Performance Comparison of Algorithms for Finding Transcription Factor Binding Sites

Saurabh Sinha*
Martin Tompa

Department of Computer Science and Engineering
Box 352350
University of Washington
Seattle, WA 98195-2350 U.S.A.
{saurabh,tompa}@cs.washington.edu

Abstract

We compare the accuracy of three motif-finding algorithms for the discovery of novel transcription factor binding sites among co-regulated genes. One of the algorithms (YMF) uses a motif model tailored for binding sites and an enumerative search of the motif space, while the other two (MEME and AlignACE) use a more general motif model and local search techniques. The comparison is done on synthetic data with planted motifs, as well as on real data sets of co-regulated genes from the yeast S. cerevisiae. *More often than not, the enumerative algorithm is found to be more accurate than the other two on the yeast data sets, though there is a noticeable exclusivity in the accuracy of the different algorithms. The experiments on synthetic data reveal, not surprisingly, that each algorithm outperforms the others when motifs are planted according to its motif model.*

1 Introduction

One of the major challenges facing biologists is to understand the mechanisms governing the regulation of gene expression. An important step in unraveling the regulatory interactions is to identify common binding sites in the regulatory regions of potentially co-regulated genes. This gives rise to the *motif-finding* problem: *Given a set of sequences (promoter regions), detect overrepresented motifs that are good candidates for being transcription factor binding sites.*

There are numerous motif-finding programs from which to choose, including MEME [1], Consensus [5], AlignACE [8], Oligo-Analysis [11], YMF [9, 10], AnnSpec [12], and Projection [4], and little guidance to select among them. The only previous performance comparisons that have been done for some of these programs are by Pevzner and Sze [7] and by Buhler and Tompa [4], using simulated data generated according to a motif model that may not be accurate for transcription factor binding sites.

In this paper, we compare the accuracy of an enumerative motif-finding algorithm YMF [9, 10] to that of two popular contenders, MEME [1] and AlignACE [8]. The comparison is done on real data sets from yeast, and on synthetic data with planted motifs. (See Blanchette *et al.* [2] for an analogous performance comparison among phylogenetic footprinting algorithms.) The experiments on synthetic data indicate that each algorithm's accuracy depends on the fit between the stochastic process planting the motifs and the algorithm's underlying motif model. On real data sets in yeast, YMF is more accurate than the other two algorithms more often than not.

The three programs we have chosen represent three very different types: an enumerative method, a Gibbs sampler, and a local search based on Expectation-Maximization. We wanted to include Oligo-Analysis [11], but it does not seem to be available for downloading.

2 Motif Models and Algorithms

Both MEME and AlignACE model motifs as a *weight matrix*. This is a matrix with one row for each of the 4 bases, and one column for each position in the

*Current address: Box 25, The Rockefeller University, 1230 York Ave, New York, NY 10021, saurabh@lonnrot.rockefeller.edu motif. The number in the i^{th} row and j^{th} column is the frequency with which the i^{th} base is found in the j^{th} position of the motif.

MEME uses statistical modelling techniques to automatically choose the best width, number of occurrences, and description of each motif. It uses an Expectation-Maximization algorithm to fit a two-component finite mixture model to the sequence data, the two components being the motif and the nonmotif (background) parts of the sequences. AlignACE uses a variant of Gibbs sampling [6].

YMF uses a *consensus model* of motifs. A motif for this program is a string over the alphabet $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}, \mathtt{R}, \mathtt{Y}, \mathtt{S}, \mathtt{W}, \mathtt{N}\}$, with consecutive `N`'s only at the center, and a limited number of `R` (`A` or `G`), `Y` (`C` or `T`), `S` (`C` or `G`), and `W` (`A` or `T`) characters. The character `N` is called a *spacer* and the characters `R`, `Y`, `S`, and `W` are called *degenerate symbols*. The number of nonspacer characters in a motif is called the *significant length* of the motif. YMF searches the entire space of motifs and reports the motifs sorted by their z-scores [9]. For each reported motif, YMF also reports the *significance* of its z-score, which is the probability that a motif with that z-score or higher is found in random sequences of the same length and number as the input. A program called FindExplanators (Blanchette and Sinha [3]) is used to remove *redundant* motifs from the output of YMF. This program takes a list of motifs output by YMF and reports a user-specified number of independently significant motifs from this list.

The compared versions of these three tools are as follows:

1. For MEME, we used version 3.0.4 available at `ftp.sdsc.edu/pub/sdsc/biology/meme/`.
2. For AlignACE, we used the Linux version available at `atlas.med.harvard.edu/download/`.
3. For the YMF-FindExplanators suite, we used the versions available at `bio.cs.washington.edu/software.html`.

3 Comparison Outline

We use the following score for measuring the performance of a motif-finding algorithm. Let $S = \{S_1, S_2, \ldots S_n\}$ be the set of n input sequences. For any motif m, let I_{mi} be the set of positions in sequence S_i that are occupied by an occurrence of m (which may occur 0 or more times in S_i). Assume that we know the occurrences of the "true" motif m^k, and are evaluating the motif m^r reported by an algorithm. The performance score Φ is defined as follows:

$$\Phi(S, m^k, m^r) = \frac{\sum_{i=1}^{n} |I_{m^k i} \cap I_{m^r i}|}{\sum_{i=1}^{n} |I_{m^k i} \cup I_{m^r i}|}.$$

In other words, it is the number of positions, over all sequences, where occurrences of the known and reported motifs overlap, divided by the total number of positions at which the known *or* the reported motif occurs. This measure was proposed by Pevzner and Sze [7] for similar purposes. Note that if the reported occurrences exactly concur with the known occurrences, the score is 1, and when the reported and known occurrences have no position in common, it is 0.

Each experiment begins with a set of sequences, and locations of occurrences of the known motif m^k in these sequences. Each of the three algorithms is executed on the sequences, and required to report T motifs, T being a parameter. Each of these T motifs is treated in turn as the reported motif m^r, and the performance score Φ is computed as described above. The best of the T performance scores thus obtained is considered as the score of the corresponding algorithm in the current experiment. Our experiments use $T \in \{1, 3\}$. On real data sets, we allowed $T > 1$ to avoid penalizing an algorithm that finds the known motif, but not necessarily as the highest ranked. This resulted in improving the relative accuracy of AlignACE significantly.

MEME can be made to report the T best motifs by using the parameter *nmotifs*. AlignACE has an internal control on the number of motifs to report. However, its motifs are ordered by their "MAP" scores, which can be used to pick the top T motifs. The YMF-FindExplanators suite of programs is executed as follows. YMF is run three times, with parameter values ($l = 6, \lambda = 11, \delta = 2, t = 1000$), ($l = 7, \lambda = 0, \delta = 2, t = 1000$), and ($l = 8, \lambda = 0, \delta = 2, t = 1000$), respectively, where l is the significant length, λ is the maximum number of spacers allowed in the motif, δ is the maximum number of degenerate symbols, and t is the number of motifs to report. FindExplanators is then executed separately on each list of t motifs reported by YMF, and the T best motifs are selected in each execution, for a total of $3T$ motifs. These are then sorted by the *significance* of their z-scores, and the best T motifs from the sorted list of $3T$ are reported. We henceforth refer to the YMF-FindExplanators suite as YMF.

4 Comparison on Synthetic Data

In this section, we describe experiments where sequences are generated at random, a randomly chosen motif is planted in these sequences, and the accuracy of the three motif-finders is evaluated on the sequences. The planted motif is chosen either according to a consensus model or a weight matrix model. In each of these experiments, n sequences, each of length 1000, are generated according to a 3^{rd} order Markov model trained on the promoter regions of all yeast genes.

4.1 Motifs of the Consensus Model

In one class of experiments, the planted motif m^k follows the consensus model that YMF uses. (See Section 2.) Each experiment is parameterized by the number n of sequences, the number T of motifs reported by each algorithm, the motif m^k to be planted, and z, the strength with which it is planted. m^k is of length 8, has no spacers, and has up to 2 degenerate symbols. The relevant details of the experiments are as follows.

1. We compute the number N of times that m^k needs to occur in the n generated sequences so that its z-score [9] is the value of the parameter z. m^k is then planted at N random positions in the n sequences. Each occurrence is planted independently in a sequence chosen at random (with replacement), at a random position, while ensuring that previously planted occurrences are not overwritten. The planting process ensures that there are exactly N planted occurrences of the motif, including chance occurrences. If m^k has any degenerate symbol, it is instantiated into one of its two possibilites with equal probability. All occurrences are planted in the same orientation, since each of the algorithms tested looks at both strands for motif occurrences.

2. YMF is run as described in the previous section, except that the significant length of motifs is restricted to 8. Let Φ_y denote the performance score.

3. MEME is run with the following parameter values: the exact length of the motifs is 8 (set using parameters $w = 8$, and *nomatrim*), multiple occurrences are allowed per sequence (*mod* = *tcm*) since this is true of the planting process, and the background distribution is the 3^{rd} order Markov model used to generate the random sequences (*bfile* = *Markov*). Let Φ_m be the performance score obtained.

4. AlignACE is run with motif length 8 (using the parameters *numcols* and *nocols*). The *oversample* parameter, which controls the fraction of the motif space searched, is set to 2 to allow more exhaustive search than the default case. Let Φ_a denote the performance score.

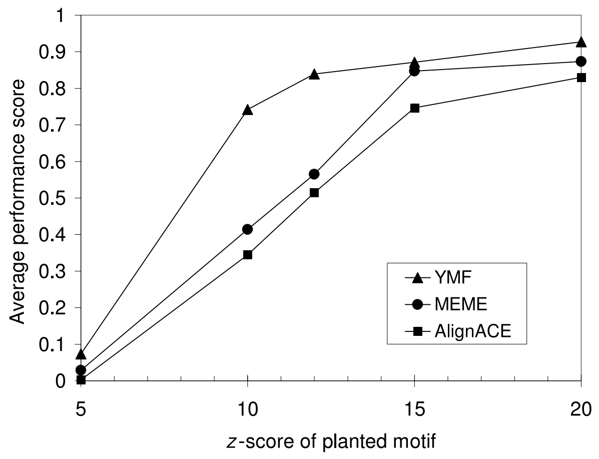

Figure 1: Performance of three motif-finding algorithms (YMF, MEME, and AlignACE) on 10 sequences of length 1000 each, with planted consensus motifs. Each point represents the average of the performance scores for a particular algorithm and for a specific z-score of the planted motif, the average being over 100 experiments, each using a different planted motif.

The experiment was repeated for different values of T, n and z, and for each (T, n, z) triplet, 100 different motifs were chosen at random and used as m^k. Fig. 1 summarizes the results for $T = 3$ and $n = 10$. It shows, for each z, the average of Φ_y, Φ_m, Φ_a over the 100 different runs for that value of z. YMF is more accurate than both MEME and AlignACE in these experiments. Further experiments showed that the performance gap increases for $n = 20$, since more sequences imply a greater likelihood of the local search algorithms not reaching the global optimum. The same trend was observed for $T = 1$. We did not do rigorous comparisons for $n > 20$, since MEME takes prohibitively long to complete its execution for such large values of n. We also noted that the running time of YMF scales better with increasing n. On a Pentium III machine with 512 MB RAM, and for $n = 10$, YMF, MEME and AlignACE ran for an average of 100 sec, 323 sec, and 84 sec respectively. For $n = 20$, these average running times were 147 sec, 956 sec, and 310 sec respectively.

We also compared four different versions of MEME among each other, in the same experimental framework as above. These variants differed in the values of the *mod* parameter, which may be set to *tcm* (multiple occurrences per sequence) or *zoops* (at most one occurrence per sequence), and the *bfile* parameter, which may be set to *Markov* or to *default*, the latter using single nucleotide frequencies in the input sequences. The *mod* = *tcm* and *bfile* = *Markov* combination was the most accurate, which is why it was used in the comparison to other motif finders. In the case of AlignACE, we compared the accuracy of two variants with the *oversample* parameter set to 1 and 2 respectively. The latter was found to perform marginally better. Since AlignACE executes relatively fast, allowing it more computational time by setting this parameter is justified.

Note that YMF has an intrinsic advantage over the other two algorithms in this experimental set-up, because the planted motifs exactly match its motif model. The results of this section show that if the YMF motif model is accurate for the application of interest, then YMF should perform significantly better than the weight matrix based local search approaches, even for few sequences. The results also reveal the effects of different parameter settings on the accuracy of MEME and AlignACE.

4.2 Motifs of the Weight Matrix Model

In this class of experiments, the planted motif m^k follows the weight matrix model that is used by both MEME and AlignACE. Each experiment is parameterized by the number n of sequences, the number T of motifs reported by each algorithm, the number N of occurrences of the motif, the relative entropy R of the weight matrix motif to be planted, and a flag called "uneven", which indicates how the relative entropy is distributed among the positions of the motif. The planted motif is of length $l = 8$. Once the n random sequences (of length 1000 each) have been generated, the motif is planted as follows. Compute the distribution of the relative entropy (R bits) in the l columns: if the *uneven* parameter is false, then each column contributes an equal amount of relative entropy (R/l bits); if the parameter is true, then $\lfloor R/r_{max} \rfloor$ randomly chosen positions contribute r_{max} bits each to the relative entropy, where r_{max} is the maximum possible relative entropy of any column, and the remaining bits are evenly distributed over the remaining columns. Construct an alignment of N strings (over the alphabet {A, C, G, T}) of length l, one column at a time, each column having relative entropy as decided in the previous step. Such a column is chosen at random from all columns of N entries that have the same relative entropy. This step gives us the N motif occurrences that are then planted at random positions in the n sequences, allowing multiple occurrences per sequence.

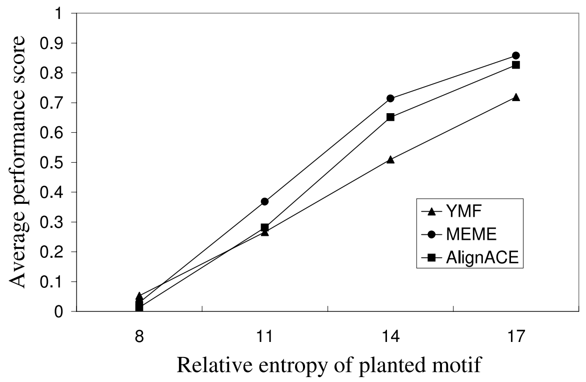

Figure 2: Performance of YMF, MEME, and AlignACE on 10 sequences of length 1000 each, with planted weight matrix motifs. Each point represents the average of the performance scores for a particular algorithm and for a specific strength of the planted motif, the average being over 50 experiments, each using a different planted motif.

The three algorithms were parameterized as in Section 4.1. The experiment was done for $n \in \{10, 20\}$, $T = 3$, $N = 20$, $R \in \{8, 11, 14, 17\}$, and both values of *uneven*. For each choice of parameter values, 50 experiments were performed with different planted motifs. Figure 2 summarizes the results for $n = 10$ and *uneven* = *false*. We notice that both MEME and AlignACE are more accurate than YMF, because the planted motifs match their motif model. The same trend is observed for *uneven* = *true*, and for $n = 20$.

5 Comparison on Yeast Regulons

The database SCPD [13] reports several regulons in yeast, each regulon being a set of co-regulated genes whose promoters share binding sites for the same transcription factor. For each regulon, the known binding sites are tabulated in the database. The following sequence of steps was performed to obtain the performance scores on each regulon. There is a single parameter T, which is the number of motifs each algorithm is allowed to report.

1. Extract 800 bp long promoter regions of genes in the regulon. Let these be $S = \{S_1, S_2, \ldots S_n\}$.

Use the tabulated occurrences of the known binding site for this regulon as the motif m^k.

2. Run YMF on S, as described in Section 3, to obtain T motifs. Let the performance score be denoted by Φ_y.

3. Run MEME on S to obtain T motifs of length between 6 (parameter *minw*) and 17 (parameter *maxw*). MEME is able to decide the best width of the motif itself, but since YMF uses the prior knowledge that yeast binding sites are usually between 6 and 17 in length, we supplied this information to the MEME algorithm, for fairness. Multiple occurrences are allowed per sequence (*mod* = *tcm*) and the background distribution used is the 3^{rd} order Markov model trained on all yeast promoters (*bfile* = *Markov*). Let Φ_m be the performance score.

4. Run AlignACE on S to obtain the top T motifs with 6 conserved columns ($numcols = 6$), with the *oversample* parameter set to 2 to allow more exhaustive sampling of the search space than the default case. Let Φ_a denote the performance score.

The experiment was performed for all 34 regulons in SCPD that have at least three genes. Some of them do not have a catalogued consensus sequence for the transcription factor binding sites, often because the tabulated binding sites do not have a good consensus, in the sense of YMF's motif model. (It is possible that a weight matrix is the superior model for these binding sites.) These experiments truly represent the typical scenario for the transcription factor binding site problem. The results, for $T = 3$, are presented in Table 1. For each regulon, we declare an algorithm the "winner" if it has the highest performance score for that regulon, and this score is above 0.10. Each row of the table corresponds to a separate regulon, and the score of the "winning" algorithm for that regulon is underlined.

In Table 2, we see the pairwise comparison between YMF and each of several different variants of MEME and AlignACE. The results for both $T = 3$ and $T = 1$ are tabulated. Notice that the best performing variants of MEME and AlignACE are those considered in Table 1.

6 Discussion

YMF outperforms both MEME and AlignACE on 11 of the 34 regulons in Table 1, MEME winning 9 times and AlignACE 5 times. (In the remaining nine

Table 1: Performance comparison of different motif finders on yeast regulons. "Size" is the number of genes in the regulon. The columns labelled "time" report the time to completion for each algorithm, in seconds.

Regulon	Size	YMF		MEME		AlignACE	
		Φ_y	time	Φ_m	time	Φ_a	time
ABF1	19	$\underline{0.33}$	171	0.01	1645	0.00	280
BAS1	6	0.02	176	0.03	246	0.02	162
CAR1	12	$\underline{0.31}$	151	0.25	771	0.20	183
CPF1	3	$\underline{0.62}$	110	0.49	86	0.02	72
CSRE	4	0.28	125	$\underline{0.32}$	149	0.25	75
GAL4	6	0.61	176	$\underline{0.66}$	232	0.61	171
GATA	4	$\underline{0.57}$	128	0.19	149	0.54	124
GCN	38	$\underline{0.25}$	414	0.00	8523	0.00	641
GCR1	6	0.05	196	0.20	252	$\underline{0.31}$	78
GLN3	3	0.00	129	0.00	83	0.00	100
HAP1	5	$\underline{0.15}$	139	0.12	208	0.10	110
HAP2	4	0.00	93	0.00	150	0.02	93
HSE	6	$\underline{0.39}$	158	0.23	247	0.31	216
MATA1	3	0.19	101	$\underline{0.20}$	85	0.11	75
MATA2	7	0.06	197	$\underline{0.36}$	359	0.03	175
MCB	6	0.54	122	0.15	238	$\underline{0.55}$	222
MCM1	23	0.32	557	$\underline{0.51}$	3532	0.50	243
MIG1	9	0.28	188	0.00	505	$\underline{0.29}$	230
PDR3	7	$\underline{0.73}$	174	0.43	357	0.47	134
PHO2	3	0.00	126	0.00	84	0.00	89
PHO4	5	$\underline{0.26}$	161	0.05	209	0.22	123
RAP1	16	0.09	645	$\underline{0.31}$	2036	0.23	260
REB1	14	$\underline{0.39}$	396	0.34	1628	0.01	164
ROX1	3	0.00	90	0.03	83	0.00	25
RPA	3	$\underline{0.20}$	99	0.15	80	0.00	84
SCB	3	0.60	137	0.61	85	$\underline{0.84}$	75
SFF	3	0.00	136	0.00	80	0.05	111
STE12	4	0.60	176	0.02	144	$\underline{0.71}$	124
TBP	17	0.00	379	0.00	2253	0.00	277
UASCAR	3	0.02	178	$\underline{0.13}$	85	0.06	71
UASH	18	0.00	180	0.01	2301	0.00	393
UASPHR	17	0.01	556	0.02	2205	0.06	302
UIS	3	0.01	124	$\underline{0.43}$	82	0.20	104
URS1H	13	0.57	388	$\underline{0.73}$	1386	0.42	192
Wins		11		9		5	
#scores ≥ 0.2		18		16		16	
#scores ≥ 0.33		11		9		8	
#scores ≥ 0.5		8		4		6	

Table 2: Comparison of YMF with different variants of MEME and AlignACE on yeast regulons. M1: MEME with *mod = tcm* and *bfile = Markov.* M2: MEME with *mod = zoops* and *bfile = Markov.* M3: MEME with *mod = zoops* and *bfile = default.* A-6, A-8, A-10: AlignACE with *oversample = 2* and *numcols* = 6,8,10 respectively. The numbers in parentheses indicate how many of the wins/losses were by a margin more than 0.1.

Var.	T=3		T=1	
	YMF win	YMF loss	YMF win	YMF loss
M1	14 (10)	11 (7)	10 (10)	10 (4)
M2	15 (10)	6 (6)	12 (9)	6 (5)
M3	14 (12)	8 (7)	13 (11)	4 (3)
A-6	14 (8)	9 (6)	12 (11)	6 (4)
A-8	16 (10)	5 (2)	13 (11)	5 (1)
A-10	16 (14)	9 (6)	14 (13)	4 (3)

regulons, all algorithms had scores below 0.1.) The last three rows of the table also suggest that YMF has an edge over the other algorithms. For each of three different thresholds of success (0.2, 0.33, and 0.5), YMF has more successes than MEME or AlignACE. Table 2 shows that there is a substantial gap between YMF and AlignACE (with *numcols* = 6) in terms of the number of wins, in favour of the former, for both $T = 3$ and $T = 1$, the gap being slightly greater for $T = 1$. If we declare a win to be a "clear" win if the winning score is at least 0.1 more than the losing score, we find a similar performance gap between these two algorithms in terms of the number of "clear" wins, and the gap increases significantly from $T = 3$ to $T = 1$. There is a smaller performance gap between YMF and MEME (with *mod = tcm* and *bfile = Markov*) for $T = 3$ (again, in favour of the former), and both perform equally well for $T = 1$. However, in terms of "clear" wins, YMF has a sizeable advantage. These statistics indicate that YMF is more accurate than the other two algorithms (on yeast regulons) more often than not.

As we can see from Table 2, typically a large fraction of the wins of any of the three algorithms are "clear" wins. Thus there is substantial exclusivity in the accuracy of the algorithms. For instance, if we consider all regulons in Table 1 where at least one of the algorithms had a score above 0.33, we find that there are 14 such regulons, and only four of them have all three algorithms crossing the 0.33 threshold. A similar observation is made for the 0.5 score threshold. We believe that this exclusivity in accuracy is largely because the motifs in different regulons are more or less suited to a specific algorithm's motif model. For example, YMF has "clear" wins for regulons ABF1, CPF1, GCN, and PDR3, which have the binding site consensi TCRNNNNNNACG, TCACGTG, TGANTN, and TCCGYGGA respectively. Note that all of these, with the slight exception of TGANTN (GCN), belong to the YMF motif model. On the other hand, MEME has clear wins on MATA2, UIS, and URS1H. SCPD tabulates the binding site consensus of MATA2 as CRTGTWWWW, which allows positional variations that can be better captured by a weight matrix than a consensus motif. SCPD tabulates no consensus for UIS and URS1H, again for the same reason.

Another noteworthy observation in Table 1 is the time taken by the three programs. In 27 of the 34 cases, MEME has the longest completion time, and in all cases with 9 or more genes, it takes more than twice the time taken by the next fastest algorithm. The convergence criterion in the program may be configured to require less computation time, and we have not investigated the effect of such configuration on the performance comparison. We also note that AlignACE is often the fastest of the three algorithms, although YMF usually has a similar time to completion.

We noted that the different variants of AlignACE (with *numcols* $\in \{6, 8, 10\}$, respectively) had some exclusivity in accuracy among themselves. While YMF and MEME both have an internal control over the length of the motif to be reported, a single run of AlignACE reports motifs with the same number of conserved columns. We believe that the performance scores of AlignACE would improve if the program was able to choose from motifs with different numbers of conserved columns. Another observation about AlignACE is that the performance score fluctuates markedly between executions that use different random number seeds, and a method that combines the results from such executions might boost the accuracy of the algorithm. Different variants of MEME also showed some degree of exclusivity. For instance, the variant with *mod = zoops* secured a "clear" win over the *mod = tcm* variant for the regulons CAR1, CPF1, CSRE, PHO4, and RAP1, while the opposite was true for GAL4, GCR1, MATA2, SCB, and UIS. (Data not shown.) With the exception of PHO4, all of the former regulons have an average of 1.5 or fewer binding sites per sequence, while all of the latter regulons have more than 1.5 sites per sequence.

Our experiments with synthetic sequences and planted consensus motifs showed that the accuracy of MEME and AlignACE degrades relative to that of

YMF as the number of sequences increases. This may be the result of the former algorithms getting stuck at local optima. However, this effect was not pronounced in the experiments with yeast regulons – in the 10 regulons with more than 10 genes, YMF secured four wins while MEME was the winner in three cases. On the other hand, both MEME and AlignACE have very low performance scores (0.01 or below) for the ABF1 and GCN regulons, which are two of the three largest regulons considered.

7 Conclusion

We have compared the accuracy of YMF for finding transcription factor binding sites to that of two algorithms MEME and AlignACE that use a more powerful motif model. Each class of algorithms was found to be more accurate than the other when the planted motif follows its motif model more closely. On real data sets in yeast, YMF is the most accurate algorithm more often than not, though there is substantial exclusivity in the accuracy of the various algorithms.

Acknowledgments

This material is based upon work supported in part by a Microsoft Fellowship, in part by the National Science Foundation under grants DBI-9974498 and DBI-0218798, and in part by the National Human Genome Research Institute under grant R01 HG02602-01.

References

[1] T. L. Bailey and C. Elkan. Fitting a mixture model by expectation maximization to discover motifs in biopolymers. In *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, pages 28–36, Menlo Park, CA, 1994. AAAI Press.

[2] M. Blanchette, S. Kwong, and M. Tompa. An empirical comparison of tools for phylogenetic footprinting. In *3rd IEEE Symposium on Bioinformatics and Bioengineering*. IEEE Computer Society, Mar. 2003.

[3] M. Blanchette and S. Sinha. Separating real motifs from their artifacts. *Bioinformatics*, 17:S30–S38, 2001.

[4] J. Buhler and M. Tompa. Finding motifs using random projections. *Journal of Computational Biology*, 9(2):225–242, 2002.

[5] G. Z. Hertz and G. D. Stormo. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, 15(7/8):563–577, July/August 1999.

[6] C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wootton. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, 262:208–214, 8 October 1993.

[7] P. Pevzner and S.-H. Sze. Combinatorial approaches to finding subtle signals in DNA sequences. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 269–278, San Diego, CA, Aug. 2000. AAAI Press.

[8] F. P. Roth, J. D. Hughes, P. W. Estep, and G. M. Church. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nature Biotechnology*, 16:939–945, Oct. 1998.

[9] S. Sinha and M. Tompa. A statistical method for finding transcription factor binding sites. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 344–354, San Diego, CA, Aug. 2000. AAAI Press.

[10] S. Sinha and M. Tompa. Discovery of novel transcription factor binding sites by statistical overrepresentation. *Nucleic Acids Research*, 30(24):5549–5560, Dec. 2002.

[11] J. van Helden, B. André, and J. Collado-Vides. Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *Journal of Molecular Biology*, 281(5):827–842, Sept. 4, 1998.

[12] C. T. Workman and G. D. Stormo. ANN-Spec: a method for discovering transcription factor binding sites with improved specificity. In *Pacific Symposium on Biocomputing*, pages 464–475, Honolulu, Hawaii, Jan. 2000.

[13] J. Zhu and M. Q. Zhang. SCPD: a promoter database of the yeast *Saccharomyces cerevisiae*. *Bioinformatics*, 15(7/8):563–577, July/August 1999. `http://cgsigma.cshl.org/jian/`.