.90	0.93	0.90	0.88
SP-STAR	0.98	0.98	1	0.96	0.96	0.84	0.83	0.69	0.64	0.23

Table 1: Comparison of performance of the various algorithms for samples with implanted $(15, 4)$-signals (FM model). The sequence length n varies from 100 to 1000. Every entry in the table represents the average performance coefficient over eight samples each containing 20 i.i.d. sequences of length n. For WINNOWER ($k = 3$), the performance stays over 0.80 even for sequence length 1300.

are preserved. In most cases, the final graph is small enough to reveal the signal edges and to detect the unknown signal. An important feature of WINNOWER is that it can find multiple signals.

Our winnowing approach reduces the signal finding problem to finding large *cliques* in multipartite graphs. Other signal finding algorithms implicitly try to find large cliques in the same graph either by a greedy algorithm (e.g., CONSENSUS) or by a Metropolis style algorithm (e.g., Gibbs sampler). In fact, there are some similarities between the Gibbs sampler and the Metropolis process on cliques studied by Jerrum (1992). However, while Metropolis style algorithms are successful for some combinatorial optimization problems, they are notoriously inefficient for others. In particular, in the paper titled *Large cliques elude the Metropolis process*, Jerrum (1992) proves that the Metropolis process takes super-polynomial time to find a clique that is only slightly better than that produced by the greedy heuristic. This result probably helps to explain why Gibbs sampler was just slightly better than the greedy CONSENSUS algorithm in our tests (Table 1).

Probably the best tools for finding short l-letter signals are the pattern-driven algorithms (Brazma *et al.* 1998) that test all 4^l l-letter patterns, score each pattern by the number of approximate occurrences in the sample (or by a more involved function) and find the high-scoring patterns (Staden 1989; Pesole *et al.* 1992; Wolfertstetter *et al.* 1996; van Helden, Andre, & Collado-Vides 1998; Tompa 1999). However, an exhaustive search through all 4^l l-letter patterns becomes impractical for $l > 10$ and Tompa (1999) raised the problem of extending this approach for longer patterns. One way around this problem is to limit the search to the patterns appearing in the sequences from the sample (Bailey & Elkan 1995; Li, Ma, & Wang 1999; Gelfand, Koonin, & Mironov 2000). If, by chance, the signal pattern has an exact occurrence in the sample, this approach is as good as the rigorous pattern-driven approach. If not (which is usually the case for biological samples), the hope is that the instances of the pattern appearing in the sample will reveal the pattern itself. However, in the case of subtle signals, this approach needs to be taken with caution. The problem is that numerous spurious similarities disguise the instances of the signal since the scores of random patterns may exceed the scores of the instances of the real pattern (although it does not exceed the score of the real pattern itself!). In our second SP-STAR approach, we emulate the pattern-driven algorithm by selecting a scoring function and a local improvement heuristic that allow one to better separate a signal from noise. We further extend the proposed algorithms to handle corrupted samples (i.e., samples with a signal present only in a portion of all sequences) and samples with biased nucleotide composition.

Finding Signals via Winnowing

Let $\mathcal{S} = \{s_1, \ldots, s_t\}$ be a sample of t n-letter sequences, each sequence containing an (l, d)-signal, i.e., a signal of length l with d mismatches. Since every two instances of the signal differ by at most $2d$ mismatches, one may generate $\binom{t}{2}$ pairwise sequence comparisons that reveal all pairs of substrings of length l that have at most $2d$ mismatches. However, for a $(15, 4)$-signal, only a small portion of these pairs correspond to the real signal while most pairs correspond to spurious similarities. Most existing algorithms try to find the subtle signal in this forest of spurious similarities, a very difficult problem. For example, CONSENSUS (Hertz & Stormo 1999) assembles the best pairwise similarities into a multiple similarity in a hope that the best pairwise similarities capture the signal. This may not be true for the subtle $(15, 4)$-signals. We use a different approach: instead of trying to find a signal in a forest of spurious similarities, we first try to cut the forest. After it is done, the signal finding problem becomes simple since in most cases the remaining similarities correspond to the signal. However, the forest-cutting procedure is non-trivial and time-consuming since we have to ensure that only the spurious similarities are being cut-off and the signal is retained.

Following Vingron & Pevzner (1995), we represent the signal finding problem in a simple geometric framework. Consider an integer point $(p_1, \ldots, p_t)$ in t-dimensional space, for which we do not know the coordinates (p_i is the position of the signal in the sequence s_i from the sample). Suppose we observe the projections (p_i, p_j) of this point onto each pair of dimensions i and j $(1 \leq i < j \leq t)$ as well as some other points (spurious similarities or noise). Suppose also that we cannot distinguish the point representing the projection of the signal from the ones representing noise. The *winnowing* problem is to reconstruct the t-dimensional point $(p_1, \ldots, p_t)$ given $\binom{t}{2}$ projections (with noise).

For the Challenge Problem, the signal point in every pairwise comparison is hidden among $\approx 20,000$ spurious points and finding the signal among that many spurious similarities is a daunting task. However, the signal $(p_1, \ldots, p_t)$ generates *consistent* points in $\binom{t}{2}$ projections (in the sense of Vingron & Pevzner (1995)), in contrast to *inconsistent* points corresponding to spurious similarities. This observation allows one to filter out most (though possibly not all) spurious similarities.

A naive version of the winnowing idea was first studied in a context of multiple alignment by Vihinen (1988). Later, Roytberg (1992) proposed superimposing pairwise dot-matrices by choosing one reference sequence and using it as an instrument for filtering spurious edges. Vingron & Argos (1991) and Vingron & Pevzner (1995) further developed this idea (for multiple alignment) by using any two reference sequences for edge filtration in an iterative fashion. In this paper, we further develop the winnowing approach and improve the filtration efficiency by using any number of reference sequences, adjusting this idea for finding subtle signals and arriving at an algorithm that can detect increasingly more and more subtle signals at the expense of an increase in the running time.

The WINNOWER Algorithm

Given a sample $\mathcal{S} = \{s_1, \ldots, s_t\}$, a parameter l (length of the signal), and a parameter d (maximum number of mutations), construct a graph $G(\mathcal{S}, l, 2d)$ as follows. For each position j in sequence s_i, construct a vertex representing the substring s_{ij} of length l starting at position j in s_i, where $1 \leq j \leq n-l+1$ ranges over all valid starting positions. Connect vertex s_{ij} with vertex s_{pq} by an edge if $i \neq p$ and the distance between s_{ij} and s_{pq} does not exceed $2d$. For simplicity, we consider signals with substitutions only (Hamming distance), although the approach can be generalized to include insertions and deletions (compare with Rocke & Tompa (1998)).

A *clique* in a graph is a set of vertices any two of which are connected by an edge. Any two occurrences of an (l, d)-signal in $\mathcal{S}$ correspond to an edge in the $G(\mathcal{S}, l, 2d)$ graph (the expected number of mismatches between any two occurrences of the signal is $2d - 4d^2/3l$, slightly less than $2d$). Therefore, any (l, d)-signal corresponds to a *clique* of size t in $G(\mathcal{S}, l, 2d)$ thus reducing the signal finding problem to finding large cliques in a graph. Finding large cliques in a graph is a difficult problem in general (Garey & Johnson 1979). However, in signal finding, we usually deal with special kinds of multipartite graphs with "almost random" edges corresponding to spurious similarities. This problem is related to the *hidden clique* problem in random graphs (Jerrum 1992; Alon, Krivelevich, & Sudakov 1998). We explore the specifics of such graphs and propose an approach based on removing edges that surely are not contained in a large clique.

WINNOWER uses the notion of an extendable clique. A vertex u is a *neighbor* of a clique $C = \{v_1, \ldots, v_k\}$ if $\{v_1, \ldots, v_k, u\}$ is a clique in the graph. A clique is called *extendable* if it has at least one neighbor in every part of the multipartite graph G. An edge is called *spurious* if it does not belong to any extendable clique of size k. One way to impose increasingly strict conditions as k increases is to ensure that all spurious edges are deleted after winnowing and only extendable cliques remain in the final graph. This approach was taken in Vingron & Argos (1991) and Vingron & Pevzner (1995) for $k = 2$. However, deleting the edges that do not belong to extendable cliques is too weak for filtering spurious edges in the case $k > 2$. WINNOWER is based on the observation that every edge in a maximal t-clique in G belongs to at least $\binom{t-2}{k-2}$ extendable cliques of size k. This observation leads to filtering *inconsistent* edges, i.e., edges that belong to less than $\binom{t-2}{k-2}$ extendable cliques. For $k = 3$, this condition removes edges that belong to less than $t - 2$ extendable cliques, a rather powerful filtering procedure.

For $k = 1$, a vertex u is a neighbor of vertex v if (u, v) is an edge in the graph. The simplest strategy for winnowing is to ensure that in the final graph, every vertex has at least one neighbor in every part of G and to delete the vertices that do not satisfy this condition (compare with Roytberg (1992)). This approach is often inadequate, giving a large final graph even for relatively simple signals. For $k = 2$, a vertex u is a neighbor of an edge (v, w) if $\{u, v, w\}$ is a triangle in the graph. WINNOWER ensures that every edge has at least one neighbor in every part of G (compare with Vingron & Pevzner (1995)). This algorithm works well in many cases (better performance than CONSENSUS, GibbsDNA and MEME for the Challenge Problem), but requires further improvement in finding very subtle signals that are beyond the possibilities of existing algorithms. This improvement is achieved by running WINNOWER for cliques of size $k > 2$ (Table 1).

WINNOWER is an iterative algorithm that converges to a collection of extendable cliques by deleting inconsistent edges. Define $x_{i,j,p,q}$ as 0 if the jth letter in s_i coincides with the qth letter in s_p, and as 1 otherwise. If the number of mismatches between s_{ij} and s_{pq} is $d_{i,j,p,q}$, the number of mismatches between $s_{i,j+1}$ and $s_{p,q+1}$ is $d_{i,j+1,p,q+1} = d_{i,j,p,q} - x_{i,j,p,q} + x_{i,j+l,p,q+l}$. This observation leads to an $O(N^2)$ algorithm for the construction of the graph $G(\mathcal{S}, l, d)$, where N is the total number of nucleotides in $\mathcal{S}$. Below we estimate WINNOWER's running time and demonstrate that it can be run with $k = 3$ and even larger clique sizes for many challenging signal finding problems (in contrast to its rather high $O(N^{k+1})$ complexity in the general case).

We estimate the running time of WINNOWER for a random multipartite graph with t n-vertex parts and with probability of every edge being p. The expected number of edges between vertex v and a given part of this graph is $D = pn$. A given set of k vertices in different parts of this graph forms a clique with probability $p^{\binom{k}{2}}$, and therefore, the expected number of k-cliques in this graph is $\binom{t}{k} n^k p^{\binom{k}{2}}$. Checking whether each clique is extendable in this graph requires roughly $O(ktD)$ time per clique, and therefore, the running time of the algorithm can be estimated as $O(\binom{t}{k} n^k p^{\binom{k}{2}} ktD)$ provided that the list of k-cliques is given. For $k = 3$, it gives an $O((tD)^4)$ estimate of the running time ($D \approx 30$ for the Challenge Problem). A simple probabilistic analysis demonstrates that the practical values of p for WINNOWER with k-cliques are limited to $p \leq 1/\sqrt[k]{n}$. It implies that the running time of WINNOWER for practical values of p is bounded by $O(t^3 n^{1.5})$ for $k = 2$ and by $O(t^4 n^{2.66})$ for $k = 3$. For many practical instances, WINNOWER has about the same running time for $k = 2$ and for $k = 3$. The explanation for this surprising observation is that, for $k = 3$, WINNOWER converges in a smaller number of iterations since the condition for edge removal is much stronger.

In the absence of prior knowledge about the signal, it is difficult to choose the parameters l and d for graph construction. One way to solve this problem is to run the algorithm with progressively increasing d until the final graph is not empty (for a fixed l). This approach does not lead to a significant increase in the running time since, for small d, the graph G has very few edges. Tests of WINNOWER demonstrated that usually, depending on l and d, the final graph is either almost empty (no signal), small with one large clique (signal), or very large (noise). As a result, the size of the final graph can be used to evaluate the reliability of found signals and to estimate the length of the signal l. To generalize WINNOWER for the case of corrupted samples and samples with biased nucleotide composition, we further relax the edge removal condition and construct the multipartite graph based on the notion of the statistical significance of edges rather than the distance between substrings. As a result, biases in frequencies, irregularities, and repeats in DNA sequences affect only the pre-processing stage of the algorithm (graph construction) and do not affect the algorithm itself. It amounts to assigning to every edge the probability that s_{ij} and s_{pq} differ by the observed number of mutations (assuming the empirical probabilities of nucleotides in the biased sample). Afterwards, we retain only the edges in the graph that reflect significant similarities.

The SP-STAR Algorithm

WINNOWER requires substantial computational resources (both time and memory) and becomes rather slow for finding subtle signals in very large samples. Another drawback is that it treats all edges of the graph $G(\mathcal{S}, l, d)$ equally without distinguishing between edges corresponding to high and low similarities. An alternative approach is to score each candidate signal (collection of substrings in $\mathcal{S}$) and to formulate the signal finding problem as a combinatorial optimization problem. For simplicity, we first assume that the length of the signal is known and that it appears (with some mutations) in every sequence. Later we extend the approach to deal with biologically more adequate cases of corrupted samples and samples with biased nucleotide composition.

Define the distance $d(P, s_i)$ between a pattern P and a sequence s_i from the sample as the minimum distance between P and s_{ij} for $1 \leq j \leq n - l + 1$. Define the distance between the pattern P and the sample $\mathcal{S}$ as $d(P, \mathcal{S}) = \sum_{1 \leq i \leq t} d(P, s_i)$. A pattern-driven algorithm for the *Consensus Problem* tests all 4^l l-letter patterns P and finds a *median string* P (Li, Ma, & Wang 1999) with the minimum distance $d(P, \mathcal{S})$ to the sample. The pattern-driven approach was first advocated by Staden (1989) and proved to be very successful in a recent work by Tompa (1999) (with a different scoring function). However, an exhaustive search through all 4^l l-letter patterns to compute $\min_{\text{all } l\text{-letter patterns } P} d(P, \mathcal{S})$ becomes impractical for large l and Tompa (1999) raised a problem of extending this approach for longer patterns. To emulate the pattern-driven approach for long patterns, Fraenkel *et al.* (1995) and Rigoutsos & Floratos (1998) proposed first building a dictionary of frequent short patterns and combining them into longer patterns afterwards. Recently, Li, Ma, & Wang (1999) proposed a different alternative to exhaustive search by devising a polynomial time approximation scheme for finding the median string. Their approach is based on the generation and analysis of all combinations of r l-letter substrings from the sample. Since it is prohibitively time-consuming for large r, the question arises of how good this approach is for practical values of r, in particular, for $r = 1$. In this case, one first computes $\min_{P \in \mathcal{S}} d(P, \mathcal{S})$ where the search is limited to all l-letter patterns P appearing in the sequences from the sample. Gelfand, Koonin, & Mironov (2000) recently applied a similar approach for finding regulatory sites in *Archaea*.

We explain the bottleneck of this approach for finding subtle signals. Let P be an (l, d)-signal that appears as $P_1, \ldots, P_t$ in sequences $s_1, \ldots, s_t$ from the sample $\mathcal{S}$. The Gelfand, Koonin, & Mironov (2000) approach is based on a hope that

$$\min_{P \in \mathcal{S}} d(P, \mathcal{S}) = \min_{1 \leq i \leq t} d(P_i, \mathcal{S})$$

since in this case the selected pattern P_i represents an instance of the signal, and therefore approximates the signal well. However, it is not true for subtle signals and instead of selecting P_i, $\min_{P \in \mathcal{S}} d(P, \mathcal{S})$ often selects a random pattern that has nothing in common with the signal. The problem is that in the case of subtle signals, there is a good chance that $d(W, \mathcal{S})$ is less than

$\min_{1\leq i\leq t} d(P_i, \mathcal{S})$ for a random pattern W. If the best instances of P_i in sequences $s_1, \ldots, s_t$ are $P_1, \ldots, P_t$, we expect that $d(P_i, \mathcal{S})$ is close to $2d(t-1)$ (since P_i and P_j typically differ in $2d$ positions). However, for subtle signals, any pattern W often has approximate occurrences $W_1, \ldots, W_t$ in every sequence within distance $2d$ and therefore $d(W, \mathcal{S})$ is close to $2d(t-1)$ with high probability for a randomly chosen pattern W. As a result, $d(W, \mathcal{S})$ for a random pattern W may turn out to be lower than $d(P_i, \mathcal{S})$, thus leading to the selection of a random pattern instead of P_i and missing a signal.

The following sum-of-pairs scoring used in SP-STAR provides a better separation between the signal pattern and random patterns and allows one to detect subtle signals. Let $W \in \mathcal{S}$ be a string of length l and let $W_1, \ldots W_t$ be the best instances of W in sample sequences $s_1, \ldots, s_t$ (ties are broken arbitrarily). SP-STAR uses a *sum-of-pairs* function $D(W, \mathcal{S}) = \sum_{1\leq i<j\leq n} d(W_i, W_j)$ that better separates signal from noise than $d(W, \mathcal{S})$. The reason is that the typical value of $d(W_i, W_j)$ for a random string W may be as large as $4d$ (since W_i and W_j are strings typically distance $2d$ apart from W), while the typical value for $d(P_i, P_j)$ is $2d$ (since P_i and P_j are correlated through (unknown) signal pattern P and are expected to be distance $2d$ apart).

SP-STAR is a heuristic that first finds a pattern W corresponding to $\min_{W\in\mathcal{S}} D(W, \mathcal{S})$. This can be achieved in $O(N^2)$ time by using the same argument as in the graph construction phase of the WINNOWER algorithm, where N is the total number of nucleotides in $\mathcal{S}$. The advantages of the sum-of-pairs scoring for signal finding were also demonstrated in a recent work by Akutsu, Arimura, & Shimozono (2000). However, in the case of subtle signals, sum-of-pairs scoring points out to some instances of the signal rather than to the signal itself and some further work is needed to find the signal. SP-STAR further uses a local improvement strategy to improve the initially found signal. Define the *majority string* for a collection of strings $\mathcal{W} = \{W_1, \ldots, W_t\}$ as the string W', whose ith letter is the most frequent ith letter in $\mathcal{W}$, with ties broken arbitrarily. The best occurrences of a majority string W' in each sequence s_i define a new potential signal $W'_1, \ldots, W'_t$. SP-STAR repeats this procedure until there are no more improvements in the score. Since local improvements may take a long time, they are performed only on a fraction of the best initial signals.

Instead of the majority string approach, this local improvement strategy can be performed in many ways, for example, using profiles, Gibbs sampler or any other local improvement algorithms. Computational experiments indicate that the following local improvement procedure gives good results in practice. The idea is that some of the positions of the patterns found at intermediate steps may not be significant and should be ignored while looking for the best instances of the majority strings. The pattern score $D(W, \mathcal{S})$ can be decomposed into a sum of position scores and a position is defined as significant if the corresponding position score is below a threshold. Insignificant positions are ignored while finding the best instances of the majority strings. The procedure is repeated until there are no more improvements. At this point, several variants of the (possibly gapped) majority string are constructed to define new potential signals and to test for further improvements. These steps are repeated again until there are no improvements.

Extensions to the SP-STAR Algorithm

Signals with Unknown Length. When the length of the signal l is unknown, SP-STAR can be run for each length in consideration over a range of lengths. The problem with this approach is that (i) it is rather slow, and (ii) the sum-of-pairs distance is not suitable for comparing signals of different lengths and it is unclear what the "best" length is. We found that a simple transition from distance-based scores to similarity-based scores allows us to analyze signals of different lengths. The resulting algorithm has approximately the same running time as the original distance-based algorithm with the fixed signal length l.

If a candidate signal W is represented by a collection of the best instances $W_1, \ldots, W_t$, the similarity score for W is defined as $S(W) = \sum_{1\leq i<j\leq t} S(W_i, W_j)/\binom{t}{2}$, where $S(W_i, W_j)$ is the *similarity* between W_i and W_j, defined as the sum of premiums for matches and penalties for mismatches (e.g., we found that using $+2$ as the match premium and -1 as the mismatch penalty works well). With the similarity score $S(W)$, we usually define a position as significant if its score is positive.

Let $[l_{min}, l_{max}]$ be the range of target lengths under investigation. We run SP-STAR with strings of length l_{max} but change the scoring approach by scoring the best subregion of length between l_{min} and l_{max} with maximum sum-of-column score (instead of the full-length region of length l_{max}). For the local improvement step, instead of finding a majority nucleotide for all l_{max} positions, only the positions of the best-scoring subregion are considered while other positions are ignored when finding the best instances of the majority string in each sample sequence. Each instance is again of length l_{max} and the best scoring subregion instead of the entire region is used to define the new signal. The local improvement procedure also tries to incorporate an extra position to the left or to the right of the best subregion (within the entire region) as variants of the majority string in order to better estimate the signal length.

Gapped Signals. The version of SP-STAR described in the previous section already can deal with gapped signals by returning only significant positions of the found signal. Another way to deal with gapped signals is to change the definition of the score of a signal (collection of strings) by summing up only positive column scores and to use only these positions while looking for the best instances of the majority strings. This score only considers significant positions and thus may better reflect the alignment.

Finding Signals in Samples with Biased Nucleotide Composition. To deal with samples with biased nucleotide composition, we use the background probabilities to define new (information-theoretic) match premiums and mismatch penalties. As a result, the best instances of the string in the sample are controlled by the background distribution of nucleotides.

Finding Signals in Corrupted Samples. Most biological samples are corrupted, i.e., the signal is present only in a fraction of sequences from the sample. For example, usually only a third of sequences in the bacterial Ribosome Binding Site samples contain an analog of the Shine-Dalgarno signal. This raises a problem of comparing the candidate signals that appear in a different number of sequences. An elegant approach to this problem was recently suggested by Tompa (1999). We found that a naive normalization of the score by the number of its occurrences in the sample works well in practice. In the future, we plan to incorporate Tompa (1999) scoring in our algorithm.

The algorithm for corrupted samples sorts the best instances of the signal in decreasing order of fitness to the majority string, and computes the normalized score for the top i instances ($1 \leq i \leq t$). The next signal (collection of strings) is set to be the one with the best normalized score over all i. The next majority string is computed from these i instances rather than from all the instances. Although experimental results show that this approach often overestimates the number of sequences containing the signal, the error is usually small.

Test Samples and Performance Evaluation

A common test set for signal finding in DNA sequences is a sample of experimentally confirmed *E. coli* CRP binding sites containing 18 105-nucleotide sequences (Stormo & Hartzell III 1989). There are a total of 23 CRP binding sites in this sample, so some sequences contain more than one site. Similar to other signal finding algorithms, SP-STAR finds the correct sites in 17 (out of 18) sequences and forms a majority string **TGTGA**nnnngn**TCACA**, which is close to the consensus. A good performance of most signal finding algorithms for this sample is not surprising since the average mutation rate among ten conservative positions of the CRP signal is $p = 0.23$, a bit below the level when difficulties in signal finding begin to occur (for samples with 105-nucleotide sequences).

To compare the performance of different signal finding algorithms, it is important to have complete control over the characteristics of test samples. Following Workman and Stormo (2000), we generate samples $\mathcal{S}$ from i.i.d. sequences of fixed length and implant randomly mutated signals at randomly chosen positions in these sequences. Two different approaches are used to implant a signal pattern in sample sequences. In the first approach, a randomly mutated pattern with exactly d substitutions is implanted in each sequence (Fixed number of Mutations or FM model). In the second approach, each nucleotide of the signal pattern is mutated with probability p into any of the three remaining nucleotides (Variable number of Mutations or VM model). The FM model with d mutations is similar to the VM model with probability of mutation $p = d/l$.

Let K be the set of known signal positions in a sample and let P be the set of predicted positions. For WINNOWER, P consists of all positions covered by vertices in the final graph, where vertex s_{ij} covers all positions from j to $j+l-1$ in s_i. We use either the *performance coefficient* $|K \cap P|/|K \cup P|$ or the standard *correlation coefficient* (Brazma *et al.* 1998) for performance evaluation.

Figure 1 illustrates the performance of CONSENSUS, GibbsDNA and MEME, while Figure 2 illustrates the performance of WINNOWER ($k = 2$) and SP-STAR (FM model with unbiased nucleotide composition). The tests revealed a sharp drop in the performance as soon as the number of mutations exceeds a certain threshold. This phenomenon makes it difficult to compare the performance of different algorithms in terms of the performance coefficient or the correlation coefficient (since it is either close to 0 or close to 1).

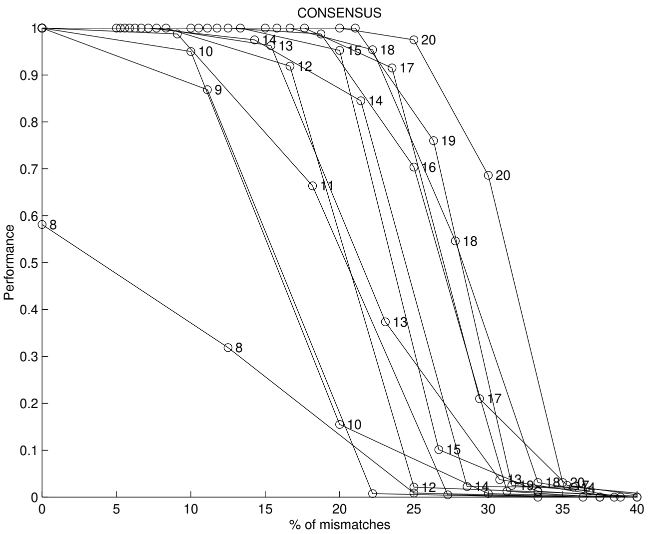
To better reveal the merits and demerits of different algorithms, we fix the parameters l and d and vary the length n of the sequences in the sample (Table 1). CONSENSUS, GibbsDNA and MEME start to break at length 300 to 400, while WINNOWER ($k = 2$) starts to break at length 700 to 800. WINNOWER ($k = 3$) works through the whole range of lengths. SP-STAR breaks at length 800 to 900 (random patterns start to out-score the signal pattern at these lengths).

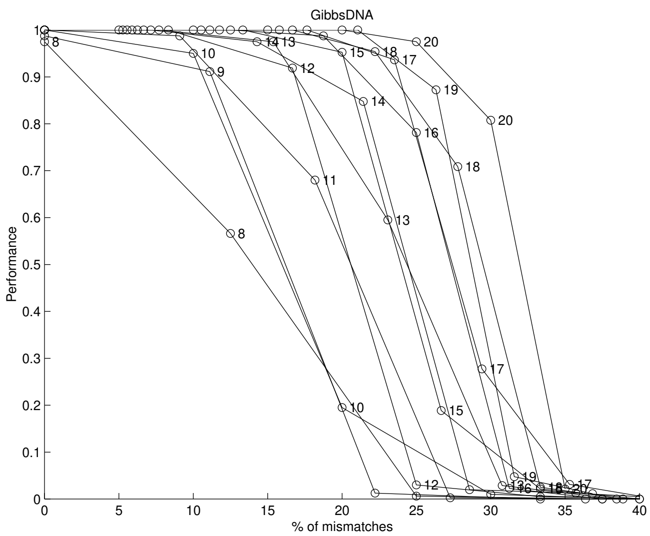
Figure 3 illustrates the performance of SP-STAR, CONSENSUS, GibbsDNA and MEME under the VM model with unbiased nucleotide composition. The performance of these algorithms in the VM model improves due to the fact that, with high probability, one of the implanted strings becomes very close to the signal string and therefore is easier to detect. SP-STAR still performs better than CONSENSUS, GibbsDNA or MEME for subtle signals. Performance for WINNOWER falls since it is not designed to capture instances of the signals that are more than $2d$ apart. The VM model allows such signals and the constraints used by WINNOWER need to be relaxed to deal with this situation (to be described elsewhere). We also remark that in our tests of existing algorithms, MEME turned out to be the best in correctly identifying the length of the signal.

Figure 4 shows the performance of SP-STAR on a biased sample and a corrupted sample (FM model). For many biased samples, the performance of SP-STAR, CONSENSUS, GibbsDNA and MEME is about the same, probably due to the fact that the randomly generated signal "stands out" better against the non-uniform background and the problem becomes easier. For cor-

Maximum number of mismatches tolerated
by existing signal finding algorithms

signal length	8	9	10	11	12	13	14	15	16	17	18	19	20
#mismatches	1	1	1	2	2	3	3	3	4	4	5	5	6





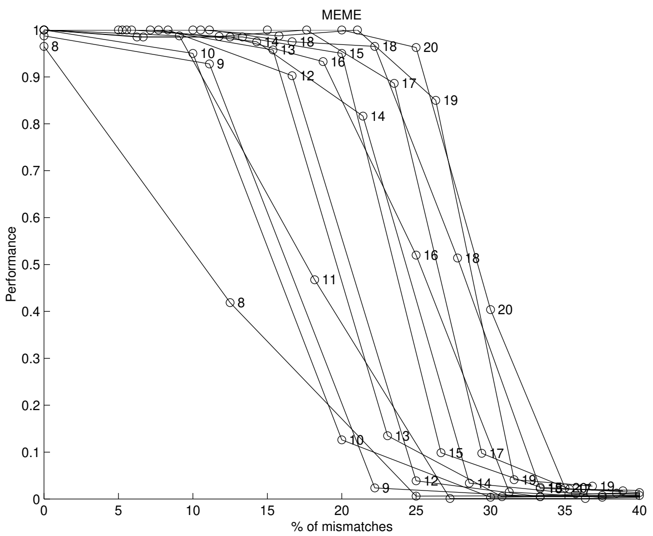


Figure 1: Performance of existing signal finding algorithms. The algorithms used in the tests are CONSENSUS, GibbsDNA (the version of the Gibbs sampler modified for work with DNA sequences obtained from Dr. Michael Zhang, Cold Spring Harbor Laboratory) and MEME. **[Top left]** The entries of the table represent the maximum number of mismatches that are tolerated by the algorithms (average performance coefficient exceeds 50%). For example, CONSENSUS, GibbsDNA and MEME find $(15,3)$-signals but all fail to find $(15,4)$-signals (for samples of 20 sequences, each 600-nt long). The performance of these three algorithms is remarkably consistent, with GibbsDNA slightly better than the other algorithms (CONSENSUS fails for $(13,3)$-signals while MEME fails for $(11,2)$-, $(13,3)$- and $(20,6)$-signals). **[Graphs]** Each algorithm is tested for eight samples of 20 random i.i.d. sequences of length 600 nucleotides with uniform nucleotide composition (FM model). The known signal length l is used as an input parameter to all programs. For CONSENSUS, the first matrix among the list of matrices from the final cycle is used as the output alignment. GibbsDNA is set to disregard fragmentation and the alignment with the best complete log-likelihood ratio over 100 runs is returned. MEME is run in oops mode. For each (l,d) pair, eight different random samples are generated and the performance is taken to be the average over these samples. Numbers on the graphs show the signal length l. The number of mismatches d ranges from 0 to 7. The x-axis represents the percent of mismatches. Data points with the same signal length are connected.

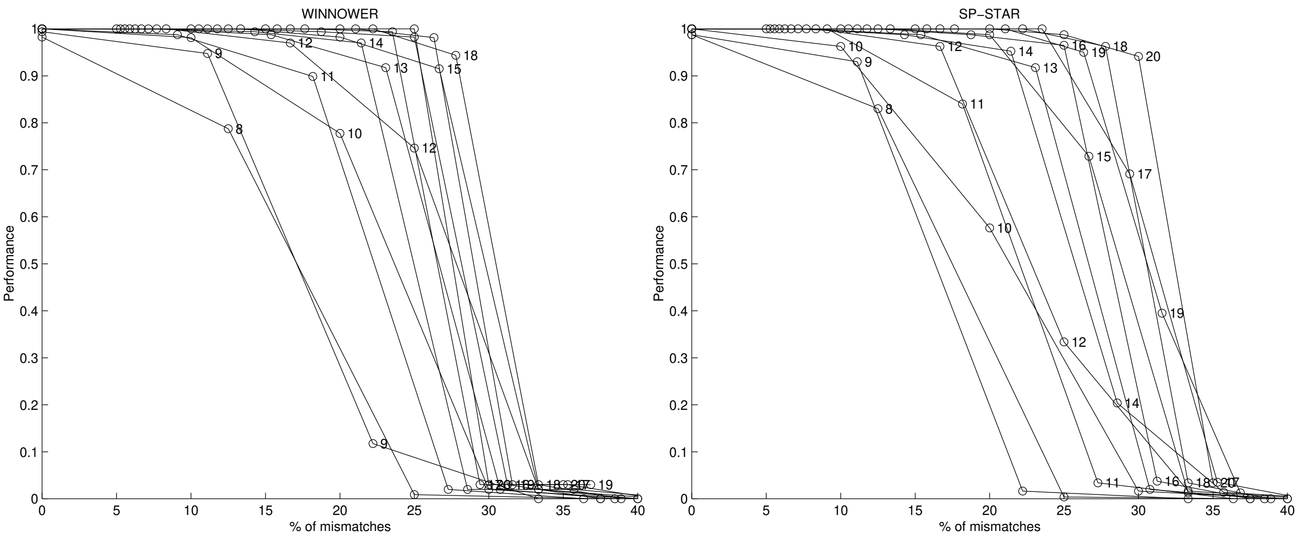


Figure 2: Performance of WINNOWER ($k = 2$) (left) and SP-STAR (right). Samples are constructed in the same way as in Figure 1 (FM model). SP-STAR is run with local improvement on the top 10% initial signals.

rupted samples (12 out of 20 sequences contain a signal), SP-STAR is less accurate in estimating the number of sites when the signal is short, but more accurate with long subtle signals. The performance coefficient drops by 5 to 10% when compared to the non-corrupted samples.

Discussion

Most existing signal finding algorithms are either probabilistic or machine learning implementations of different local search strategies. The shortcoming of these algorithms is that, for subtle signals, they often converge to local optima that represent random patterns rather than a signal. Li, Ma, & Wang (1999) recently proposed a different combinatorial paradigm with a proven performance that avoids local optima. However, the running time of the Li, Ma, & Wang (1999) algorithm is too high for practical implementations. This paper proposes two new combinatorial algorithms for signal finding. The first one, based on the winnowing idea, is able to find rather subtle signals that are difficult to find for existing algorithms. Although WINNOWER shows an excellent performance on simulated samples in FM mode, further work is needed to retain this performance for biological samples and for samples simulated in VM mode. The second algorithm, SP-STAR, is based on a new insight into the design of scoring functions. For future development of these algorithms, we emphasize the following remaining problems (among many): (i) finding multipart signals in SP-STAR; (ii) using more than one best instance of the candidate signals in every sequence while selecting majority strings; (iii) further reducing the WINNOWER's running time and memory requirement for $k > 2$; (iv) taking into account the "edge scores" in WINNOWER; and (v) relaxing filtering conditions to adjust WINNOWER for corrupted samples. To address these problems, we have recently implemented and tested a hybrid algorithm that combines the ideas of WINNOWER and SP-STAR in a single framework (to be described elsewhere).

Acknowledgments

We are grateful to Michael Zhang for providing a version of Gibbs sampler modified for finding signals in DNA sequences. We thank Mikhail Gelfand and Andrey Mironov for many comments on the manuscript and for multiple samples of bacterial regulatory sites that were used in the tests of the algorithms (to be described elsewhere). We are also grateful to Russell Impagliazzo for a discussion on the hidden clique problem.

References

Akutsu, T.; Arimura, H.; and Shimozono, S. 2000. On approximation algorithms for local multiple alignment. In Istrail, S.; Pevzner, P.; and Waterman, M., eds., *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology (RECOMB-00)*, 1–7. Tokyo, Japan: ACM Press.

Alon, N.; Krivelevich, M.; and Sudakov, B. 1998. Finding a large hidden clique in a random graph. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 1998)*, 594–598. SIAM Press.

Bailey, T., and Elkan, C. 1995. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning* 21:51–80.

Brazma, A.; Jonassen, I.; Eidhammer, I.; and Gilbert, D. 1998. Approaches to the automatic discovery of

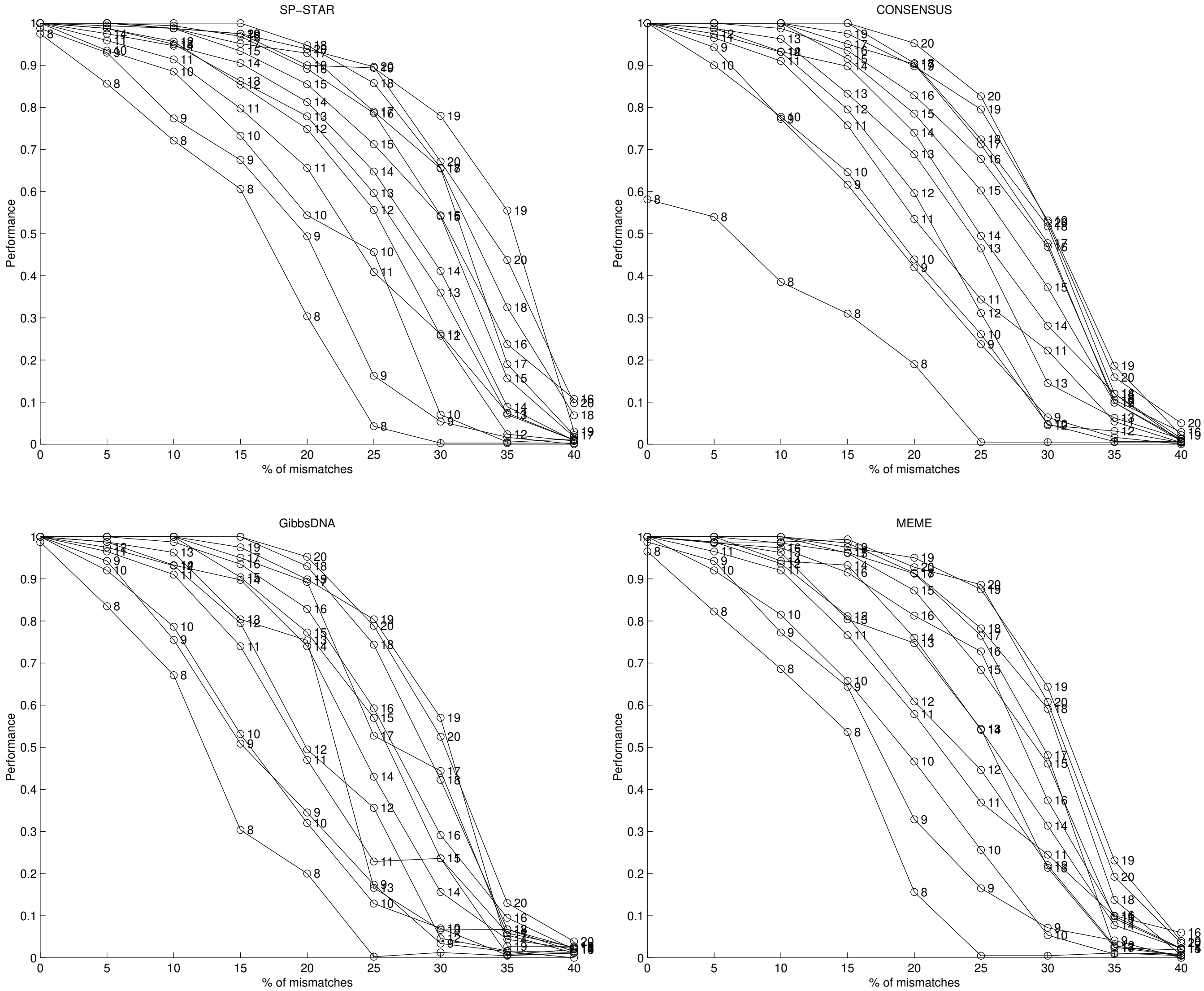


Figure 3: Performance of SP-STAR, CONSENSUS, GibbsDNA and MEME under the VM model. Samples are constructed in the same way as in Figure 1 but in the VM model.

patterns in biosequences. *Journal of Computational Biology* 5:279–305.

Fraenkel, Y.; Mandel, Y.; Friedberg, D.; and Margalit, H. 1995. Identification of common motifs in unaligned DNA sequences: application to *Escherichia coli* Lrp regulon. *Computer Applications in Biosciences* 11(4):379–387.

Garey, M., and Johnson, D. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W.H. Freeman and Co.

Gelfand, M.; Koonin, E.; and Mironov, A. 2000. Prediction of transcription regulatory sites in archaea by a comparative genomic approach. *Nucleic Acids Research* 28(3):695–705.

Hertz, G., and Stormo, G. 1999. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics* 15:563–577.

Jerrum, M. 1992. Large cliques elude the Metropolis process. *Random Structures and Algorithms* 3(4):347–359.

Kelly, T., and Smith, H. 1970. A restriction enzyme from *Hemophilus influenzae. II. Journal of Molecular Biology* 51:393–409.

Lawrence, C.; Altschul, S.; Boguski, M.; Liu, J.; Neuwald, A.; and Wootton, J. 1993. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science* 262:208–214.

Li, M.; Ma, B.; and Wang, L. 1999. Finding similar regions in many strings. In *Proceedings of the 31st ACM*

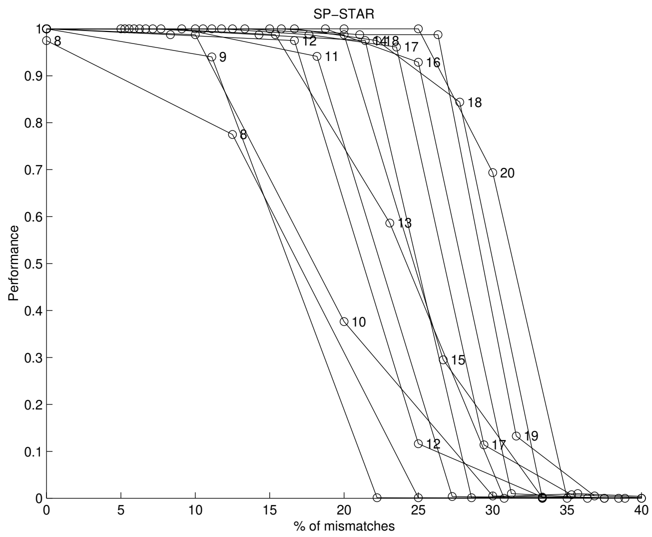


Figure 4: Performance of the extended SP-STAR algorithm (FM model) for samples with nucleotide composition A:C:G:T being 1:1:1:2 (left) and for corrupted samples (uniform nucleotide composition) where only 12 out of 20 sequences contain a signal (right).

Annual Symposium on Theory of Computing, 473–482.

Neuwald, A.; Liu, J.; and Lawrence, C. 1995. Gibbs motif sampling: detection of bacterial outer membrane protein repeats. *Protein Science* 4(8):1618–1632.

Pesole, G.; Prunella, N.; Liuni, S.; Attimonelli, M.; and Saccone, C. 1992. WORDUP: an efficient algorithm for discovering statistically significant patterns in DNA sequences. *Nucleic Acids Research* 20(11):2871–2875.

Rigoutsos, I., and Floratos, A. 1998. Combinatorial pattern discovery in biological sequences. *Bioinformatics* 14:55–67.

Rocke, E., and Tompa, M. 1998. An algorithm for finding novel gapped motifs in DNA sequences. In Istrail, S.; Pevzner, P.; and Waterman, M., eds., *Proceedings of the Second Annual International Conference on Computational Molecular Biology (RECOMB-98)*, 228–233. New York, New York: ACM Press.

Roytberg, M. 1992. A search for common pattern in many sequences. *Computer Applications in Biosciences* 8:57–64.

Staden, R. 1989. Methods for discovering novel motifs in nucleic acid sequences. *Computer Applications in Biosciences* 5(4):293–298.

Stormo, G., and Hartzell III, G. 1989. Identifying protein-binding sites from unaligned DNA fragments. *Proceedings of the National Academy of Sciences USA* 86:1183–1187.

Tompa, M. 1999. An exact method for finding short motifs in sequences with application to the Ribosome Binding Site problem. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, 262–271. Heidelberg, Germany: AAAI Press.

van Helden, J.; Andre, B.; and Collado-Vides, J. 1998. Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *Journal of Molecular Biology* 281(5):827–842.

Vihinen, M. 1988. An algorithm for simultaneous comparison of several sequences. *Computer Applications in Biosciences* 4:89–92.

Vingron, M., and Argos, P. 1991. Motif recognition and alignment for many sequences by comparison of dot-matrices. *Journal of Molecular Biology* 218:33–43.

Vingron, M., and Pevzner, P. 1995. Multiple sequence comparison and consistency on multipartite graphs. *Advances in Applied Mathematics* 16:1–22.

Wolfertstetter, F.; Frech, K.; Herrmann, G.; and Werner, T. 1996. Identification of functional elements in unaligned nucleic acid sequences by a novel tuple search algorithm. *Computer Applications in Biosciences* 12(1):71–80.

Workman, C., and Stormo, G. 2000. ANN-SPEC: a method for discovering transcription factor binding sites with improved specificity. In *Pacific Symposium on Biocomputing*, 464–475.