Finding Motifs Using Random Projections

Jeremy Buhler*
Department of Computer Science
Box 1045
Washington University
One Brookings Drive
St. Louis, MO 63130 USA
jbuhler@cs.wustl.edu
Voice: 314-935-6180
Fax: 314-935-7302

Martin Tompa
Department of Computer Science and Engineering
Box 352350
University of Washington
Seattle, WA 98195-2350 USA
tompa@cs.washington.edu

*corresponding author

Abstract

The DNA motif discovery problem abstracts the task of discovering short, conserved sites in genomic DNA. Pevzner and Sze recently described a precise combinatorial formulation of motif discovery that motivates the following algorithmic challenge: find twenty planted occurrences of a motif of length fifteen in roughly twelve kilobases of genomic sequence, where each occurrence of the motif differs from its consensus in four randomly chosen positions. Such "subtle" motifs, though statistically highly significant, expose a weakness in existing motif finding algorithms, which typically fail to discover them. Pevzner and Sze introduced new algorithms to solve their (15,4)-motif challenge, but these methods do not scale efficiently to more difficult problems in the same family, such as the (14,4)-, (16,5)-, and (18,6)-motif problems.

We introduce a novel motif discovery algorithm, PROJECTION, designed to enhance the performance of existing motif finders using random projections of the input's substrings. Experiments on synthetic data demonstrate that PROJECTION remedies the weakness observed in existing algorithms, typically solving the difficult (14,4)-, (16,5)-, and (18,6)-motif problems. Our algorithm is robust to nonuniform background sequence distributions and scales to larger amounts of sequence than that specified in the original challenge. A probabilistic estimate suggests that related motif-finding problems that PROJECTION fails to solve are in all likelihood inherently intractable. We also test the performance of our algorithm on realistic biological examples, including transcription factor binding sites in eukaryotes and ribosome binding sites in prokaryotes.

1 Introduction

The DNA motif discovery problem abstracts the task of discovering short, conserved sites in genomic DNA sequence. Pevzner and Sze (Pevzner and Sze, 2000) studied a precise combinatorial formulation of this problem that had previously been considered by Sagot (Sagot, 1998). This formulation, the planted motif problem, is of particular interest because it is intractable for commonly used motif finding algorithms.

> **Planted (l, d)-Motif Problem:** Let M be a fixed but unknown nucleotide sequence (the motif consensus) of length l. Suppose that M occurs once in each of t background sequences of common length n, but that each occurrence of M is corrupted by exactly d point substitutions in positions chosen independently at random. Given the t sequences, recover the motif occurrences and the consensus M.

A particular parameterization of this problem, the so-called "challenge problem" of (Pevzner and Sze, 2000), plants a (15,4)-motif (that is, a motif of length $l = 15$ with $d = 4$ substitutions per occurrence) in each of $t = 20$ background sequences of common length $n = 600$ composed of independent random bases with equal frequencies. Such a well-conserved motif should be easy to identify because it is highly unlikely to occur by chance in twenty random sequences of the specified length and composition. The values of n, t, and l in the challenge problem are typical of such motif discovery problems as finding transcription factor binding sites in a collection of coregulated gene promoter regions in yeast.

A number of algorithms to find motifs have been proposed previously, including Bailey and Elkan's MEME (Bailey and Elkan, 1995), Hertz and Stormo's CONSENSUS (Hertz and Stormo, 1999), Lawrence et al.'s Gibbs sampler, (Lawrence et al., 1993), and the algorithms in (Lawrence and Reilly, 1990) and in (Rocke and Tompa, 1998). These algorithms all try to find a motif that maximizes some score, such as a likelihood ratio, designed to distinguish true motifs from chance aggregations of background l-mers. The algorithms employ heuristic methods based on local search, such as Gibbs sampling, expectation maximization, or a greedy approach, to maximize their score functions.

Although local search-based motif finders have seen much success in practice, Pevzner and Sze showed (Pevzner and Sze, 2000) that CONSENSUS, GibbsDNA, and MEME all perform poorly on the (15,4)-motif challenge problem. When presented with an instance of the challenge problem, local search methods usually terminate at a local maximum of their score function corresponding to a randomly occurring pattern in the input, missing the planted motif despite its much higher score. These failures suggest that the performance of conventional motif finders is strongly influenced by the precise distribution of mutations within a motif.

In addition to the local search techniques mentioned above, there are also a number of motif-finding algorithms, based on enumeration of all motifs or all mutation patterns, that are guaranteed to find the highest-scoring motif in the input. See, for example, (Blanchette et al., 2002), (Brāzma et al., 1998), (Galas et al., 1985), (Sagot, 1998), (Sinha and Tompa, 2000), (Staden, 1989), (Tompa, 1999), and (van Helden et al., 1998). Unfortunately, these enumerative algorithms become impractical for motifs as long or with as many mutations as those in the challenge problem. Other motif discovery algorithms have been proposed in (Fraenkel et al., 1995) and (Rigoutsos and Floratos, 1998).

Pevzner and Sze developed two novel algorithms, WINNOWER and SP-STAR, to more reliably solve the (15,4)-motif challenge problem. Briefly, WINNOWER constructs a graph whose vertices correspond to all l-mers present in the t input sequences, with an edge connecting two vertices if and only if the corresponding l-mers differ in at most $2d$ positions and do not both come from the same sequence. WINNOWER then looks for a clique of size t in this graph. The second algorithm, SP-STAR, is a more conventional local search method that starts in turn from each individual l-mer x in the input, chooses the closest match to x from every other input sequence, and uses a sum-of-pairs score and iterative refinement to converge on a good motif.

Although WINNOWER and SP-STAR usually find planted (15,4)-motifs, they are less successful at solving more difficult planted motif problems. For example, the (14,4)-, (16,5)-, and (18,6)-motif problems, all with the same background length and composition as the challenge problem, prove intractable for the new algorithms as well as for existing local search techniques (see Table 1).

In this work, we introduce a new motif-finding algorithm, Projection, that ameliorates the limitations of existing motif finders by using <u>random projections</u> of the input, an approach distinct from all the motif discovery algorithms listed above. The key idea of Projection is to partition the set of all l-mers in the input sequences into buckets, such that some bucket receives several occurrences of the desired motif and little else. Having several copies of the motif in hand greatly enhances the ability of local search techniques to find motifs that would otherwise be missed. To achieve the desired partition, we choose k of the l positions in the unknown motif at random (the value of k to be determined later), then hash every l-mer x into a bucket $f(x)$ determined by its bases at these k positions. A bucket receiving an unusually large number of l-mers has an elevated probability of being enriched for the motif.

Antecedents of our PROJECTION algorithm include projection-based techniques for sparse indexing of databases and for high-dimensional computational geometry. Estimates of similarity based on sparse sampling of positions from feature vectors have long been used in machine vision to match a perceived object against a database of known objects (Duda and Hart, 1973, Chapter 6); in the vision and computational geometry communities, this technique has a distinguished history under the name "geometric hashing" (Wolfson and Rigoutsos, 1997). Analytical proof of the sensitivity of uniform, randomized geometric hashing for finding near neighbors of points in a high-dimensional metric space appears in Indyk and Motwani's work on "locality-sensitive hashing" (Indyk and Motwani, 1998; Gionis et al., 1999), building on theoretical work in random projection by, e.g., Johnson and Lindenstrauss (Johnson and Lindenstrauss, 1984), Bourgain (Bourgain, 1985), and Linial, London, and Rabinovich (Linial et al., 1994).

Rigoutsos and Califano observed that projection can be applied to search problems in bioinformatics, specifically to detect similarity between pairs of biosequences, by treating a fixed-length sequence as a feature vector whose features are individual residues (nucleotides or amino acids). This observation led to their FLASH algorithm for detecting strong pairwise local alignments between biosequences (Rigoutsos and Califano, 1993). Buhler (Buhler, 2001) took a somewhat different approach, applying Indyk and Motwani's locality-sensitive hashing technique to obtain randomized sensitivity guarantees for genomic sequence similarity search independent of the underlying sequence composition. Both of these applications, like the earlier work in vision and geometry, focused on detecting similar pairs of sequences, though the techniques used extend naturally to finding sets of sequences that are all pairwise highly similar, after the fashion of WINNOWER. Linial et al. (Linial et al., 1997) used yet another randomized projection algorithm to cluster proteins.

The PROJECTION motif finder extends previous projection-based search techniques to solve a multiple alignment problem that is not effectively addressed by pairwise alignment. We show that random projection can directly address the problem of finding sets of sequences close to a common consensus without first computing pairwise distances among them. Our approach of picking uniform random projections reduces the extent to which PROJECTION's performance depends sensitively on particular properties of the motif being sought and simplifies some a priori choices, such as projection size and number of projections, that must be made to parameterize the algorithm. The new algorithm both explicitly meets Pevzner and Sze's planted motif challenge and demonstrates that random projection is a useful initialization technique to improve the sensitivity of local search-based motif finders.

We have found that PROJECTION performs better than existing local search or clique-finding motif finders on planted (l, d)-motif problems. In experiments with randomly generated input sequences, PROJECTION typically solved the difficult (14,4)-, (16,5)-, and (18,6)-motif problems. PROJECTION is also efficient, typically solving Pevzner and Sze's planted (15,4)-motif challenge problem in under two minutes on a 667 MHz Alpha workstation. The algorithm is robust to changes in the background base distribution and continues to outperform existing methods as the length n of the input sequences increases.

A probabilistic analysis given in Section 3.2 suggests (again for Pevzner and Sze's parameters $t = 20$ and $n = 600$) that those small values of d for which PROJECTION fails to recover planted (l, d)-motifs are in all likelihood inherently intractable. Specifically, problem instances with these parameters are likely to contain spurious motifs that are as well conserved as the planted motif. For example, twenty random sequences of length 600 (with no planted motif) are expected to contain at least one (9,2)-motif by chance, whereas the expected number of (10,2)-motifs that they contain is approximately 10^{-7}. Similar statements can be made for (11,3)- vs. (12,3)-motifs, (13,4)- vs. (14,4)-motifs, (15,5)- vs. (16,5)-motifs, and (17,6)- vs. (18,6)-motifs. Thus, there is a rather sharp line between those planted motif problems that PROJECTION solves, and those that inherently cannot be solved.

The remainder of this work is organized as follows. Section 2 describes the PROJECTION algorithm, including both the key random projection phase and the local search that benefits from it, and provides some insight into why the algorithm works. Section 3 is devoted to experimental results. In Section 3.1, we test PROJECTION's performance on synthetic instances of the (15,4)-motif challenge and more difficult planted motif problems. Sections 3.3 and 3.4 respectively determine the performance impact of introducing background sequences with nonuniform distributions and longer lengths than in the original challenge problem.

In Sections 3.5 and 3.6, we complement our synthetic results by addressing realistic biological motif finding problems, first identifying transcription factor binding sites in the promoter regions of eukaryotic genes, then tackling the problem of finding ribosome binding sites in prokaryotes. The promoter data sets contain just a few motif occurrences in an equal number of sequences, while the ribosome binding site data sets consist of thousands of nucleotide sequences, only a fraction of which contain the motif. We validate the motifs found by PROJECTION by comparing them to published sites from the literature and, for ribosome binding sites, to their complementary 16S rRNA sequences.

2 The PROJECTION Algorithm

2.1 Why Planted Motif Problems are Difficult

To motivate the development of PROJECTION, we begin with an example to illustrate why local search-based motif finders have difficulty solving planted (l, d)-motif problems. Figure 1 shows two hexamer motifs, **A** and **B**, each consisting of five approximate occurrences of the consensus sequence `CCATAG`. Each occurrence differs from the consensus by exactly two substitutions. In terms of mutations per occurrence, these two motifs are equally well conserved. However, given equal amounts of background sequence, motif **A** is more likely than **B** to be found by motif finders, such as MEME and GibbsDNA, that are based on local search. Figure 1 here

The difference between motifs **A** and **B** is that the mutations in **B**, as specified in the planted (l, d) problem formulation, are distributed uniformly across its positions, while those in **A** are confined to its two center positions. This difference makes **B** harder to find for two reasons. First, despite having the same number of mutations, motif **B** has only slightly more than half the information content of **A** (4.4 versus 8.2 bits). Algorithms that score motifs using statistical measures related to information content will have more trouble separating motif **B** from the background.

A second, more insidious problem is the fact that the average Hamming distance between occurrences of motif **B** is large – 3.6 substitutions versus only 1.6 for motif **A**. Local search methods typically start their search by guessing a single occurrence (alternatively, several independent occurrences) of the motif, then try to find additional occurrences by selecting l-mers similar to the initial guess. Local search is likely to terminate at a local maximum different from the motif if the background contains substantial "noise," i.e. random l-mers that are more similar to the initial guess than are other true occurrences of the motif. A larger average distance between motif occurrences increases the chance that the l-mers most similar to an initially guessed occurrence are not other copies of the motif but random sequences from the background. For example, a DNA hexamer chosen uniformly at random has probability 0.038 of matching a fixed motif occurrence to within two substitutions, but it has probability 0.466 of matching it within four substitutions[1]. Hence, local search methods using typical initialization strategies encounter substantially more noise when finding motif **B** and so are more likely to fail.

We can make local search more robust to a large average distance between motif occurrences by

[1] Using more realistic parameters, a random 15-mer will match a fixed occurrence of a (15,4)-motif to within four substitutions with probability 1.2×10^{-4}, but it has a greater than 5% chance of matching within eight substitutions.

improving the way the search is initialized. Suppose we initially guess $s > 1$ occurrences of motif **B**. For example, suppose $s = 3$, and we somehow guess the first, second, and fourth occurrences of **B** in Figure 1. The consensus of these three strings is `CNATAG`, which differs from the true motif's consensus in only one position and from its occurrences by an average of 2.6 substitutions – substantially less than the average distance between one occurrence and another. Starting with multiple occurrences permits an initial guess that more closely resembles the true motif and so decreases the opportunity for noise to confound subsequent local search.

The main problem with a more robust initialization strategy is its computational cost. We do not know where the motif appears in the input, so to find s occurrences we might naively guess every (multi)set of s l-mers in the input as a starting point for local search, requiring $\binom{t}{s}(n - l + 1)^s$ searches overall. Even for small s, exhaustive enumeration and testing of all such guesses is prohibitively expensive. PROJECTION takes a different approach: it randomly samples multisets of at least s (non-independent) l-mers in a way that is biased toward picking sets of motif occurrences. Specifically, PROJECTION prefers sets of l-mers that are likely to be similar to (i.e. to have few substitutions versus) a common consensus.

2.2 Initialization through Random Projection

The initial goal of PROJECTION is to guess at least s occurrences of an (unknown) planted motif. To this end, the algorithm performs m independent trials, each of which may generate multiple guesses. In each trial, it chooses a random projection f and hashes each l-mer x in the input to a bucket labeled $f(x)$. Any bucket receiving sufficiently many entries is explored as a potential motif, using a local search-based refinement procedure described in the next section.

As outlined in Section 1, the projections f are constructed by choosing k positions uniformly at random without replacement from the set $\{1 \ldots l\}$, for a value of k to be determined later. If x is an l-mer, then $f(x)$ is simply the k-mer that results from concatenating the bases at the selected k positions of x. Viewing x as a point in an l-dimensional Hamming space, $f(x)$ is the projection of x onto a k-dimensional subspace.

Let M be the unknown motif's consensus. Define the planted bucket to be that bucket labeled with hash value $f(M)$. The fundamental intuition underlying random projection is that, if $k < l - d$, there is a good chance that at least s occurrences of M will hash together into the planted bucket. At the same time, if k is not too small, it is unlikely that many random l-mers from the background sequence will hash to the planted bucket, because each such l-mer must agree with M in all k sampled positions. Thus, the l-mers in the planted bucket will likely be highly enriched for the planted motif. Of course, the algorithm does not know which bucket is the planted one and so attempts to recover the motif from every bucket that receives at least s l-mers. (On occasion, PROJECTION actually succeeds in recovering the correct motif by refining some bucket other than the planted bucket, which is an added bonus.)

To fully specify the PROJECTION algorithm, we must describe how to compute both the projection size k and the number of trials m as a function of the algorithm's parameters l, d, n, t, and s and the probability q that the algorithm successfully hashes at least s motif occurrences to the planted bucket. We must also describe how to choose s, which while notionally an input to the algorithm is not a "natural" parameter for motif finding.

We choose the projection size k so as to minimize contamination of the planted bucket by random background sequences. Suppose we require at most E background l-mers per bucket on average. PROJECTION hashes $t(n-l+1)$ l-mers into 4^k buckets, so if we set

$$k \geq \log_4 \left(\frac{t(n-l+1)}{E} \right),$$

then the expected number of background l-mers per bucket is at most E. We normally fix $E < 1$ so that the planted bucket is expected to contain less than one l-mer from the background sequence.

The number of trials m must be set large enough so that with probability at least q, some trial produces a planted bucket containing at least s motif occurrences. The probability q is computed over both the random choices of projections and the random distribution of mutations in the problem instance. We generally set $q = 0.95$, which is high enough that PROJECTION often produces planted buckets with at least s motif occurrences in several trials, providing some robustness against unsuccessful refinements.

Because PROJECTION chooses its projections uniformly at random, each motif occurrence in the planted model hashes to the planted bucket with probability $\hat{p}(l,d,k)$, defined by

$$\hat{p}(l,d,k) = \frac{\binom{l-d}{k}}{\binom{l}{k}}.$$

In particular, those planted occurrences for which the d mutated positions are disjoint from the k hash positions will hash to the planted bucket. Let $\hat{t}$ be an estimate of the number of input sequences containing a planted motif occurrence ($\hat{t} = t$ for our synthetic challenge problems and promoter examples). Then the probability that fewer than s planted occurrences hash to the planted bucket in a given trial is $B_{\hat{t},\hat{p}(l,d,k)}(s)$, where $B_{t,p}(s)$ is the probability that there are fewer than s successes in t independent Bernoulli trials, each trial having probability p of success. We may assume that the trials for different motif occurrences are independent because the problem formulation states that mutations appear independently at random in each occurrence.

If PROJECTION is run for m trials, the probability that s or more motif occurrences hash to the planted bucket in at least one trial is

$$1 - \left[B_{\hat{t},\hat{p}(l,d,k)}(s)\right]^m \geq q.$$

In order to satisfy this inequality, choose

$$m = \left\lceil \frac{\log(1-q)}{\log(B_{\hat{t},\hat{p}(l,d,k)}(s))} \right\rceil. \tag{1}$$

It remains to describe how to choose the bucket size threshold s. We must choose s large enough so that a bucket with at least s true motif occurrences is likely to produce the planted motif after refinement. Unfortunately, we know of no theory that determines a lower bound for s from this criterion, though we have found empirically that in experiments on problems containing 4–20 motif occurrences in background sequences of 600–1000 bases, setting $s = 3$ or 4 usually works well. An alternative lower bound for s follows from efficiency considerations: we wish to discard buckets composed entirely of background l-mers, which are unlikely to be useful as starting points for refinement. We assume for simplicity that in a collection of random background sequences, the number of l-mers x that project to a common hash value y is approximately Poisson-distributed with a mean μ_y given by

$$\mu_y = t(n-l+1) \Pr_x [f(x) = y]$$

where the latter probability is computed over the background distribution of l-mers x. The Poisson assumption derives from the fact that we are counting occurrences of a short fixed pattern, namely the bases of y appearing in the positions read by f, in the background sequence. We set the threshold size s for each bucket to the larger of its empirically derived value and the 90th percentile value of the bucket's size distribution.

In the case of nonuniform background base frequencies, the probability $Pr[f(x) = y]$ may be different for each bucket, producing larger thresholds for buckets likely to contain many background

l-mers purely by chance. For a uniform background, this probability is always $1/4^k$. In the experiments performed in this work, the empirical value of s is larger than the lower bounds implied by the 90th percentile size for most buckets, so we use the empirical s to calculate m.

Using the above criteria for m, k, and s, finding motifs in problems of the sizes considered here requires at most thousands of trials, and usually many fewer, to produce a bucket containing enough motif occurrences for effective refinement.

2.3 Motif Refinement and Scoring

PROJECTION refines each sufficiently large bucket in hopes of recovering the planted motif. If the bucket being refined is the planted bucket, its l-mers already match the motif's consensus in at least k positions. These positions plus the information in the remaining $l - k$ positions of the bucket's l-mers provide a strong signal, starting from which a few iterations of refinement should lead to the correct motif.

It is important to note that PROJECTION is primarily an initialization strategy that produces starting points for refinement. We describe one particular refinement method below, but PROJECTION could instead be adjoined to the local search phases of existing motif finders like MEME or GibbsDNA. The refinement phase, whatever algorithm it uses, is ultimately responsible for producing the most significant motif from the starting points offered and discarding all other candidates. Hence, we need not be concerned (except for reasons of efficiency) by the fact that most large buckets passed to refinement are "false positives" that do not lead to a significant motif.

Our primary tool for refining candidate motifs is expectation maximization (EM), as formulated for the motif finding problem by Lawrence and Reilly (Lawrence and Reilly, 1990). This EM formulation derives from the following simplified probabilistic model. An occurrence of the motif appears exactly once in each input sequence. Motif occurrences are generated at random from a $4 \times l$ weight matrix model W whose (i, j)th entry gives the probability that base i appears in position j of an occurrence, independent of its other positions. The remaining $n - l$ residues in each sequence are chosen independently at random according to a background base distribution P. Although this model only approximates the motifs of real biosequences, it is both simple and sensitive enough to let EM identify meaningful motifs in practice. Bailey and Elkan (Bailey and Elkan, 1995) give more accurate motif models, though fitting their parameters from sequence data requires significantly more computation.

Let T be a set of t input sequences, and let P be the background distribution. EM-based refinement seeks a weight matrix model W^* that maximizes the likelihood ratio

$$LR(W^*) = \frac{\Pr(T \mid W^*, P)}{\Pr(T \mid P)},$$

that is, a motif model that explains the observed sequences much better than the background model alone. The position at which the motif occurs in each sequence is not fixed a priori, making computation of W^* difficult because $\Pr(T \mid W^*, P)$ must be summed over all possible positions for the occurrences. To address this difficulty, the core EM algorithm (Dempster et al., 1977) specifies an iterative calculation that, given an initial guess W_1 at the motif model, converges linearly to a locally maximum-likelihood model in the neighborhood of W_1.

PROJECTION performs EM refinement on every bucket with at least s l-mers. It forms an initial guess W_b from a bucket b as follows: set $W_b(i, j)$ to be the frequency of base i among the jth positions of all l-mers in b. This guess forms a centroid for b, in the sense that positions that are well conserved in b are strongly biased in W_b, while poorly conserved positions are less biased. In order to avoid zero entries in W_b, we add a Laplace correction of p_i to $W_b(i, j)$, where p_i is the probability of base i in the background sequence.

Because EM converges only linearly, running it to convergence for every W_b would be computationally prohibitive. Fortunately, just a few iterations of EM (five in our implementation) can significantly improve a well-chosen starting model to the point where it identifies the planted motif. Let W_b^* be the candidate motif model refined from W_b. We form a guess at the planted motif by selecting from each input sequence the l-mer x with the largest likelihood ratio $\Pr(x \mid W_b^*)/\Pr(x \mid P)$. This multiset S_b of l-mers represents the motif in the input that is most consistent with the model W_b^*.

PROJECTION generates refined guesses for every sufficiently large bucket and for every trial, but we wish to pick a single best motif to report to the user. For the biological examples of Section 3.5, we score each refined motif S_b according to its likelihood as follows. Let W_{S_b} be the weight matrix model inferred from S_b (in the same way that we form initial guesses). Then the likelihood ratio score $LR(S_b)$ is defined as

$$LR(S_b) = \prod_{x \in S_b} \frac{\Pr[x \mid W_{S_b}]}{\Pr[x \mid P]}$$

We report the motif S^* that maximizes this score over all buckets and all m trials.

To maximize the number of motif occurrences recovered in the synthetic challenge problems of Section 3.1, we perform a further combinatorial refinement of each S_b. This further refinement process is similar to SP-STAR (Pevzner and Sze, 2000) but uses a different score function. Compute the consensus M_b of the sequences in S_b, and define the score $\sigma(S_b)$ to be the number of sequences in S_b whose Hamming distance to S_b is at most d. Let S'_b contain the l-mer from each input sequence that is closest in Hamming distance to M_b. If $\sigma(S'_b) > \sigma(S_b)$, replace S_b by S'_b and repeat. This refinement usually converges in a few iterations. Again, we report the motif S^* for which $\sigma(S^*)$ is maximum over all buckets and all m trials.

3 Experimental Results

3.1 Challenge Problems on Synthetic Data

We first tested PROJECTION on synthetic problem instances generated according to the planted (l, d)-motif model. Pevzner and Sze (Pevzner and Sze, 2000) showed such problems to be intractable to most existing motif finders even for $l = 15$ and $d = 4$, so they are natural test cases for our algorithm. We produce problem instances as follows: first, a motif consensus M of length l is chosen by picking l bases at random. Second, $t = 20$ occurrences of the motif are created by randomly choosing d positions per occurrence (without replacement) and mutating the base at each chosen position to a different, randomly chosen base. Third, we construct t background sequences of length $n = 600$ using $n \cdot t$ bases chosen at random. Finally, we assign each motif occurrence to a random position in a background sequence, one occurrence per sequence. All random choices are made uniformly and independently with equal base frequencies. This generation procedure corresponds to the "FM" model used in the challenge problem described by Pevzner and Sze.

We report the performance of PROJECTION using the performance coefficient of (Pevzner and Sze, 2000), defined as follows. Let K denote the set of $t \cdot l$ base positions in the t occurrences of a planted motif, and let P denote the corresponding set of base positions in the t occurrences predicted by an algorithm. Then the algorithm's performance coefficient on the motif is defined to be $|K \cap P|/|K \cup P|$. When all occurrences of the motif are found correctly, the performance coefficient achieves its maximum value of one.

Table 1 compares the performance of PROJECTION with that of previous motif discovery algorithms on sets of 100 random problem instances, each generated as described above. All experiments used projection size $k = 7$ and bucket size threshold $s = 4$, which combined with the problem parameters requires numbers of trials m as shown in the table. The values of m are determined by Equation (1). For each set of problem parameters, we give the average performance coefficient for PROJECTION as well as the number of problem instances (out of 100) for which it correctly recovered the planted motif's consensus. For comparison, we provide corresponding average performance coefficients for three other algorithms: GibbsDNA, WINNOWER ($k = 2$), and SP-STAR. The data for previous algorithms was collected by Pevzner and Sze and summarized in (Pevzner and Sze, 2000, Figures 1 and 2). Table 1

In every line of Table 1, the average performance coefficient of PROJECTION is at least as great as that of any of the previous algorithms. PROJECTION correctly solved planted (11,2)-, (13,3)-, here

(15,4)-, (17,5)-, and (19,6)-motif problems at least 98 times out of 100; in these cases, average performance coefficients less than one occurred primarily because our algorithm, like any motif finder, sometimes picked as a motif occurrence a background l-mer that was at least as similar to the correct consensus as was the true occurrence. The (11,2)-, (13,3)-, and (15,4)-motif problems were roughly equally accessible to WINNOWER, somewhat less so to SP-STAR.

PROJECTION's improved performance is more striking on the more difficult planted (14,4)-, (16,5)-, (17,5)-, (18,6)-, and (19,6)-motif problems. Our algorithm's performance on these problems substantially exceeds that of previous algorithms, including those of Pevzner and Sze, which typically fail to find the planted motifs. Finding each synthetic motif in the most difficult (18,6) problem required about one hour on a 667 MHz Alpha workstation; easier problems like (15,4) were typically solved in only a few minutes.

3.2 Limitations on Solvable (l,d)-Motif Problems

Although PROJECTION performs well on the planted motifs of Table 1, it generally fails to find motifs with slightly different parameters, such as (9,2)-, (11,3)-, (13,4)-, (15,5)-, or (17,6)-motifs (again for $t = 20$ and $n = 600$). We naturally investigated why our algorithm tends to fail on problems that seem quite similar to the original challenge.

A probabilistic analysis suggests that problems involving planted motifs with the parameters given above are quantitatively different from the problems in Table 1. For example, twenty random sequences of length 600, with no planted motif, are expected to contain more than one spurious (9,2)-motif by chance, whereas the expected number of (10,2)-motifs that they contain is approximately 6.1×10^{-8}. We derive these estimates as follows. Let

$$p_d = \sum_{i=0}^{d} \binom{l}{i} \left(\frac{3}{4}\right)^i \left(\frac{1}{4}\right)^{l-i}$$

be the probability that a fixed l-mer occurs with up to d substitutions at a given position of a random sequence. Then the expected number of length-l motifs that occur with up to d substitutions at least once in each of t random length-n sequences is approximately

$$E(l,d) = 4^l \left(1 - (1 - p_d)^{n-l+1}\right)^t.$$

This expectation is only an estimate because overlapping occurrences of a given consensus string M do not occur independently in the background.

Table 2 lists relevant values of $E(l, d)$ and $E(l+1, d)$ for comparison. In each line of the table, the expected number of spurious (l, d)-motifs is around 1–5, whereas the expected number of spurious $(l+1, d)$-motifs is negligible. We therefore expect that on the specified (l, d)-problems, PROJECTION, or for that matter any other algorithm, is likely to report a spurious motif as good as the planted motif, even if it usually succeeds on the corresponding $(l+1, d)$-motif problems. Table 2 here

The values $E(l, d)$ in Table 2 are only estimates of the expectations, which we do not know how to compute precisely. However, we do know from an exhaustive enumeration of 9-mers and an exact calculation of their probabilities in twenty random 600-mers that the expected number of spurious (9,2)-motifs is 1.621. (The probability calculation was done using an algorithm described in (Tompa, 1999, Section 3.1).) Thus, the estimates may not be too inaccurate in practice. Such an exhaustive analysis for much greater values of l is unfortunately computationally impractical.

To further corroborate our analysis, we ran PROJECTION on sets of twenty random instances of the planted (l, d)-motif problems of Table 2 generated as described in the previous section. The algorithm's performance on these sets is also reported in Table 2, including the average performance coefficient, the number of problem instances in which the correct consensus was found, and the number of instances where we instead found a spurious (l, d)-motif appearing in all twenty input sequences. Where PROJECTION failed to find either the correct or an equally good spurious motif, it found a motif (again not the planted one) occurring in nineteen of the twenty input sequences. These experiments provide further evidence that the (l, d)-motif problems that PROJECTION consistently fails to solve are fundamentally less tractable because they contain spurious motifs that are as well conserved as, and hence cannot be distinguished from, the planted motif.

3.3 Performance vs. Background Base Distribution

Real biosequences frequently have base compositions different from the equal base frequencies used in the experiments of Section 3.1. We therefore tested the performance of PROJECTION on synthetic motif-finding problems with background sequences of varying composition. These problems were constructed and solved identically to those of Section 3.1, except that we chose a background $G + C$ fraction θ_{GC}, then generated the background sequence from a distribution with $\Pr[G] = \Pr[C] = \theta_{GC}/2$ and $\Pr[A] = \Pr[T] = (1 - \theta_{GC})/2$. We continued to generate planted motifs from a distribution with equal base frequencies.

Figure 2 shows the performance of PROJECTION on both (15,4)- and more challenging (14,4)-motifs at different background compositions. The algorithm's performance, as measured by the average performance coefficient, was not significantly changed on (15,4)-motifs down to 35% $G+C$ and on (14,4)-motifs down to 40% $G+C$. Below these thresholds, recovery of the motif drops off precipitously, with average performance coefficients below 0.1 occurring at 35% $G+C$ for (14,4)-motifs and 25% $G+C$ for (15,4)-motifs.

The major performance drops for highly biased $G+C$ contents occur because a biased background is more likely to produce problem instances that, like those of Section 3.2, by chance contain spurious motifs at least as good as the planted motif. In support of this claim, we observe that in every problem instance with 35% or lower $G+C$ where the algorithm failed to find the planted motif, it found a collection of twenty l-mers that were at least as well conserved. For (14,4)-motifs in particular, this behavior is radically different from the normal failure mode at 45% and 50% $G+C$: of the roughly 15% of trials that failed at these $G+C$ contents, none found a spurious motif as well-conserved as the planted one. This observed shift from finding suboptimal motifs to finding spurious but optimal motifs provides strong empirical evidence for the high frequency of spurious motifs at more biased $G+C$ contents.

We conclude that the performance of the PROJECTION algorithm on sequences of biased $G+C$ content is limited primarily by the appearance of spurious motifs as good as the planted motif. In the absence of such spurious motifs, PROJECTION's ability to find the planted motif is robust to moderate changes in $G+C$.

Figure 2 here

3.4 Performance vs. Background Sequence Length

We tested PROJECTION's ability to handle increasingly noisier problems by finding planted motifs in increasingly large amounts of background sequence. Longer backgrounds contain more random l-mers similar to real motif occurrences, as well as more collections of l-mers almost as well-conserved as the true motif. Both phenomena increase the chance that local search will terminate at a suboptimal local maximum instead of finding the true motif.

Figure 3 shows the performance of PROJECTION on (15,4)- and (14,4)-motifs for background lengths ranging from $n = 600$ to $n = 2000$. Other than the increased length n, problem instances were generated identically to those of Section 3.1. All experiments used the parameters given in Table 1 for (15,4)- and (14,4)-motifs. In these experiments, we retained the projection size $k = 7$ at all lengths, even though for backgrounds longer than 800 bases, setting $k = 7$ causes the average bucket size to exceed one l-mer. Setting $k = 8$ would have kept the average bucket size below one but would have required an order of magnitude more iterations ($m = 1987$ for (15,4) and

$m = 14860$ for (14,4)) to maintain a 95% probability of producing a planted bucket with at least $s = 4$ occurrences of the motif. Figure 3

PROJECTION's performance on (15,4)-motifs degraded gracefully with increasing length, from here an average performance coefficient of 0.93 for $n = 600$ to 0.53 at $n = 2000$. As predicted, some of the decay could be attributed to failures to find the planted motif at all (in 32 of 100 problem instances for $n = 2000$), while the rest was attributable to admixture of background l-mers into otherwise correct motifs. The complete failures were consistent with a proliferation of suboptimal local maxima. For example, at $n = 2000$ the failed problem instances all yielded spurious (15,4)-motifs with sixteen to eighteen occurrences, which by the estimates of Section 3.2 are expected to occur frequently by chance at this background size.

Finding (14,4)-motifs proved much harder at increased sequence lengths. The algorithm's ability to find the planted motif degraded more rapidly, from an average performance coefficient of 0.72 at $n = 600$ to 0.17 at $n = 1400$. Again, the frequent appearance of spurious motifs with sixteen or seventeen occurrences arose concurrently with the failure of local search. However, as the background length increased to 1000 bases and beyond, an increasing fraction of failures were again attributable to the presence of spurious (14,4)-motifs with a full twenty occurrences. By $n = 1400$, such spurious motifs accounted for nearly half of all observed failures, compared to none at $n = 800$. Again, this behavior provides empirical evidence that the performance of PROJECTION is in part limited by the increasing frequency of spurious motifs in longer backgrounds.

Increased length testing is useful not only to determine the absolute performance of PROJECTION but also to compare its performance to that of previous algorithms, including the specialized methods of Pevzner and Sze. Previous motif finders usually fail to find (15,4)-motifs when $n = 1000$, exhibiting a performance coefficient of 0.23 or less (Pevzner and Sze, 2000, Table 1). Only WINNOWER, a non-local-search-based motif finder, maintains an average performance coefficient of at least 0.8 at this background size, and that only with parameter $k = 3$. PROJECTION's comparatively high performance, even for $n = 2000$, again demonstrates the power of augmenting ordinary local search with intelligent initialization.

3.5 Transcription Factor Binding Sites

To test PROJECTION on realistic biological data, we used it to find known transcriptional regulatory elements upstream of eukaryotic genes. We examined orthologous sequences from a variety of organisms taken from regions upstream of four types of gene: preproinsulin, dihydrofolate reductase (DHFR), metallothioneins, and c-fos[2]. These sequences are known to contain binding sites for specific transcription factors. We also tested a collection of promoter regions[3] from the yeast S. cerevisiae that are known to contain a common cell-cycle-dependent promoter, the ECB element (McInerny et al., 1997).

The motifs in these data sets are much better conserved than those in our synthetic problem instances, with little variation and a structure more like the simple motif **A** of Figure 1 than like the more subtle **B**. In general, we have been unable to locate published examples of biological motifs as subtle as those of Section 3.1, but the dearth of such examples need not imply that subtle motifs do not exist biologically. Motifs like those in the planted (l, d) model are inaccessible to existing computational search techniques, and a high degree of conservation is necessary to detect a motif at all given only four or five occurrences. For example, a (15,4)-motif occurring only five times with uniformly distributed mutations would be statistically meaningless in a background like those of our promoter data sets. Many published motifs were likely inferred using few enough sequences that a subtle (l, d)-motif would have been impossible to detect.

In all experiments, we set $l = 20$ and $d = 2$ to reflect the approximate lengths and degrees of conservation of published motifs. Following the rule of Section 2.2 that we should choose k large enough to achieve less than one expected background l-mer per bucket, we chose a uniform projection size $k = 7$ for all experiments. Because the inputs contained only 4–5 sequences, we set a smaller than usual bucket size threshold $s = 3$ so as not to demand that all motif occurrences in the input end up in one bucket. We note that the parameters that generated the results reported here were chosen *a priori* without prior experimentation on these data sets; subsequent testing with slightly different parameters suggests that our results are robust to small changes in l and k. Given the high expected amount of conservation and the low size threshold, the numbers of iterations m computed from Equation (1) proved quite small, requiring only a few seconds' running time. Motifs were scored by likelihood ratio score as described in Section 2.3.

Table 3 gives for each experiment the consensus strings of the highest-scoring motifs found by PROJECTION, along with published motifs that closely match substrings of these consensuses. For experiments listing multiple motifs, the first motif listed is the one of highest score. The locations of motif occurrences were not known a priori, so we do not give performance coefficients. Analysis

[2] Sequences were kindly provided by M. Blanchette; see (Blanchette, 2001) for a list of organisms used and an alternative approach to finding motifs in these sequences.

[3] Genes used: SWI4, CLN3, CDC6, CDC46, and CDC47.

of the preproinsulin promoter region yielded a motif known from the TRANSFAC database (Wingender et al., 1996), while the other four experiments all produced motifs corresponding to experimentally verified transcription factor binding sites (see Table 3). Table 3 here

In cases where a data set contained several distinct known binding sites, we attempted to find additional motifs beyond that of highest score. To find multiple motifs, we masked the motif of highest score, replacing each base in its occurrences with `X`, then reran the motif finder on the masked sequences. This procedure proved effective in finding the additional documented motifs listed in Table 3. In the preproinsulin data, the second motif found contained the well-known CT-II promoter element (Boam et al., 1990), while the second and third motifs from the metallothionein data respectively contained the MREd and MREf promoter elements of (Andersen et al., 1987).

In addition to the promoter sequences listed in Table 3, we ran PROJECTION on a set of twenty 1000-base *C. elegans* promoter regions containing the "X box" motif `RYYNYYATRRNRAC`, the target site for the DAF-19 transcription factor (Swoboda et al., 2000). The genes from which these sequences are taken were chosen by P. Swoboda (personal communication) because their expression is likely regulated by DAF-19. Some genes exhibit empirical evidence of such regulation, while the remainder were chosen because they exhibit an occurrence of the X box motif between 50 and 300 bases upstream of the translation start site.

The X box looks somewhat more like the subtle motifs for which PROJECTION was designed. Only four of the fourteen positions in the motif are perfectly conserved across all twenty occurrences; of the remaining positions, one is poorly conserved, while the rest exhibit a strong preference for either purine or pyrimidine, with one base appearing in 13–19 occurrences. PROJECTION easily found nineteen of twenty known motif occurrences (performance coefficient 0.90) using parameters $l = 14$, $d = 2$, $k = 8$, $m = 6$, and $s = 4$; again, further experimentation suggested that the algorithm's performance was not highly sensitive to the exact parameters used. The twentieth annotated occurrence was not found, but the l-mer reported by PROJECTION had a higher likelihood ratio than the annotated occurrence and, based on its position in the sequence, could conceivably be a second occurrence of the X box site.

Although the motifs we found are not particularly subtle and indeed have previously been found by existing methods (Blanchette, 2001), the results of these experiments are noteworthy for two reasons. First, we achieved good performance even with a fairly primitive refinement strategy that did not include, e.g., score corrections for motif length or iteration of EM to convergence. We expect that random projection would yield even better performance if adjoined to a more sophisticated local search procedure. Second, because PROJECTION selectively samples good starting points for local search, it uses fewer restarts than the usual approach of starting from each l-mer in the input in turn. As shown in Table 3, the number of starting points in the various experiments ranged from 22% to 75% of the input sequence length – substantially fewer than the number of starts required with the usual search initialization. For the X box, the number of starts was 2832, just 14% of the input length. Moreover, the reported motifs were invariably found during several different iterations, so we could have been even more aggressive in reducing m and therefore the number of starts. Future work should determine how aggressively m can be reduced for "easy" motifs like those of this section without sacrificing sensitivity.

3.6 Ribosome Binding Sites

To test PROJECTION's robustness on a very different sort of biological example, we applied it to the problem of finding prokaryote ribosome binding sites. A ribosome binding site problem instance consists of thousands of short DNA sequences ($n = 20$) taken from just upstream of the translation start site of each of an organism's genes. The goal is to identify the site ($l \approx 6$) at which the 16S rRNA of the ribosome binds to mRNAs transcribed from the genes. It is known that this binding site is approximately complementary to a short sequence near the $3'$ end of the 16S rRNA (Kozak, 1983).

The ribosome binding site problem poses challenges to PROJECTION not encountered in previous sections. First, because of incorrect gene annotation and other limitations, only a fraction of the sequences in any problem instance actually contain a ribosome binding site. To model this phenomenon, we set $\hat{t} = t/3$ in Equation (1) when determining the number of iterations to perform. Second, the total amount of sequence in this problem is sufficiently large that we cannot choose k to simultaneously satisfy $k < l - d$ and achieve a contamination threshold of fewer than tens or hundreds of background l-mers. Instead, we set $k = l - d - 1$, as large as possible, and set the bucket size threshold s to twice the average bucket size $t(n - l + 1)/4^k$. This bound should on average select buckets in which motif occurrences (which are numerous in these examples) outnumber background l-mers, that is, buckets with more signal than noise. Because prokaryote genomes often have highly biased composition, some buckets may still be much larger than the threshold s, but these buckets are discarded by the Poisson filtering heuristic described in Section 2.2.

For all ribosome binding site experiments, we chose $l = 6$, $d = 1$, and projection size $k = 4$. Table 4 shows the problem sizes t, thresholds s, and numbers of iterations m for each experiment.

Again, these values were chosen without prior experimentation on the data set. The motif predicted by Projection is shown in the column labeled "Motif." Each experiment finished in under three minutes on an 800 MHz Intel Pentium III workstation. Table 4 here

Although random projection continued to perform well in choosing appropriate starting points for refinement, the very different features of the ribosome binding site problems compared to the other motif finding examples described here exposed the limitations of our simple refinement procedure, forcing us to make a somewhat *ad hoc* modification to it. In particular, likelihood ratio scoring favored motifs with unusual nucleotide composition (e.g. `TCAGGA` for *E. coli*), even if they were relatively infrequent in the input, while combinatorial refinement as described in the last paragraph of Section 2.3 chose very common strings without regard for the unusualness of their composition (e.g. `TAAAAT` for *T. maritima*). Each of these problems with refinement afflicted roughly half the examples tested; in each case, the known ribosome binding site motif was found but no longer received the highest score and so was not reported. In an effort to compromise between the importance of high frequency and meaningful composition, we ultimately altered our refinement strategy by choosing the motifs in Table 4 using σ scoring but without performing combinatorial refinement after expectation maximization.

We believe that the difficulties we encountered in refining candidate motifs in ribosome binding site problems stems from the fact that our motif model does not properly account for sequences lacking an occurrence of the motif. Scoring by likelihood is a common and well-founded technique, but by choosing an occurrence from *every* input sequence, we include a large number of background l-mers that corrupt the consensus reported for the motif. Our solution to this problem, though effective, remains *ad hoc*. A future version of Projection should instead incorporate refinement based on a probabilistic motif model that accounts for sequences with no motif occurrence, in particular the ZOOPS model (Bailey and Elkan, 1995) used by MEME.

Many pieces of evidence corroborate the ribosome binding site motifs predicted by Projection. The first is the complementarity of these motifs to the $3'$ end of the 16S rRNA sequences (with the possible exception of *H. influenzae*), as shown in Table 4. More corroboration follows from the well-known fact that in many bacteria, the binding site for the 16S rRNA during translation initiation is the Shine-Dalgarno sequence `AAGGAGG` or a large substring of it (Kozak, 1983; Lewin, 1997). The reported motifs for the four bacteria in Table 4 agree quite well with this sequence. In archaea such as *M. jannaschii*, the $3'$ end of the 16S rRNA is missing a few terminal nucleotides compared to the bacterial rRNA sequences, and the 16S rRNA binding site is instead `AGGTGAT` or a large substring of it (Woese, personal communication). Hayes and Borodovsky (Hayes and Borodovsky, 1998) discovered the motif `GGTGA` in M. jannaschii using a Gibbs sampler, and Tompa (Tompa, 1999) discovered similar binding sites in both this and three other archaeal genomes.

Tompa used a very different enumerative statistical algorithm to solve the ribosome binding site problem, ranking motifs by their z-scores. All the motifs found by PROJECTION are in good agreement with the highest-scoring motifs that his algorithm reported. For example, the last column of Table 4 shows for each problem instance the pentamer motif, allowing no substitutions, with highest z-score. Note the strong overlap between each of these 5-mers and the corresponding PROJECTION prediction.

Randomization is not strictly necessary to find good starting points for refinement in the ribosome binding site problem. There are only fifteen different projections of a hexamer into four dimensions, so one could efficiently test all possible projections rather than picking them at random. Indeed, because the embedded motifs are so short, this particular problem has been addressed enumeratively without resorting to iterative search techniques at all (Tompa, 1999). The significance of our ribosome binding site results is rather to show that PROJECTION is capable of solving motif-finding problems that are quite different both from the typical applications of Section 3.5 and from the formal motif model for which it was designed.

4 Conclusions and Future Extensions

We have described PROJECTION, a new algorithm for finding motifs based on random projection. PROJECTION was designed to efficiently solve problems from the planted (l, d)-motif model, which it does more reliably and for substantially more difficult instances than previous motif finding algorithms. PROJECTION is robust to changes in background sequence composition and, to some extent, to long background sequences that create noisier motif finding problems. For $t = 20$ and $n = 600$, our algorithm achieves performance close to the best possible, being limited primarily by the statistical considerations of Section 3.2.

Despite its development in a particular formal model, PROJECTION performs well on real biological motif finding problems, even cases as dissimilar from the model as the ribosome binding site problem. As a general sampling technique for initializing local search, our method can extend a variety of existing motif finding algorithms, both increasing their sensitivity to difficult motifs and reducing the number of searches required to find easier motifs. Even so, we continue to seek biological motifs that are more subtle than those described in Sections 3.5 and 3.6.

We intend to improve our implementation of PROJECTION to incorporate additional features common to practical motif-finding algorithms. Basic improvements include a complexity correction that would allow predicting the length of the motif, as well as extending EM refinement to handle sequences with multiple motif occurrences or, as discussed in Section 3.6, sequences with no occurrence at all. Moreover, while EM uses a probabilistic motif model, the analysis that parameterizes PROJECTION is based on a simpler consensus model. Extending our analysis to more general motif models, besides being of theoretical interest, might enable more intelligent parameter choices for easy motifs like those of Section 3.5, allowing us to reduce the number of iterations performed.

A major open question is how to extend PROJECTION to find motifs whose occurrences contain insertions and deletions with respect to the consensus as well as substitutions. A general extension seems extremely difficult, both because random projection depends on sampling corresponding positions from each l-mer in the input and because the probabilistic model used by refinement only permits substitutions. The latter problem is characteristic not only of our method but also of many other popular motif finders. A simpler extension that has proven more tractable in practice (Cardon and Stormo, 1992; Sinha and Tompa, 2000; Marsan and Sagot, 2000) is to handle motifs with one or a few variable-length spacers. The dimeric structure of many transcription factors suggests that motifs with one central spacer, as occur in e.g. S. cerevisiae and E. coli, are a biologically common case worth addressing in our algorithm.

Acknowledgments

We thank Pavel Pevzner and Sing-Hoi Sze for providing the performance coefficients given in Table 1 for their algorithms and for the Gibbs sampler. Thanks are also due to Mathieu Blanchette for collecting the orthologous promoter sequences used in Section 3.5, to Peter Swoboda for information on the X box motif, and to our anonymous referees for suggesting enhancements and clarifications to the paper. This work was supported in part by a Fannie and John Hertz Foundation Fellowship and in part by the National Science Foundation under grant DBI-9974498.

References

Andersen, R. D., Taplitz, S. J., Wong, S., Bristol, G., Larkin, B., and Herschman, H. R. 1987. Metal-dependent binding of a factor in vivo to the metal-responsive elements of the metallothionein 1 gene promoter. Molecular and Cellular Biology 7, 3574–81.

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

Blanchette, M. 2001. Algorithms for phylogenetic footprinting. In RECOMB01: Proceedings of the Fifth Annual International Conference on Computational Molecular Biology, 49–58. Montreal, Canada.

Blanchette, M., Schwikowski, B., and Tompa, M. 2002. Algorithms for phylogenetic footprinting. To appear in Journal of Computational Biology.

Boam, D. S. W., Clark, A. R., and Docherty, K. 1990. Positive and negative regulation of the human insulin gene by multiple trans-acting factors. Journal of Biological Chemistry 265, 8285–96.

Bourgain, J. 1985. On Lipschitz embedding of finite metric spaces in Hilbert space. Israeli Journal of Mathematics 52, 46–52.

Brāzma, A., Jonassen, I., Vilo, J., and Ukkonen, E. 1998. Predicting gene regulatory elements in silico on a genomic scale. Genome Research 15, 1202–15.

Buhler, J. 2001. Efficient large-scale sequence comparison by locality-sensitive hashing. Bioinformatics 17, 419–28.

Cardon, L. R. and Stormo, G. D. 1992. Expectation maximization algorithm for identifying protein-binding sites with variable lengths from unaligned DNA fragments. Journal of Molecular Biology 223, 159–70.

Dempster, A. P., Laird, N. M., and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society 39, 1–38.

Duda, R. O. and Hart, P. E. 1973. Pattern Classification and Scene Analysis. John Wiley & Sons.

Fraenkel, Y. M., 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 the Biosciences 11, 379–87.

Galas, D. J., Eggert, M., and Waterman, M. S. 1985. Rigorous pattern-recognition methods for DNA sequences: Analysis of promoter sequences from Escherichia coli. Journal of Molecular Biology 186, 117–28.

Gionis, A., Indyk, P., and Motwani, R. 1999. Similarity search in high dimensions via hashing. In Proceedings of the 25th International Conference on Very Large Databases. Edinburgh, Scotland.

Hayes, W. S. and Borodovsky, M. 1998. Deriving ribosomal binding site (RBS) statistical models from unannotated DNA sequences and the use of the RBS model for N-terminal prediction. In Pacific Symposium on Biocomputing, 279–90.

Hertz, G. Z. and Stormo, G. D. 1999. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. Bioinformatics 15, 563–77.

Indyk, P. and Motwani, R. 1998. Approximate nearest neighbors: Towards removing the curse of dimensionality. In Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, 604–13. Dallas, TX.

Johnson, W. and Lindenstrauss, J. 1984. Extensions of Lipschitz mappings into a Hilbert space. Contemporary Mathematics 26, 189–206.

Kozak, M. 1983. Comparison of initiation of protein synthesis in procaryotes, eucaryotes, and organelles. Microbiological Reviews 47, 1–45.

Lawrence, C. E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F., and Wootton, J. C. 1993. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. Science 262, 208–14.

Lawrence, C. E. and Reilly, A. A. 1990. An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. Proteins: Structure, Function, and Genetics 7, 41–51.

Lewin, B. 1997. Genes VI. Oxford University Press.

Linial, M., Linial, N., Tishby, N., and Yona, G. 1997. Global self-organization of all known protein sequences reveals inherent biological signatures. Journal of Molecular Biology 268, 539–56.

Linial, N., London, E., and Rabinovich, Y. 1994. The geometry of graphs and some of its algorithmic applications. In Proceedings 35th Annual Symposium on Foundations of Computer Science, 577–91. Los Alamitos, CA.

Marsan, L. and Sagot, M.-F. 2000. Extracting structured motifs using a suffix tree – algorithms and application to promoter consensus identification. In RECOMB00: Proceedings of the Fourth Annual International Conference on Computational Molecular Biology, 210–19.

McInerny, C. J., Partridge, J. F., Mikesell, G. E., Creemer, D. P., and Breeden, L. L. 1997. A novel Mcm1-dependent element in the SWI4, CLN3, CDC6, and CDC47 promoters activates M/G_1-specific transcription. Genes & Development 11, 1277–88.

Means, A. L. and Farnham, P. G. 1990. Transcription initiation from the dihydrofolate reductase promoter is positioned by *hip1* binding at the initiation site. Molecular and Cellular Biology 10, 653–61.

Natsan, S. and Gilman, M. 1995. YY1 facilitates the association of serum response factor with the c-fos serum response element. Molecular and Cellular Biology 15, 5975–82.

Pevzner, P. and Sze, S.-H. 2000. Combinatorial approaches to finding subtle signals in DNA sequences. In Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, 269–78. AAAI Press, San Diego, CA.

Rigoutsos, I. and Califano, A. 1993. FLASH: a fast look-up algorithm for string homology. In Proceedings of the First International Conference on Intelligent Systems for Molecular Biology, 56–64.

Rigoutsos, I. and Floratos, A. 1998. Motif discovery without alignment or enumeration. In RECOMB98: Proceedings of the Second Annual International Conference on Computational Molecular Biology, 221–7. New York, NY.

Rocke, E. and Tompa, M. 1998. An algorithm for finding novel gapped motifs in DNA sequences. In RECOMB98: Proceedings of the Second Annual International Conference on Computational Molecular Biology, 228–33. New York, NY.

Sagot, M.-F. 1998. Spelling approximate repeated or common motifs using a suffix tree. In Lucchesi, C. L. and Moura, A. V., eds., Latin '98: Theoretical Informatics, volume 1380 of Lecture Notes in Computer Science, 111–27. Springer.

Sinha, S. and Tompa, M. 2000. A statistical method for finding transcription factor binding sites. In Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, 344–54. AAAI Press, San Diego, CA.

Staden, R. 1989. Methods for discovering novel motifs in nucleic acid sequences. Computer Applications in the Biosciences 5, 293–8.

Swoboda, P., Adler, H. T., and Thomas, J. H. 2000. The RFX-type transcription factor DAF-19 regulates sensory neuron cilium formation in C. elegans. Molecular Cell 5, 411–21.

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, 262–71. AAAI Press, Heidelberg, Germany.

van Helden, J., André, 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, 827–42.

Wingender, E., Dietze, P., Karas, H., and Knüppel, R. 1996. TRANSFAC: a database on transcription factors and their DNA binding sites. Nucleic Acids Research 24, 238–41. `http://transfac.gbf-braunschweig.de/TRANSFAC/`.

Wolfson, H. J. and Rigoutsos, I. 1997. Geometric hashing: an overview. IEEE Computational Science and Engineering 4(4), 10–21.

Table 1:

AVERAGE PERFORMANCE COEFFICIENTS FOR PLANTED (l, d)-MOTIFS IN SYNTHETIC DATA[a]

l	d	GibbsDNA	WINNOWER	SP-STAR	PROJECTION	Correct	m
10	2	0.20	0.78	0.56	0.80 ± 0.02	100	72
11	2	0.68	0.90	0.84	0.94 ± 0.01	100	16
12	3	0.03	0.75	0.33	0.77 ± 0.03	96	259
13	3	0.60	0.92	0.92	0.94 ± 0.01	100	62
14	4	0.02	0.02	0.20	0.71 ± 0.05	86	647
15	4	0.19	0.92	0.73	0.93 ± 0.01	100	172
16	5	0.02	0.03	0.04	0.67 ± 0.06	77	1292
17	5	0.28	0.03	0.69	0.94 ± 0.01	98	378
18	6	0.03	0.03	0.03	0.73 ± 0.06	82	2217
19	6	0.05	0.03	0.40	0.94 ± 0.01	98	711

[a]Each problem instance consists of $t = 20$ sequences each of length $n = 600$. Average performance coefficients of GibbsDNA, WINNOWER ($k = 2$), and SP-STAR are from Pevzner and Sze (personal communication), who averaged over eight random instances. For PROJECTION, we report averages and 95% confidence intervals for performance coefficient over 100 random instances with projection size $k = 7$ and threshold $s = 4$.

Table 2:

Statistics of spurious (l, d)-motifs in synthetic data[a]

l	d	$E(l,d)$	$E(l+1,d)$	a.p.c.	Correct	Spurious	19/20	m
9	2	1.6	6.1×10^{-8}	0.28	11	5	4	1483
11	3	4.7	3.2×10^{-7}	0.026	1	13	6	2443
13	4	5.2	4.2×10^{-7}	0.062	2	15	3	4178
15	5	2.8	2.3×10^{-7}	0.018	0	7	13	6495
17	6	0.88	7.1×10^{-8}	0.022	0	8	12	9272

[a]Parameters used were $k = 7$, $s = 4$, m as shown. Column headings: "a.p.c" = average performance coefficient over twenty problem instances; "Correct" = instances yielding correct motif consensus; "Spurious" = instances yielding equally good but spurious consensus; "19/20" = instances yielding a consensus with nineteen occurrences.

Table 3:

PERFORMANCE OF PROJECTION ON EUKARYOTIC PROMOTER SEQUENCES [a]

Sequence	Input Size (seqs / bases)	m	Start Points	Best (20,2) Motif from PROJECTION	Published Reference Motif
preproinsulin	4 / 7689	15	5759	GAAATTGCAGCCTCAGCCCC	CCTCAGCCCC [b]
			5322	CCCTAATGGGCCAGGCGGCA	CTAATG [c]
DHFR	4 / 800	15	175	TGCAATTTCGCGCCAAACTT	ATTTCnnGCCA [d]
metallothionein	4 / 6823	15	3363	CTCTGCGCCCGGACCGGTTC	TGCRCYCGG [e]
			3136	GTGCGCTCGGCTCTGCCAAG	TGCGCTCGG [f]
			2937	AGGGAGCTCTGCACACCACC	TGCACACCG [g]
c-fos	4 / 3695	8	1071	ATATTAGGACATCTGCGTCA	...CCATATTAGGACATCT [h]
yeast ECB	5 / 5000	8	1339	GGAAATTTCCCGTTTAGGAA	TTtCCcnntnaGGAAA [i]

[a] All motifs were found using parameters $l = 20$, $d = 2$, $k = 7$, and $s = 3$. Underlined portions of motifs indicate matches to known sequence features. "Start points" counts total number of buckets used as start points for EM refinement.

[b] TRANSFAC signal (Wingender et al., 1996)

[c] CT-II element (Boam et al., 1990)

[d] non-TATA transcription start signal (Means and Farnham, 1990)

[e] MREa promoter (Andersen et al., 1987)

[f] MREd promoter (Andersen et al., 1987)

[g] MREf promoter (Andersen et al., 1987)

[h] $3'$ end of c-fos serum response element (Natsan and Gilman, 1995)

[i] yeast early cell cycle box (McInerny et al., 1997)

Table 4:

(6,1)-motifs found as candidate 16S rRNA binding sites in prokaryotes[a]

Organism	t	s	m	Motif	Occurrences	16S rRNA	Best z-score
M. jannaschii	1679	196	14	AGGTGA	606	GGAGGTGATCC	GGTGA
H. influenzae	1716	202	17	AGGAAA	639	TAAGGAGGTGA	AAGGA
T. maritima	1846	216	13	GGAGGT	1198	GAAAGGAGGTG	AGGTG
B. subtilis	4099	480	35	AGGAGG	2742	TAGAAAGGAGG	AGGAG
E. coli	4287	502	35	AAGGAG	1306	TAAGGAGGTGA	AGGAG

[a]All experiments were performed with projection size $k = 4$. Column headings: t = number of sequences; s = bucket size threshold, m = number of iterations; "Occurrences" = number of input sequences containing motif with up to one substitution; "16S rRNA" = reverse complement to $3'$ end of organism's 16S rRNA, which should be similar to true binding site; "Best z-score" = 5-mer with greatest z-score using the algorithm of (Tompa, 1999).

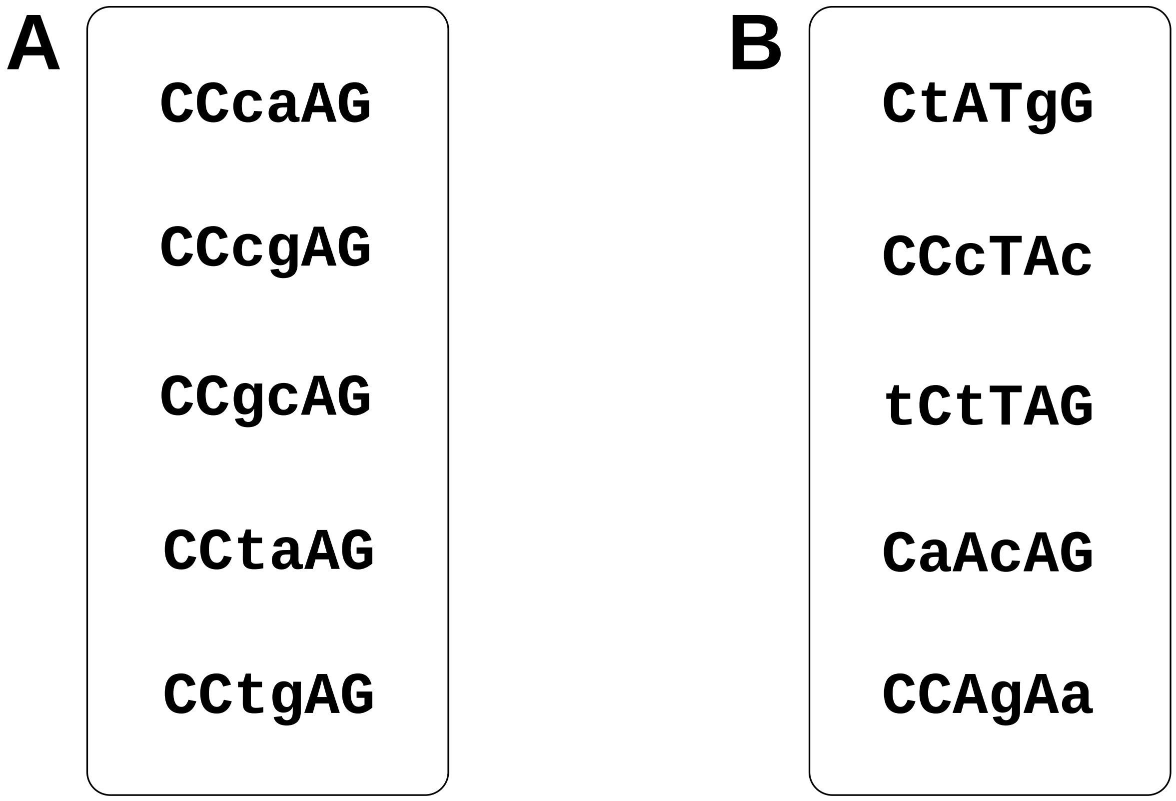

Figure 1: two hexamer motifs in which each occurrence differs by exactly two substitutions from the string `CCATAG`. Lower case letters indicate substitutions. Although the two motifs have the same number of substitutions overall, motif **B**'s substitutions are distributed uniformly throughout its occurrences (as in the challenge problem), while **A**'s substitutions are concentrated in its two center positions.

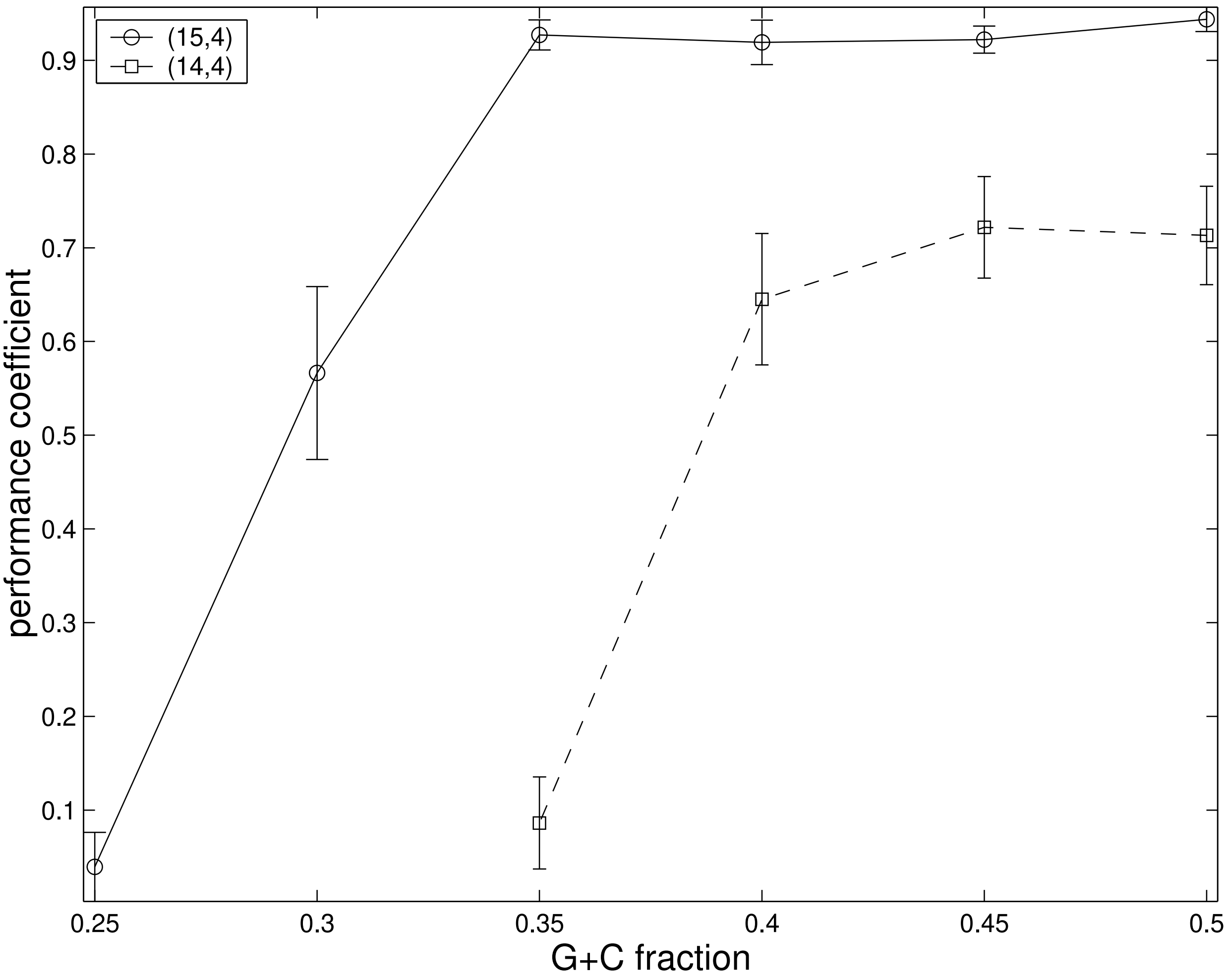

Figure 2: performance of PROJECTION on synthetic motif-finding problems with unequal background base frequencies. Problems were generated as in Section 3.1, except that the background $G+C$ fraction was set to values different from 0.5 as shown. Problem instances contained either (15,4)-motifs (solid line) or (14,4)-motifs (dashed line) generated with equal base frequencies. Error bars indicate 95% confidence intervals over 100 random trials.

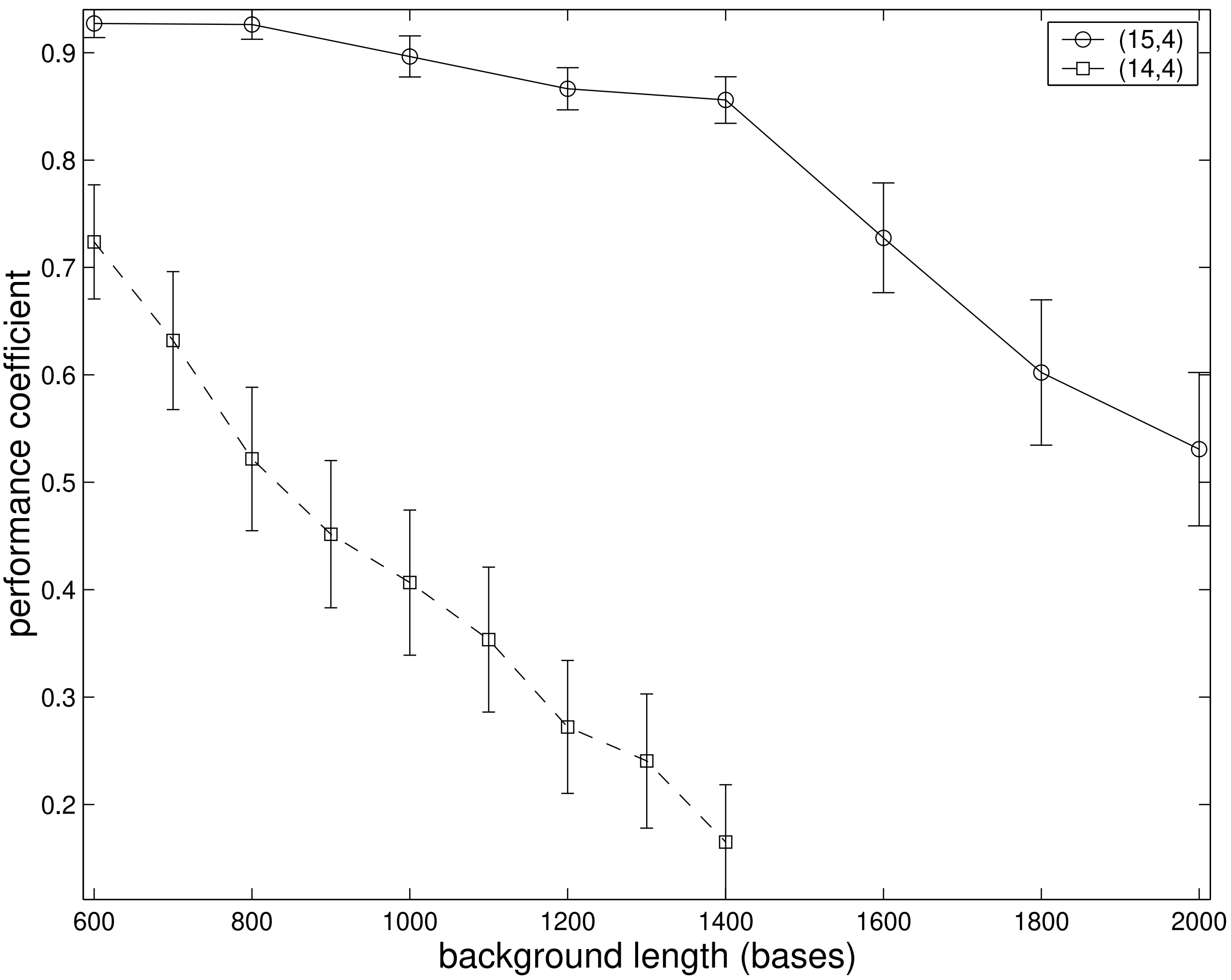

Figure 3: performance of PROJECTION on synthetic motif-finding problems with increasing background sequence lengths. Problems were generated as in Section 3.1, except that the background sequence length was scaled from $n = 600$ to $n = 2000$ as shown. The number of sequences was maintained at $t = 20$. Problem instances contained either (15,4)-motifs (solid line) or (14,4)-motifs (dashed line). Error bars indicate 95% confidence intervals over 100 random trials.