PhyloGibbs: A Gibbs Sampler Incorporating Phylogenetic Information

Rahul Siddharthan[1], Erik van Nimwegen[2], and Eric D. Siggia[1]

[1] Center for Studies in Physics and Biology, The Rockefeller University,
1230 York Avenue, New York, NY 10021, USA
[2] Division of Bioinformatics, Biozentrum, University of Basel,
Klingelbergstrasse 50/70, CH-4056 Basel, Switzerland

Abstract. We present a new Gibbs sampler algorithm with the motivation of finding motifs, representing candidate binding sites for transcription factors, in closely related species. Since much conservation here arises not from the existence of functional sites but simply from the lack of sufficient evolutionary divergence between the species, a conventional Gibbs sampler will fail. We compare the effectiveness against conventional methods on closely-related yeast sequences. Our algorithm is also applicable to single-species or phylogenetically-unrelated sequences, and has further improvements over previous Gibbs samplers, including accounting for correlations in the “background” model, an option to search for “dimers” (pairs of motifs with variable spacing), and a “tracking” strategy that allows us to assess the significance of candidate motifs.

1 Introduction

Gene transcription is regulated by transcription factors, proteins that bind upstream of a gene and typically recognise a short conserved pattern, or “motif”, in the DNA. The development of motif-finding algorithms to scan regulatory regions and look for overrepresented motifs is thus of great interest.

For a motif finder to be effective, there must be several copies of a motif to find: it is impossible to detect just one copy of a motif without other prior knowledge, and hard to conclude that two fuzzy copies indicate overrepresentation. To increase the number of copies, one option is to examine genes that are known to be regulated by the same factor. This is not always possible, though often hints can be drawn from microarray experiments. But another option is offered by the increasing number of genomes of closely related species that have appeared in the recent past: we can increase the amount of sequence available by looking at regulatory regions of homologous genes in different closely-related species. For example, sequences of four near relatives of the yeast *S. cerevisiae* (namely, *S. kudriavzevii* [1], *S. bayanus, S. mikatae* [1,2], and *S. paradoxus* [2]) have been published, as well as two more distantly related species, *S. castellii* and *S. kluyveri* [1]. Similarly, in addition to the fruit fly *D. melanogaster*, its close relative *D. pseudoobscura* has been sequenced, and fragments of sequence for various other near Drosophila species exist. In the mammalian world, comparative genomics using the human, chimpanzee,mouse and rat genomes appears promising.

E. Eskin, C. Workman (Eds.): RECOMB 2004 Ws on Regulatory Genomics, LNBI 3318, pp. 30–41, 2005.

Attempts have been made, for example in the yeast papers by Cliften et al. [1] and Kellis et al. [2] to find motifs using this extra phylogenetic information, but these have been in the nature of phylogenetic "screens" that concentrate on conserved blocks. Here we evolve a method that accounts for both conserved and non-conserved regions in a transparent and consistent way: this is important because known functional binding sites are not always conserved in other species [3, 4]. Our starting point is the Gibbs sampler. This is a Markov-chain Monte Carlo method [5] to sample a phase space by making a choice at each step from numerous possible moves, weighted by their probabilities; it is more computationally intensive than the usual Metropolis algorithm (where a random move is tried and accepted or rejected), but in problems such as this, converges much faster. In the biological motif-finding context it was introduced by Lawrence et al. [6, 7].

In addition to phylogeny, we make other enhancements to the idea of the Gibbs sampler, the most important of which is a "tracking" mechanism to determine the significance of motifs. This is described in detail later.

2 Scoring with Phylogenetic Conservation

Motifs are often represented by "weight matrices" [8] $w_{\alpha n}$, the probability of finding base α (= A, C, G, T) at site n of the motif (summing over α to 1 for each n.) It is assumed that different columns of the weight matrix are independent.

Most intergenic DNA is probably not functional; non-functional sites are assumed to be described by a "background model" instead. Rather than use raw base counts, we use a background model that incorporates correlations, described below.

With closely related species, much sequence is conserved not because of functionality (presence of binding sites) but because the species are too recently diverged to have mutated significantly. Typical motif finders described above can get misled by such meaningless conservation; we want to account for phylogenetic conservation and adjust the scoring of motifs for this.

When one tries to align these intergenic regions, using alignment tools such as Clustalw [9] or Dialign [10], one finds that there are large blocks of sequence that are highly conserved, interspersed with significant blocks of unconserved, inserted or deleted sequence between different species. We want to treat the non-conserved blocks just as we would an independent sequence, while accounting for phylogeny in the conserved blocks.

We use the following strategy: First, we identify phylogenetically conserved blocks in the sequence, using the alignment tool Dialign [10], with rather stringent parameters for identifying conserved blocks, so that aligned regions are typically rather highly conserved.

Then, we parse the sequence into "windows" – possible sites for motifs, all necessarily all of the same length not counting gaps. In the absence of phylogenetic alignment, "windows" are simply stretches of sequence of length L (the length of the motif), as in figure 1. With phylogenetically aligned regions, windows extend across all aligned sequences – that is, if a base in one sequence has an aligned base in another sequence, that other sequence must be part of the window, as illustrated in figure 2. Windows must be "consistent": there are no "gaps" and pairs of aligned bases are always a consistent distance apart. Thus, we are assuming that a putative binding site in an aligned block is a candidate site either in all the aligned sequences, or in none of them.

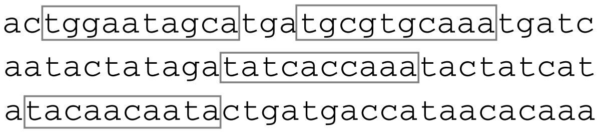
actggaatagcatgatgcgtgcaaatgatc
aatactatagatatcaccaaatactatcat
atacaacaatactgatgaccataacacaaa

Fig. 1. Independent sequences (without dialign constraints) and an example of a configuration where four windows have been placed

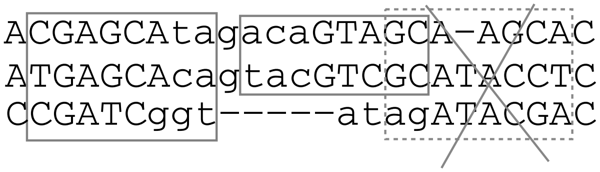
ACGAGCAtagacaGTAGCA-AGCAC
ATGAGCAcagtacGTCGCATACCTC
CCGATCggt-----atagATACGAC

Fig. 2. Aligned sequences in the fasta format output by dialign; only vertically aligned uppercase letters are assumed to have originated from a common ancestor. The dashes are inserted to align the uppercase letters; lowercase letters are not aligned and may be moved through adjacent dashes, for example the “atag” in the last line can be moved before the preceding dashes adjacent to the “ggt”, if one wants to place a window at those sites; but the subsequent uppercase letters cannot be moved. Two legitimate windows (solid borders) are shown, one encompassing all three sequences, the other encompassing two of the three. In addition, an illegitimate window is shown (dashed border) – illegitimate because it contains a deletion in a conserved block, which violates our assumption that a motif in a conserved block must be found in all species

The multiple alignment defines in this way the space of possible windows representing binding sites. These sites are sampled uniformly: a window spanning multiple sequences in an aligned block is sampled as often as a single-sequence window. A “configuration” is a particular choice of selected windows representing binding sites.

The “score” of a configuration is the probability that all the windows in that configuration were drawn from the same weight matrix, divided by the probability that all of them were drawn from a background model. (Alternatively, one could use the probability that these windows were sampled from the weight matrix, multiplied by the probability that all sites not in these windows were sampled from the background – this gives the probability of drawing the entire sequence given the current configuration of windows. It is convenient, however, to normalise this by dividing by the probability that the entire sequence was sampled from the background with no weight matrices; this gives our score.) The Gibbs sampler samples for this score; high-scoring configurations represent likely locations for binding sites.

First we describe the score for single-sequence (phylogenetically independent) windows: For a given w, the probability that the windows in a configuration C were all sampled from w is

$$P(C|w) = \prod_{i=1}^{N} \prod_{n=1}^{L} w_{\alpha_{i,n} n} \tag{1}$$

where the i'th motif has base $\alpha_{i,n}$ at position n. Since we don't know the weight matrix, we integrate over the space of all possible weight matrices (that is, over each component $w_{\alpha n}$ with $0 \leq w_{\alpha n} \leq 1$ and $\sum_{\alpha=A,C,G,T} w_{\alpha n} = 1$). This integral can be done exactly:

$$\int_w \prod_\alpha w_\alpha^{n_\alpha} = \frac{3! \prod_\alpha n_\alpha!}{(N+3)!}$$

where $N = \sum n_\alpha$ is the total number of windows and n_α is the 'base count' of base α, that is, the number of windows where base α appears at that position. Alternatively one can include a "prior probability" for weight matrices: $P(C) = \int_w P(C|w)P(w)dw$. With a suitable choice of $P(w)$ [11] this approach is equivalent to that of Liu et al. [7].

The probability that these windows were sampled from a background model is given by eq. (1) with background probabilities b replacing the weight matrix elements w. No integral is required since background probabilities are known.

For multi-sequence aligned windows, we modify the scoring for the window, assuming (irrespective of whether bases in it are uppercase or lowercase) that the bases in it, whether sampled from a weight matrix or from a background model, did not arise independently but evolved from a common ancestor.

We assume a "star" topology, with all species descending from a common ancestor; more general treatments along the same lines are possible, but complicated. Thus, looking at one column, each base in that column is a descendant of an ancestral base a. Assuming mutation rates m_i and assuming divergence a time t ago, the probability that a base in the i'th descendant is unmutated is $e^{-m_i t} = \mu_i$. The probability that it is mutated is $1 - \mu_i$. After mutation, once selection has operated and fixation occurred, the base is again represented by a sample from the same weight matrix (since the protein is generally unchanged in such close relatives). These considerations give us the following expression for the probability $P(W|w)$ that these bases in one column of the window W, descended from a common ancestor, were sampled from the same weight matrix, in terms of a "transition probability" $T(\alpha_i|a)$ that the base α_i evolved from an ancestor a [12]:

$$T(\alpha_i|a, \mu_i) = [\delta_{a\alpha_i}\mu_i + (1-\mu_i)w_{\alpha_i}] \tag{2}$$

$$P(W|w) = \sum_{a=A,C,G,T} w_a \prod_{i=1}^{N} T(\alpha_i|a, \mu_i) \tag{3}$$

(position index n omitted for simplicity). The full probability is a product of such factors over all columns. The expression is generally plausible and has the correct limits for $\mu \to 0$ (infinitely diverged species, i.e. independent sequences) and for $\mu \to 1$ (zero divergence, i.e. identical species). Note also that the transition matrix $T(\alpha_i|a)$ has the correct multiplicative property if one inserts an intermediate unknown ancestor: $\sum_b T(\alpha_i|b, \mu_1)T(b|a, \mu_2) = T(\alpha_i|a, \mu_1\mu_2)$. For further discussion, see reference [12].

For a set of windows, the probability that all these windows were sampled by the same matrix is given by a product of factors, one for each window, monomial as in eq. (1) for single-sequence windows and polynomial as in eq. (3) for multi-sequence windows. The total expression is thus likely to be a complicated polynomial, and is then integrated over all weight matrices w. (In practice, we use approximations for this integral, which we have verified are accurate.) Again, for the background scores, one substitutes w with background probabilities b and does not integrate.

A word about background probabilities: one commonly uses the raw "base counts" for each base. These are $1/4$ each in the simplest assumption, but A and T are more frequent in practice than C and G. We found it beneficial to instead use conditional probabilities for the occurrence of each base with the n preceding bases (in other words, assume a Markov model for the sequence.) That is, to calculate the background probability of the C in the sequence AGC, with two-neighbour correlation, we use

$$P(\mathrm{C}|\mathrm{AG}) = \frac{N(\mathrm{AGC})}{N(\mathrm{AGA}) + N(\mathrm{AGC}) + N(\mathrm{AGG}) + N(\mathrm{AGT})} \tag{4}$$

where N(AGC) is the actual number of occurrences of the string AGC in the sequence, etc. To calculate these numbers, it is preferable to use a larger dataset than the sequence of interest: for example, all sequenced intergenic regions in the organism, if available. If not enough sequence exists, a pseudocount of raw base counts may be added.

3 Implementation of the Sampler

With the above framework for scoring configurations of windows, we can implement the Gibbs sampler in a straightforward manner. We typically sample for multiple motifs at a time, which in effect means assigning different "colours" to the selected windows; each colour is scored separately and the total score is a product over all colours.

We start from a random configuration. At any instant in time the configuration is a set of selected windows, in different colours. The moveset takes us from one configuration to another, possibly changing the number of windows, the number of colours, or both, and is designed to satisfy "detailed balance" so that, in the long time limit, each configuration would be visited a fraction of the time proportional to its score.

The movesets we developed can follow several different strategies, with two key kinds of moves involved, which we call the "window move" and the "colour move":

1. We can restrict the total number of colours, and the total number of windows, rigidly. At each step we will "move" a window and optionally recolour it to one of the other existing colours: that is, we will pick a window at random, remove it, and sample from all new places where it can be placed, and all possible colours it can have at the new location. We call this the "window move". To preserve the number of colours while maintaining detailed balance, we require that the window being moved was not the only one in its colour; if not, we do nothing but increment the counter. This conserves total colour number and total window number, but not the number of windows per colour. It is also a highly optimal move in escaping local minima by "freeing" windows that are blocked by differently-coloured, suboptimally-placed windows.
2. We can change the number of windows, and the number of colours. To do this, we alternate the "window moves" with another kind of move which we call the "colour move". Here, we pick a window; if it is "blocked" (an overlapping window is coloured), we make no move, but increment the time counter (this is necessary

to preserve detailed balance); otherwise, sample from all possible colours it can be given, which may be the "null" or "background" colour (ie removing a window), one of the existing colours, or a new colour (if the window was already the only one in its colour, "new colour" means the same colour as earlier). This move conserves neither colour number nor window number, but preserves detailed balance. By itself, it does not do a good job of detecting motifs (though it ought to do that in the infinite-time limit, since it is ergodic and satisfies detailed balance); rather, its role is to expand the "colour space" and "window number space" of the system, while the "window moves" do the job of actually aligning motifs together.

This strategy of mixed moves will tend to populate the sequence with as many windows and as many colours as necessary to maximise the score, simply for entropic reasons (there are more states with a lot of windows than with a few windows). Thus, each configuration will be a "parse" of the sequence into similar motifs. A benefit is that one doesn't need to guess how many copies of a motif one is looking for: if good copies exist, they will be added to existing colours, otherwise new colours will be created. The disadvantages are that this strategy may "split" fuzzy motifs into several smaller groups, and the number of motifs it yields may be so large as to be unwieldy.

3. We can use the "colour moves" to allow a flexible colour/window number, but use a "chemical potential" to constrain the number of windows. Thus, every extra added window has a cost. Alternatively, we can introduce an entropic "correction factor" to take account of the fact that there are more states with $n + 1$ windows than with n windows (until n becomes large); the factor is easy to calculate for a single sequences, harder for multiple unrelated sequences and very hard for the phylogenetically aligned case. We can also place a rigid upper limit on the total number of colours.

To use the first strategy, we need initial values for N, C and the expected motif length L. It is fine to somewhat overestimate C and L, and advisable to somewhat underestimate N compared to the number of binding sites one actually expects to see. Typically one has one or two strong motifs, and the redundant colours while sampling then act as "buffers" to control the number of motifs actually picked by the sampler.

Regardless of strategy, in addition to the above moves, there are two other moves that we use to improve performance. A "global shift" move samples all possible shifts of an entire colour by a fixed amount; this is necessary because once the program finds a good placement of windows that is shifted relative to its optimal position, it is impossible to correct this by single-window shifts: the program will stay stuck in a local minimum. A "maskbit flip" move turns on sampling of mask bits for columns that indicate whether that column is to be scored or not; this is inadvisable for short motifs but is particularly useful for long motifs that have intervening fuzzy regions (such as occur in bacterial sequences), where it would be preferable not to score the fuzzy columns.

4 An "Anneal and Track" Strategy for Assessing Significance

In sampling for a long time, all configurations will be visited in the infinite-time limit with a frequency proportional to their score. To find the configuration with the best score, we can anneal (that is, raise the score to a power β representing a fictitious inverse temperature, and slowly raise β). If there is one overwhelmingly good configuration, a well-implemented anneal will typically find it easily; if there are several comparably good configurations (for example, more binding sites than one is searching for, or different motifs corresponding to different transcription factors), the anneal will randomly choose one of them.

Annealing in the framework of our moveset gives us a good candidate set of windows (binding sites); and we may also have candidate windows by comparing multiple anneals, or from other sources altogether; but it is desirable to assess their significance in some way. The approach below does this, and as a side benefit, also assigns significance to other sites which may not have been in our candidate set.

To do this, we set up a "labelled list" of the windows that we want to track. We could even up several such "labelled lists", with different labels A, B, $C\ldots$, and track them simultaneously. Then we sample for a long time without annealing ($\beta = 1$), doing the following:

1. At each time step, there is a set of colours each with a set of selected windows. For each labelled list A, we associate *one* of the current colours with that labelled list. Unless we are exceptionally lucky, none of the current colours will precisely match the labelled list A: there will be windows in that colour that are not in the list, or vice versa. So

   (a) We examine each colour for windows from our labelled list, with all possible consistent shifts and orientations;

   (b) For each colour and shift, we note the windows from the labelled list that appear in that colour with that shift (all these windows must have the *same* shift, or opposite shift if the orientation is opposite);

   (c) We calculate a total "importance score" of these windows to that colour by totalling the "cost" of removing each of these windows from the colour (that is, the ratio of the score of the colour with that window, to the score of the colour without that window);

   (d) Finally, we choose the colour and shift that gained the highest "importance score" by the above definition. The chances of this importance score being degenerate are negligible, but if it happens, we can make an arbitrary choice.

   This defines, at every instant, for each labelled list A, a unique colour associated with it, which we call $C(A)$, and an associated global shift, $S(A)$. This is the best match we have to our labelled list in the current configuration.

2. For every window w in the sequences being sampled, whether in the labelled lists or not, we maintain a set of counters, one for each labelled list, $N(w, A)$. This is an $N_w \times N_l$ matrix, where N_w is the number of windows and N_l the number of labels.

3. At each time step, for each label A, we go through the windows in the corresponding colour $C(A)$, shift each window w by $-S(A)$ to align properly with the labelled list if $S(A) \neq 0$, call the shifted window w', and increment the corresponding counter $N(w', A)$.

Finally, we divide each counter by the total number of timesteps; this gives, for each window w and each label A, a time average of the function

$$\begin{aligned} f(A,w) &= 1 \text{ if } w \in C(A) \\ &= 0 \text{ otherwise} \end{aligned}$$

which measures how often the window w was "co-clustered" with the labelled list A. In the infinite-time limit, this is (since each state is visited a fraction of time proportional to its score)

$$\frac{T(\text{co-clustered})}{T(\text{total})} = \frac{\sum_{S \text{ where w co-clustered with A}} P(S)}{\sum_{\text{all } S} P(S)} \tag{5}$$

which is intuitively the probability that w was sampled from the same weight-matrix as the windows in the labelled list A.

Thus, for each label A, on sorting the corresponding row of $N(w,A)$ we get a list of windows ordered by the probability that they "belong" with the list A.

This turns out to be a very useful strategy, not only in finding unknown motifs, but in assessing known ones: for example, one can "seed" a sequence with one or two short sequences of known motifs, and track the known motifs to see who else gets clustered with them.

5 Performance of the Sampler

We have tested the sampler both on synthetic data generated according to the model we assume, and on actual genomic data from the five closely related yeast species *S. cerevisiae, S. paradoxus, S. bayanus, S. mikatae, S. kudriavzveii* using genes with known motifs.

The "tracking" mechanism described earlier, apart from being a useful tool in practice for significance estimates of found motifs, is also a useful benchmarking tool when looking for known motifs: tracking numbers are a measure, in a quantitative and directly relevant way, of the probability for each tracked motif site that it was drawn from the same weight matrix as the others.

On synthetic data, for purposes of benchmarking, we track the known positions of the weight matrices. In other words, we sample for a long time, and collect statistics on what fraction of the time the known motif sites actually hang together. (Thus, we are not benchmarking the anneal, which is the motif-finding step; however, sites that hang together on sampling are always found in an anneal, though the reverse is not true.)

All tests were with five sequences, generated from a single ancestral sequence of length 500bp and a certain number n of embedded copies of a motif described by a weight matrix; the polarisation of the weight matrix, the number n of embedded copies, and the phylogenetic conservation probability μ were varied.

Phylogibbs results are clearly and consistently better than results with the sampler without considering phylogeny (treating the sequences as independent), except when the phylogenetic conservation probability μ is so high that it approaches the weight matrix polarisation. Results are in figures 3 and 4.

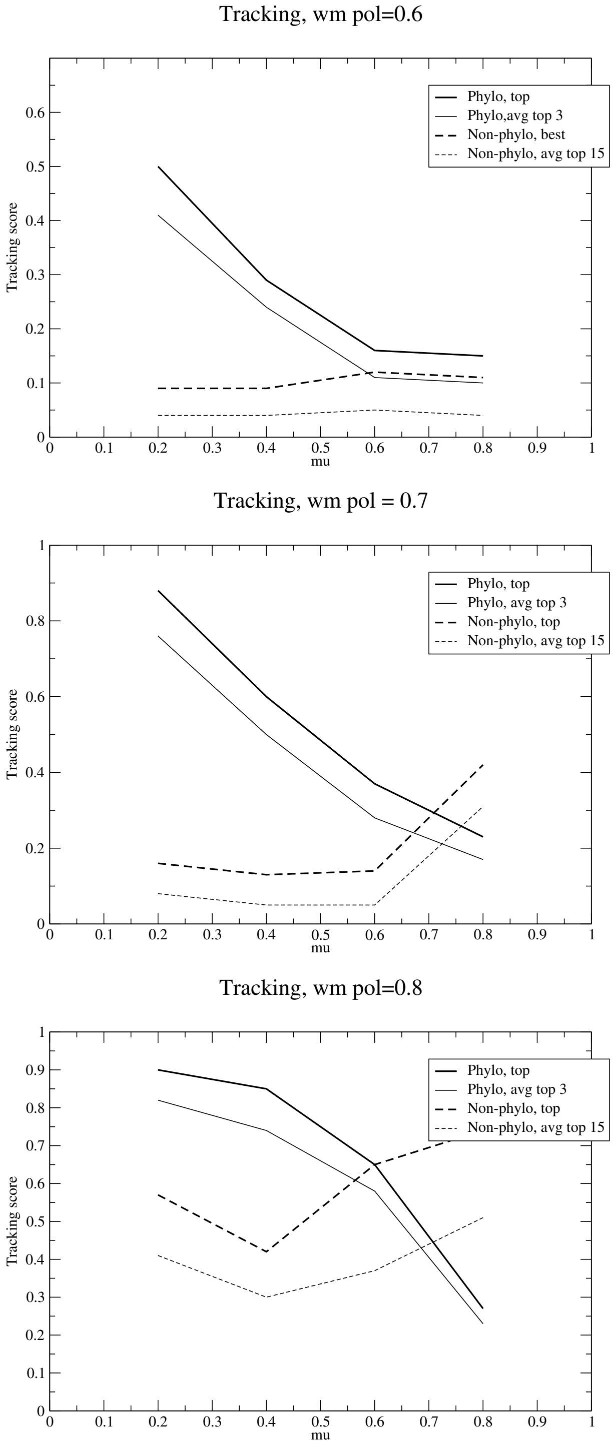

Fig. 3. Plots showing the fraction of time the Gibbs sampler clusters known motifs together, for five sequences of length 500 each descended from an ancestor, with five embedded copies of a motif represented by a weight matrix of polarisation as indicated, mutated according to our model with conservation probability μ. The known positions of the embedded weight matrices were tracked. Shown are the tracking scores of the best motif ("top") and the best 3 (phylo) or 15 (non-phylo) motifs ("avg"), each averaged over three runs, as a function of μ. Except when μ is high enough (0.8) to compare with or exceed the polarisation, phylogibbs performs clearly better. In particular, even when one or a few motifs may get comparable tracking scores in the absence of phylogeny, the average tracking score (here averaged for the top 3 motifs with phylogeny, or $3 \times 5 = 15$ motifs without) is far better with phylogeny. (The top 3, or 15, were averaged rather than all 5, or 25, because in many cases the tracking scores of the remainder in the non-phylogibbs case fell below the reporting threshold for the program)

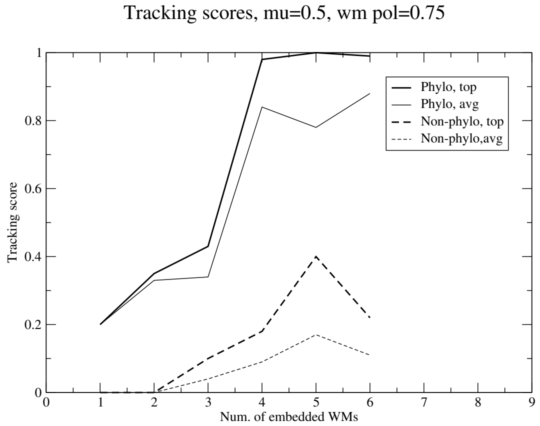

Fig. 4. Tracking scores, with and without phylogeny, for synthetic data with $\mu = 0.5$, and n weight matrices embedded, with wm polarisation 0.75. Shown are the best tracking score ("top"), and the tracking score averaged over all motifs ("avg"), as a function of n

On real data, this is usually the case, too; sometimes, when motifs are very numerous and prominent, both versions perform very well, and occasionally phylogibbs does worse, apparently because known motifs lie in conserved regions but are mutated in other species. We studied a few genes from *S. cerevisiae*, with well-documented regulatory sites, that we were interested in for other reasons. We assume a uniform conservation probability $\mu = 0.5$ (varying μ from around 0.3 to 0.7 does not make a big difference to results; $\mu = 0.3$ is a reasonable estimate from synonymous substitution in coding regions, but in practice *cis*-regulatory regions seem to have a somewhat higher conservation rate.) The anneal stage searched for 4 different colours, and 16 possible regulatory sites (avg: 4 per colour), in the phylogibbs case. Because on average 70%–80% of the *cis*-regulatory region seems to fall in aligned blocks, in the non-phylogibbs case we searched for 64 possible sites (avg: 16 per colour). The results are as follows:

- CLN3 (YAL040C): There are four well-defined copies of an element that has been called the "daughter delay element" [13], which has been implicated in the delay in budding in daughter cells. This element has consensus CCWYWGCATTTC and is instantly picked up by phylogibbs with tracking score 1.00. However, without incorporating phylogeny this motif is often not picked up at all in the initial anneal, and when it is picked up it gets a lower tracking score. Apparently this is because there are several copies of similar motifs such as CCWWW... (half of the MCM1 dimer site, which appears in several places upstream of this gene), CCNNNGC, and SSATTTC, some of which have neighbouring sequence similar to the DDE, and these tend to lead the sampler astray.
- HO (YDL227C): We used the first 1000 bp upstream region of this gene, though it has one of the longest *cis*-regulatory regions in cerevisiae. With or without phylogeny we retrieve numerous copies of the SBF binding site [15] (consensus: CACGAAA) with tracking score 1.00 for numerous copies; the MATα2 site TTACATCA is also retrieved with tracking score 1.00 with phylogeny, but without phylogeny our runs did not retrieve this motif.

- CLB1 (YGR108W): This gene is known [14] to be regulated by Ndt80 and contains the middle sporulation element (MSE) motif GWCACAAA in its *cis*-regulatory region, but not very strongly. We recover the motif with phylogibbs, but with tracking scores of 0.40-0.50. Without phylogibbs, the anneal yields ambiguous results (the motif is mixed up with other sites) and the tracking yields nothing above the reporting threshold (0.05).
- NDT80 (YHR124W): This is a key gene in sporulation [14, 16, 17] and contains the MSE in its *cis*-regulatory region; the MSE is bound by the Ndt80 gene product itself, as well as by Sum1. We find the motif with or without phylogeny, but in this case the performance is better without phylogeny: tracking scores are around 0.99 without, or 0.6 with. (However, if we use phylogibbs but lower μ to zero – that is, independent sequences but aligned – performance improves to perfect levels: that is, all copies of the motif are found with tracking score 1.00.)

The NDT80 example brings up one worthwhile point: the improvement in performance of phylogibbs comes partly from improved scoring of phylogenetically related sequences, but a significant part of the improvement is merely the much smaller state space when one aligns sequences as we do before sampling. There is a significant reduction in entropy: many configurations that are not likely positions of binding sites are simply removed from the state space. (One can, of course, contrive examples where the reduced state space hurts rather than helps because important configurations are being removed, for example because most motifs occur in conserved blocks but are mutated in all species but one; it's unlikely this is a common problem in practice.)

Availability of the Code

The code is available for download on
`http://www.physics.rockefeller.edu/~siggia/software/phylogibbs/`

Support

This work was supported by NSF grant DMR 0129848.

References

1. Cliften, P., Sudarsanam, P., Desikan, A., Fulton, L., Fulton, B., Majors J., Waterston R., Cohen B. A., Johnston M. Science **301** (2003) 71–6.
2. Kellis M. Patterson N., Endrizzi M., Birren B., Lander E. S. Nature **423** (2003) 241–54.
3. Dermitzakis E. T., Bergman C. M., Clark A. G. Mol Biol Evol. **20**(5) (2003) 703–14.
4. Emberly E., Rajewsky N., Siggia E. D. BMC Bioinformatics **4**(1)(2003) 57.
5. Liu J. S.: *Monte Carlo Strategies in Scientific Computing*. Springer-Verlag (2001).
6. Lawrence C. E. , Altschul S. F., Bogouski S. F., Liu J. S. , Neuwald, A. F. , Wooten, J. C. Science **262** (1993) 208–214.
7. Liu J. S., Neuwald A. F., Lawrence C. E. J. Amer. Stat. Assoc. **90** (1995), 1156–1170.
8. Durbin R., Eddy S., Krogh G., Mitchison G. *Biological Sequence Analysis*, Cambridge University Press (1998).
9. Thompson J. D., Higgins D. G., Gibson T. J. Nucleic Acids Res. **22** (1994) 4673–4680.

10. Morgenstern, B. Bioinformatics **15** (1999), 211–218.
11. van Nimwegen E., Zavolan M., Rajewsky N., Siggia E. D. Proc. Nat. Acad. Sci. **99** (2001) 7323–7328.
12. Sinha S., van Nimwegen E., Siggia E. D., Proceedings of the 11th international conference on Intelligent Systems for Molecular Biology (2003).
13. Laabs T. L., Markwardt D. D, Slattery M. G, Newcomb L. L., Stillman D. J., Heideman W. Proc Natl Acad Sci USA. **100**(18) (2003) 10275–80.
14. Chu S., Herskowitz I. Mol Cell **1** (1998), 685–696.
15. Breeden L., Nasmyth K. Cell **48** (1987) 389–97.
16. Hepworth S. R., Friesen H., Segall J. Mol Cell Biol. **18** (1998) 5750–61.
17. Chu S., DeRisi J., Eisen M., Mulholland J., Botstein D., Brown P. O., Herskowitz I. Science **282** (1998) 1421.