THiCweed: fast, sensitive motif finding by clustering big data sets

Ankit Agrawal[1], Leelavati Narlikar[2], Rahul Siddharthan[1*]

[1] *The Institute of Mathematical Sciences (HBNI), Chennai 600113, India*

[2] *CSIR National Chemical Laboratory, Pune 411008, India*

*Corresponding author. Author list is alphabetical.

January 30, 2017

Abstract

Motivation: Thousands of ChIP-seq (chromatin immunoprecipitation followed by sequencing) datasets are now publicly available that provide genome-wide binding profiles for hundreds of transcription factors in various species and various cell types, with thousands to hundreds of thousands of "peaks" per dataset. Transcription factors commonly bind to regions of DNA that have short conserved sequence patterns, or "motifs". *Ab initio* motif finding is a well-established problem in computational biology, but such large data sets are challenging for most existing tools. Additionally, it is common for the target proteins to bind indirectly to DNA via co-factors, with the result that the ChIP-seq peaks contain a mixture of motifs. Few tools exist to deal effectively with this problem.

Results: We describe a new approach to motif finding that models the problem as one of clustering bound regions based on sequence similarity. We take an iterative "top-down" approach of repeatedly subdividing an initial single large cluster of all input sequences into smaller and smaller clusters, while also exploring shift and reverse-strand matches of sequences to clusters. Our implementation is significantly faster than any other ChIP-seq-oriented motif-finding program we tested, able to process 5,000 sequences of 100bp length in a few minutes, or 30,000 sequences in 1-2 hours, on a desktop computer using a single CPU core. On synthetic data it outperforms all programs except one (MuMoD) on accuracy; compared to MuMoD it is somewhat less accurate but orders of magnitude faster. It is designed to perform well with "window" sizes much larger than the length of a typical binding site (7-15 base pairs), and we commonly run it with window sizes of 50bp or more. On actual genomic data it successfully recovers literature motifs, but also uncovers highly complex sequence characteristics in flanking DNA, and in many cases recovers secondary motifs (of possible, and sometimes documented, biological significance) even when they occur in less than 5% of the input sequences. We suggest that this is a powerful new approach to the analysis of ChIP-seq data.

Availability: The software is open source and available at `http://www.imsc.res.in/~rsidd/thicweed/` under the two-clause BSD license.

Contact: `rsidd@imsc.res.in`

1 Introduction

Chromatin immunoprecipitation followed by sequencing (ChIP-seq) (Johnson *et al.*, 2007) is a widely-used assay for determining transcription factor binding sites (TFBS) *in vivo*. By crosslinking the in-vivo DNA-protein complexes using formaldehyde, sonicating to break the DNA, precipitating the protein of interest using a specific antibody, reversing the crosslinks, sequencing the DNA fragments and mapping them to a reference genome, a genome-wide map of TFBS with a resolution of 100-200bp can be obtained. Newer variants like ChIP-exo (Rhee and Pugh, 2011) and ChIP-nexus (He *et al.*, 2015), which promise even higher resolution, are gaining popularity. Typically these assays yield hundreds or, in large genomes, thousands to hundreds of thousands of binding sites per factor per cell type (ENCODE Project Consortium *et al.*, 2012; Sloan *et al.*, 2016).

TFBS are generally characterised by short conserved patterns or "motifs" in the DNA sequence, commonly represented by "position weight matrices" or "position specific scoring matrices" (Stormo and Hartzell, 1989; Hertz *et al.*, 1990), a probabilistic representation where each position within a binding site is described by an independent categorical distribution over the four nucleotides. A key bioinformatic task is to identify these motifs, but *ab initio* motif detection using traditional tools such as MEME (Bailey and Elkan, 1994) and Gibbs samplers such as AlignACE (Roth *et al.*, 1998; Hughes *et al.*, 2000) and PhyloGibbs (Siddharthan *et al.*, 2005; Siddharthan, 2008) is a challenge on such large datasets. Additionally, it is common for factors to interact with DNA via co-factors and not directly, which means a mixture of different motifs may be found in the ChIP-seq data.

A previous program by one of us, MuMoD (Narlikar, 2013), was targeted at the second of these problems: it simultaneously and sensitively finds multiple motifs in a given dataset. Other programs such as Chipmunk (Kulakovskiy *et al.*, 2010, 2013; Levitsky *et al.*, 2014), Meme-Chip (Machanick and Bailey, 2011) and Weeder (Pavesi *et al.*, 2004; Zambelli *et al.*,

2014) find successive motifs sequentially, masking previously identified sites or sequences to find the next motif. But a significant prior knowledge of what to expect from the data, or multiple iterations with varying parameters and human scrutiny of the results, is required for effective usage of these programs.

The program we describe here, THiCweed, offers both speed and accuracy in finding multiple motifs in large datasets. It does not require prior information on the number of motifs or the lengths of the motif, since its approach is based on clustering rather than traditional motif-finding, and the clustering is based on stringent statistical criteria. On synthetic data, we show that it outperforms all current alternatives greatly on speed and is close to the best current alternative in terms of accuracy. On real genomic data, it reveals an unusual complexity in the architecture of sequence motifs, in particular in internal dependencies and flanking sequence.

2 Methods

There are two components to our approach:

- We describe an efficient heuristic method of clustering sequences based on hierarchically splitting larger clusters into smaller clusters. Thus, starting with one large cluster, we split it in two clusters (or three, the third consisting of poor matches to either cluster). The scoring is described below, and is based on the likelihood ratio of a sequence belonging to one or the other cluster, done iteratively starting from an initial heuristic split. We then split each new cluster into two (or three) further clusters; and proceed until no further splits are possible. For each split we apply stringent statistical criteria to accept or reject the split. There are some further optimizations described below in "Algorithm".

- In forming clusters, we optionally include shifts and reverse complements of individual sequences to find optimal clusters. This, it turns out, constitutes an effective and fast implementation of an *ab initio* motif finder on large ChIP-seq data sets, in addition to detecting the variations in motif and sequence context alluded to in the previous point. Alternatively, THiCweed can be used on sequences that have been previously aligned by a "feature" (motif) to discover additional motifs/complexities and without considering shifts and reverse complements, similar to the program NPLB (Narlikar, 2014; Mitra and Narlikar, 2016), but we don't discuss this use here.

Our clustering, based on repeated splitting of a single large cluster into a hierarchy of clusters, is in contrast to the usual hierarchial clustering, where individual data points are formed into clusters, and which requires $O(N^3)$ or at best $O(N^2 \log N)$ time for N data points. We call our approach "Top-down hierarchical clustering"; and since its purpose is to weed out "signals" in chipseq peaks, we call the program "THiCweed". (We considered "THC-weed" but it may confuse search engines.)

2.1 Algorithm

2.1.1 Top-down hierarchical clustering

We take the simpler case of input data that has been pre-aligned with all sequences of the same length, where we don't consider shifts and reverse-complements of sequences.

The algorithm is:

1. Initialize with one cluster containing all sequences.

2. For every current cluster C (ie, initially just one cluster), "split" into two clusters $C_1$and C_2, using scoring and significance criteria described below. Sequences not consistently clustering with either C_1 or C_2 are concatenated into a third cluster C_p. In each round, all these unclustered sequences from each division are concatenated into one cluster.

3. After every two iterations of step (2), if the current state has more than two clusters, reassign the poor-scoring sequences (sequences whose likelihoods in their current cluster are low) to the "best" available cluster. This appears to improve convergence to meaningful motifs.

4. Repeat from (2), until no new clusters are formed and no reassignments are made.

A typical flow through this algorithm may be:

1. Initialize the input data into one cluster C with N input sequences

2. Split C into three clusters C_1, C_2 and C_p (the last cluster contains sequences that do not cluster consistently)

3. Apply step 2 to each of the current clusters, but concatenate all poor-quality clusters into one single cluster). This may yield five clusters C_{11}, C_{12}, C_{21}, C_{22}, and C_p.

4. Take the poorest-scoring fifth of sequences and reassign them to the best cluster available. Repeat until no reassignments occur, then

   - If any reassignments occurred OR any successful splits had occurred in the previous step, return to step 2.
   - If no reassignments at all occurred in the previous step AND no splits had occurred in the step prior to that, return current clusters and exit.

5. Suppose only one successful split occurs in step 2: C_{11} splits, resulting now in six clusters C_{111}, C_{112}, C_{12}, C_{21}, C_{22}, C_p, none of which are split in step 3.

6. Also, no reassignments occur in step 4.

7. Then this is the final clustering returned by the program.

2.1.2 Scoring

Let each sequence have length L (the requirement for equal length is relaxed below). Consider a cluster C with N sequences in it, $S^1, S^2, \ldots, S^N$. The probability of seeing this data if all these sequences were drawn from the same PWM model is

$$P(C) = \prod_{i=1}^{L} \frac{\prod_\alpha \Gamma(n_{i\alpha}+c)\Gamma(4c)}{\Gamma(\sum_\alpha n_{i\alpha}+4c)\Gamma(c)^4} \quad (1)$$

where $n_{i\alpha}$ is the number of times nucleotide α appears in column i, and c is a pseudocount (0.5 by default). If the cluster contains a single sequence, this expression reduces to $(\frac{1}{4})^L$.

The likelihood that a sequence S is sampled from the same PWM as sequences in a cluster C that contains N seqs is

$$P(S|C) = \frac{P(S.C)}{P(C)} = \prod_{i=1}^{L} \frac{n_{iS_i}+c}{N+4c}$$

where $n_{i\alpha}$ is the number of occurrences of nucleotide α at position i in the cluster and S_i indicates the i'th nucleotide in the sequence S. This is implemented by the function `seqLikelihood`.

When splitting a cluster, as implemented in the `splitCluster!` function, an initial split is made by ranking each sequence by its likelihood of belonging to that cluster, and moving the "best" 25% to another cluster. Then sequences are selected in random order, removed from their current cluster and re-assigned to the most likely cluster, until no further reassignments are made.

This function is wrapped by a function `splitCluster_rand!` that assesses the significance of a split and accepts or rejects it. `splitCluster!` is called four times, resulting in four pairs of clusters, and we demand that at least three of the six pairwise cluster comparisons that result have an adjusted Rand index of greater than some threshold (by default 0.66). (If the three pairwise comparisons from the first three splits already exceeds this threshold, the fourth is not carried out.) Additionally, we demand that the ratio of the likelihoods of the two clusters post the split, to the likelihood of the unsplit cluster, as calculated from equation 1, exceed a threshold. We determine this threshold as follows: suppose the two clusters consisted of random sequences, and were split on a single position – say, one cluster contained only A or C in that position, the other only G or T – while the nucleotides at all other positions are evenly distributed. Call the log likelihood ratio of this LLR_1. This is not a significant split (it is always possible to do this, or better, for any cluster). However, if they differed in this manner in two positions – one cluster contained only AA/CC/AC/CA, the other only GG/TT/GT/TG – this would be significant. Call the log likelihood ratio here LLR_2. We demand that the LLR we observe of the splits be greater than $LLR_1 + 0.1(LLR_2 - LLR_1)$. Both LLR_1 and LLR_2 can be calculated quickly using equation 1.

If these two significance criteria are not both met, `splitCluster_rand!` rejects the split and returns the unsplit cluster. If they are met, it accepts the split, but moves sequences that were not consistently clustered during the multiple splits into a third cluster, and returns the resulting three clusters. We find in practice that these criteria suffice to avoid "spurious" splitting of synthetic sequence: THiCweed very seldom finds more clusters in the data than have been put in, though it may fail to find genuine clusters if the motifs are weak.

When reassigning sequences (step 3 of the algorithm), we consider the poorest 20% of the sequences (measured by their likelihoods in their current clusters). For each sequence, we first remove it from its current cluster, then calculate $P(C')$ for each available cluster C where $C' = C + S$, using the above formula, and add it to the best cluster.

2.1.3 Shifts and reverse complements

The algorithm remains the same as above except that we do not cluster entire sequences, but fixed-size "windows" within those sequences. Typically these windows are long: by default two-thirds the median length of sequences in the set. When splitting clusters and reassigning sequences, all possible shift positions and both possible strands are considered. The window is allowed to shift out of the sequence by up to an amount equal to half its length (ie, only partially covering the sequence); nucleotides that fall outside the known sequence are treated as N, ie unknown.

2.2 Benchmarking: synthetic data

We generated synthetic datasets consisting of sequences of length 100bp each, with motifs drawn from random PWMs placed within the central 20bp of these sequences, and otherwise random (each nucleotide having probability 0.25). The PWMs had columns sampled from Dirichlet distributions with uniform hyperparameter c (ie, each column v of each PWM was independently sampled from the distribution $P(v) \propto v_\alpha^{c-1}$). c was varied across datasets. Each sequence contained one motif, and each set contained motifs drawn from a small number of PWMs. THiCweed and four other programs (MuMoD (Narlikar, 2013), Chipmunk (Kulakovskiy *et al.*, 2010, 2013; Levitsky *et al.*, 2014), Meme-Chip (Machanick and Bailey, 2011) and Weeder2 (Pavesi *et al.*, 2004; Zambelli *et al.*, 2014) were run on these sets, in multiple-motif ZOOPS mode (zero or one occurrences of a motif per sequence). The "known" clustering of the set was the assignment of sequences to PWMs, and the "predicted" clustering for each program was the assignment of sequences to predicted motifs. The known and predicted clusters were compared using the "adjusted Rand index" ("ARI") (Hubert and Arabie, 1985) and the results plotted as a function of c. Higher ARI indicates a better match between the clusterings, with 1.0 indicating perfect agreement and 0.0 being the value expected by chance.

Three datasets are shown here. In dataset 1, for each value of c, ten files are generated each with 1000 sequences, each containing one of three motifs of length 12 each. The distribution of motifs in the sequence sets are 0.4, 0.3, 0.3. For each c, the average ARI for clustering runs across the ten files is reported. Dataset 2 is similar but with 5000 sequences per file. In dataset 3, the number of PWMs is randomly chosen between 1 and 5

in each file, and the length of each PWM is a random number between 7 and 15; in case of two PWMs, the ratio is 0.6:0.4 and in case of more than two, the first PWM occurs 40% of the time with the remainder occurring equally often. There are 1000 sequences per file and 20 files per value of c.

Commandline options:

THiCweed: no additional parameters

MuMoD: In dataset 1 and 2 "i", motif length 12 and number of "modes' 3 were specified. In dataset 2 "d" and 3, default parameters were used.

ChipMunk: In all runs, the correct number of motifs was specified. In datasets 1 and 2, the motif length range was given as 12:12, in dataset 3, as 7:15 (that is, corresponding to the actual lengths in each case).

Weeder: Default options, but with a background frequency model derived from synthetic data.

Meme-Chip (meme): dreme was disabled with "`-dreme-m 0`", with default parameters otherwise.

Meme-Chip (dreme): meme was disabled with "`-meme-nmotifs 0`"

Other notes

Despite the "filter" keyword used in the command line, Chipmunk sometimes predicts multiple motifs per sequence because it searchs for matches for predicted motifs in all sequences. For computing the adjusted Rand index, each sequence was classified to the best-matching motif, as per the score reported by Chipmunk.

2.3 Real genomic data

Here we used data from the ENCODE project (ENCODE Project Consortium *et al.*, 2012; Landt *et al.*, 2012; Sloan *et al.*, 2016), consisting of ChIP-seq peaks. We report results for ATF1, MAFF and RAD21 as representative examples. We focussed on one cell type each, took 50bp flanking sequence about each peak location, merged across replicates in that cell type, with the central 100bp taken in merged regions. Finally, "low complexity" repetitive regions were rejected for the purposes of this work (though binding there could be meaningful biologically).

Narrowpeak files were obtained from the ENCODE project website `http://www.encodeproject.org` under experiment IDs ENCSR000DNZ (ATF1, cell type K562);ENCSR000EGI (MAFF, cell type K562); and ENCSR000BMY and ENCSR000EAC (RAD21, cell type GM12878).

The resulting clusters were compared with nucleosome positioning data from ENCODE and and PhastCons (Hubisz *et al.*, 2010) phylogenetic conservation data (with other primates) from the UCSC genome site (Karolchik *et al.*, 2003), as well as distances from nearest transcriptional start sites (TSS), using custom R scripts. For TSS we used the refGene data from the hg19 release on the UCSC genome browser site.

3 Results

3.1 Synthetic data

Results for the three datasets described in Methods are plotted in Figure 1 parts A,B,C for $c = 0.1, 0.2, 0.3, 0.4, 0.5$ (smaller value of c corresponds to sharper motifs).

In all cases THiCweed was run with default parameters, and in particular, a "window size" of 66bp or two-thirds the median input sequence length. As noted, it is designed to be run with large window sizes on real genomic data. Also, the stringent criteria for splitting a cluster ensure that spurious clusters are unlikely, so setting the maximum number of clusters helps only marginally (not shown). At this point there is no option for a minimum number of clusters.

In dataset 3 and one run of dataset 2 (labeled "MuMod(d)") MuMoD was run with default parameters; in dataset 1 and the other run of dataset 2 (labeled "MuMod(i)") it was asked to look for three modes with length 12. In all cases MuMoD is the best performer, though in dataset 2 only the version with additional parameters outperforms THiCweed. Chipmunk (in ChipHorde mode) requires the exact number of motifs to be told to it, which was done in these cases, and the length (or range of lengths in the third set) of the motif was given; additionally, the centre of the sequences was weighted 0.5 with 0.1 for the remainder. Meme-Chip with its default options skips the MEME step on data sets of this size in favour of DREME, with inferior results. Enabling MEME improved the results, at a significant cost in running time. Weeder2 was run with default options but a background model derived from synthetic data, as described in Methods.

In dataset 1, which is the smallest and most "homogeneous" of the three data sets (in that each set has exactly three motifs, each of length 12bp, in the same proportions), THiCWeed is outperformed by MuMod and (marginally) by Meme-Chip(meme), and is comparable to Chipmunk. On the larger but similar dataset 2, however, it outperforms all programs except MuMoD(i). On the more varied dataset 3, where files may contain between 1 to 5 motifs with lengths varying from 7 to 15 base pairs, THiCweed is second only to MuMoD.

Importantly, while as measured by the ARI THiCweed places second to MuMod above, its running time is about two orders of magnitude less than MuMod (and also much faster than all other programs tested). On realistic ChIP-Seq data sets of tens of thousands of sequences, Meme-Chip drops the MEME step in favour of the heuristic DREME, with significantly inferior performance; and Weeder2 learns motifs from a small fraction of the sequences and uses those to analyse the rest (Zambelli *et al.*, 2014). MuMoD can take days to run on such datasets, but THiCweed handles them in an hour or two, with interesting and biologically relevant results described in the next section.

Figure 1 part (D) shows running times of all the programs tested, for synthetic input data consisting of 200, 400, 600, 800 and 1000 sequences each 1000bp long and containing two different motifs, each of length 10 sampled with Dirichlet parameter 0.2, in 60:40 proportion. Meme-Chip in MEME mode is an outlier: though its performance in accuracy is not very far

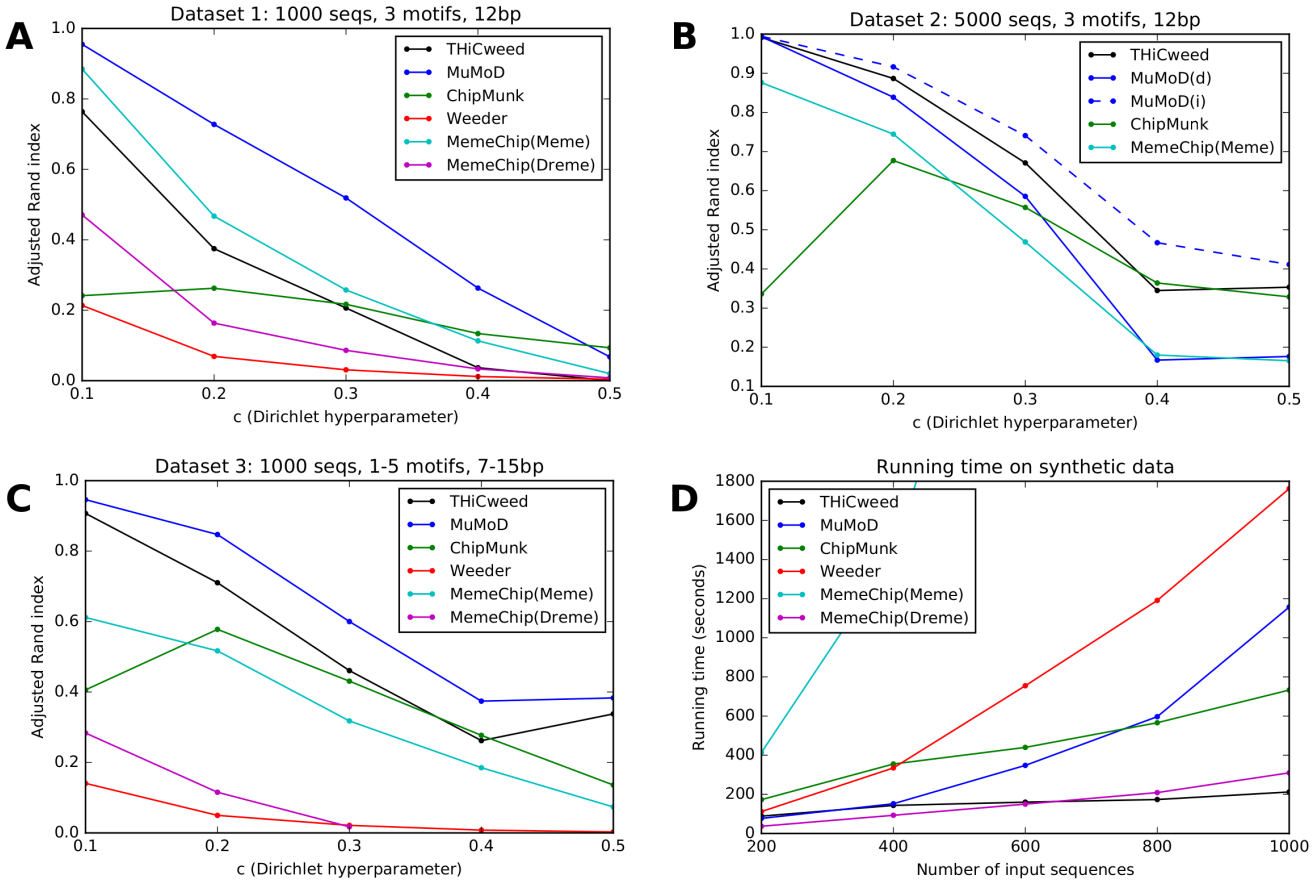

Figure 1: Parts A,B,C: Adjusted Rand index (higher is better) of predicted clustering to known clustering of synthetic data sets, containing motifs drawn from PWMs sampled columnwise from Dirichlet distributions with hyperparameter c. While competitive in all cases, THiCweed performs particularly well on the larger dataset, and on the dataset where the number of motifs and their lengths is variable. Part D: Running times of various programs on synthetic input data from 200 to 1000 sequences, each 100bp long. Numbers are totals over five runs on five different files.

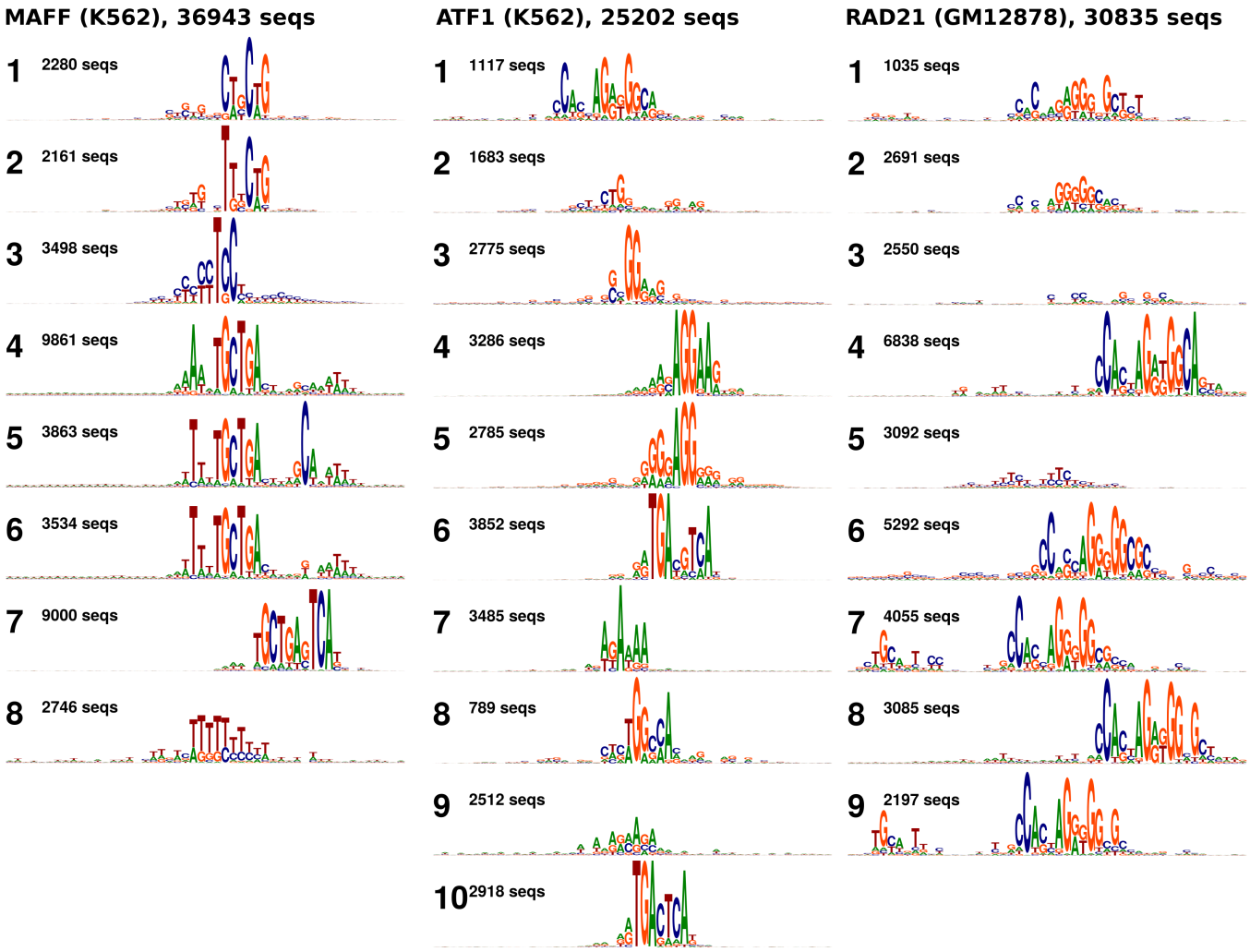

Figure 2: Results of clustering ChIP-seq peaks for three TFs using THiCweed. Nearly all clusters obtained are "informative" and, notably, many display significant "structure" in sequence flanking the core motifs, including the previously documented "M2" motif for CTCF (RAD21).

behind other programs, its running time would seem to disqualify it from serious work (and indeed it disables MEME by default for sequence sets larger than about 600×100bp). It appears that, of the other programs, Chipmunk and Meme-Chip (Dreme mode) have runtimes increasing roughly linearly with data size; MuMoD and Weeder running times increase superlinearly; and THiCweed's increase is somewhat sublinear. But this may be misleading because THiCweed's running time depends strongly on the input data, and larger numbers of clusters would entail a higher running time. Nevertheless, on real genomic data THiCweed's running time turns out to be a strong benefit compared to other programs.

3.2 ChIP-seq data from the ENCODE project

Running on actual genomic data yields a variety of different results depending on the factor being examined and the size of the dataset.

THiCweed has no "prior" knowledge of the number of different motif "architectures", but by default reports a maximum of 25. In some cases far fewer are reported. Because of the statistical criteria on splitting clusters that we use, described in Methods, we believe that large numbers of clusters, if produced, are statistically significant, but THiCweed can "recluster" the output into smaller numbers of clusters for ease of visualization, and this is done below. Also, it works with "window sizes" much larger than typical "motif lengths" that one considers; here we used 50bp.

Figure 2 shows three examples of motif output. In one case, MAFF (cell type K562), out of 36,943 sequences, the "core" motif TGCTGA appears in 26,258, but it is split into four motif "architectures" (no. 4,5,6,7) with very different flanking contexts. Somewhat similar motifs, CTBCTG and TTBCTG (where B means not-A), occur in architectures 1 and 2 (totally 4441 sequences). Architectures 3 and 8 contain other motifs, possibly caused by co-factors or indirect binding. In this case, THiCweed output only these 8 clusters.

In the second case, ATF1, out of 25,202 sequences, a "canonical" motif (JASPAR (Sandelin *et al.*, 2004; Mathelier *et al.*, 2016), Factorbook (Wang *et al.*, 2013)) TGACGTCA is recovered in a cluster of 3852 sequences, but a related motif, TGACTCA, appears in 2918 sequences: both consist of TGA and its reverse complement TCA, but the spacing is different. This is not reported by Factorbook or JASPAR for ATF1, but a similar motif is reported for ATF2. In addition, the motif for CTCF, a factor involved in a variety of roles including transcriptional activation and repression, insulator function, and chromatin remodelling (Phillips and Corces, 2009), appears in 1117 sequences. In addition, several additional AG-rich motifs are found that do not resemble previously documented motifs. In this case, THiCWeed reports 19 clusters which we reduced to 10 for ease of viewing.

In the third example, with 30835 sequences for RAD21 (a cohesin subunit) in cell type GM12878, the motif for CTCF, known to form complexes with cohesin (Millau and Gaudreau, 2011), is prominently picked up in 25193 sequences across seven architectures. Architectures 3 and 9 show a secondary motif, previously called the M2 motif (Schmidt *et al.*, 2012), prominently and architecture 1 and 4 show it faintly. In this case, THiCWeed outputs 24 architectures, which were reclustered into 9 for ease of viewing.

Finally, the "genomic backgrounds" in many cases appear to be biased: clusters 4, 5 and 6 of MAFF have AT-rich backgrounds, while cluster 6 of RAD21 has a GC-rich background, in regions extending far outside the core motifs. It is also comparatively GC-rich in the tail region of its core motif, ending with a strong CGC where other clusters often have CAC or CAG. We have noted such patterns in other ongoing work, to be explored in future.

3.2.1 Comparison with other programs

Figure 3 compares clustering results of THiCweed on 36,943 MAFF peaks with MuMoD and Chipmunk on a random selection of 10% of those peaks (3,641 peaks), the number being reduced due to inability of these programs to execute on the full set in reasonable time. All programs predict the dominant TGCTGA motif, but the differences in other predicted motifs are interesting: the surrounding context is missed (as in clusters 4,5,6 of the THiCweed clustering), and Chipmunk in particular predicts multiple motifs per sequence despite use of the "filter" option. MuMoD was run with default parameters and maximum number of modes 8; Chipmunk was asked to search for 6 motifs of length 8–20.

3.2.2 Biological relevance of these clusters

Are the motif differences across clusters significant, or an artifact of the clustering process? Based on our statistical criteria described in Methods, we feel they are significant. Further evidence arises from comparing other genomic features such as phylogenetic conservation (via PhastCons scores (Hubisz *et al.*, 2010) from the UCSC Genome Browser (Karolchik *et al.*, 2003)) and nucleosome occupancy (from ENCODE (ENCODE Project Consortium *et al.*, 2012)). Figure 4 compares each cluster for RAD21 with a plot of conservation, nucleosome occupancy (in an extended region of 1000bp on each side), and distance to the nearest TSS. It appears, for example, that cluster 6 is more likely to be found near TSS, suggesting that this GC-rich variant of the CTCF motif is possibly linked with transcriptional regulation. It also shows noticeably less inter-species conservation compared to, for example, cluster 4. Clusters 1, 2, and especially 3 show very weak CTCF motifs but this seems sufficient to ensure nucleosome positioning.

Figure 5 compares the ATF1 clusters with nucleosome positioning information. It is noteworthy that cluster 1, which shows the CTCF motif, is the only one to show a strong nucleosome positioning pattern, suggesting that this cluster genuinely contains CTCF binding sites and is not an artifact. Cluster 10 shows much stronger nucleosome positioning than cluster 6, despite the similarity in motifs (with only the gap being different). All these points suggest biological relevance to the distinct motifs found in these data.

4 Discussion

Motif-finding in large datasets produced by ChIP-seq and similar experiments is a qualitatively different problem in complexity from what traditional motif-finders are used to handle. Additionally, one could liken the problem of finding rarely-occurring motifs to finding a needle in a haystack. THiCweed, we have shown here, uses a novel clustering algorithm to comfortably handle datasets tens of thousands of sequences large, and with significant heterogeneity in motif information. It successfully picks up biologically relevant motifs, like the CTCF motif in the ATF1 data, that occur in fewer than 5% of the input sequences. Its large window size enables it to also picks up secondary motifs like the M2 CTCF motif in the RAD21 data and several peripheral motifs in the MAFF data that would be missed by conventional motif finders.

Uniquely among the programs we have tested, THiCweed achieves its combination of speed and accuracy without resorting to heuristics in scoring (as DREME and Chipmunk do, using regular expressions and "seeding" respectively) and with-

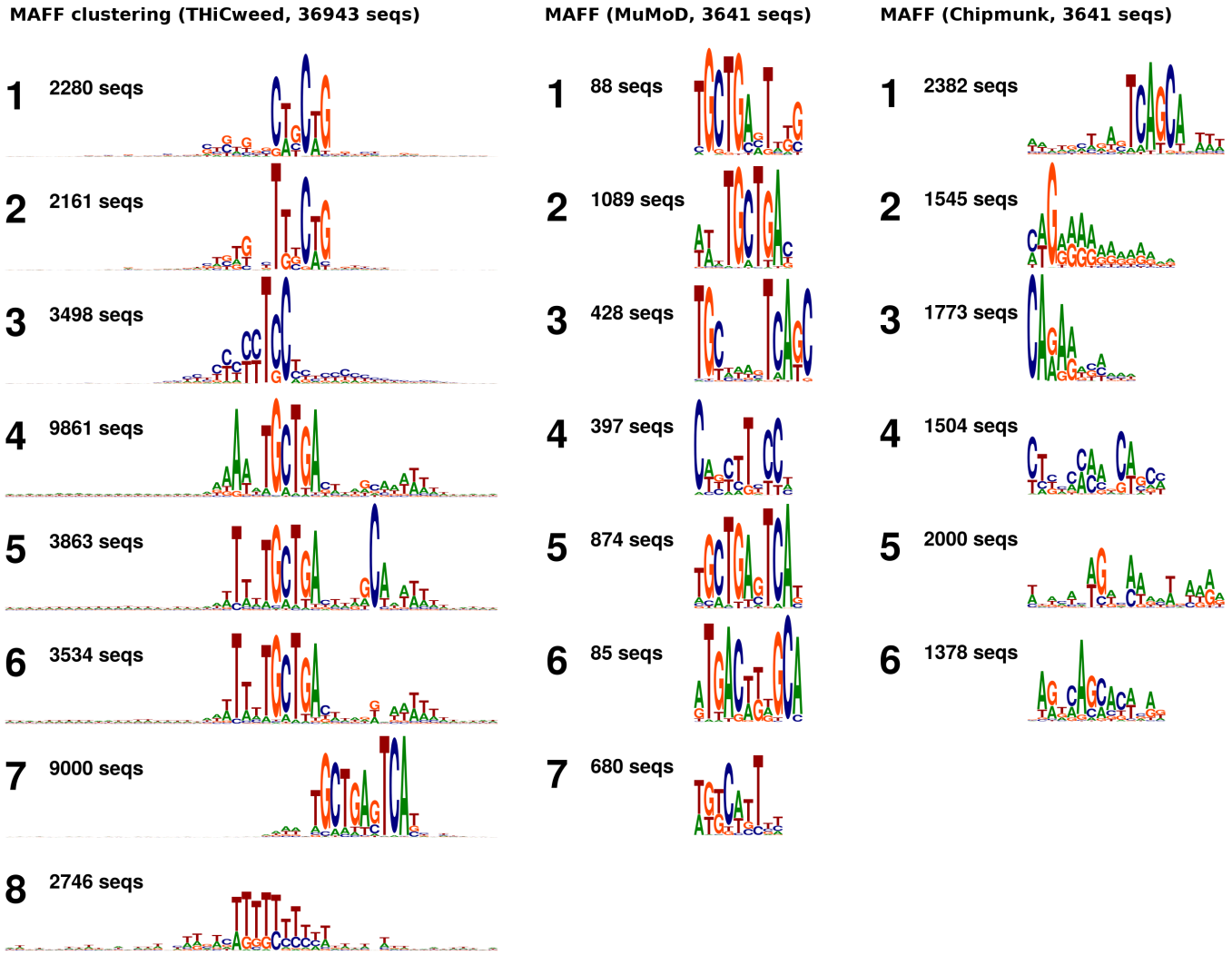

Figure 3: Comparing clustering of MAFF chipseq peaks using three programs. While all programs pick up the dominant TCAGCA motif, the surrounding context picked up by THiCweed is largely missed by the others.

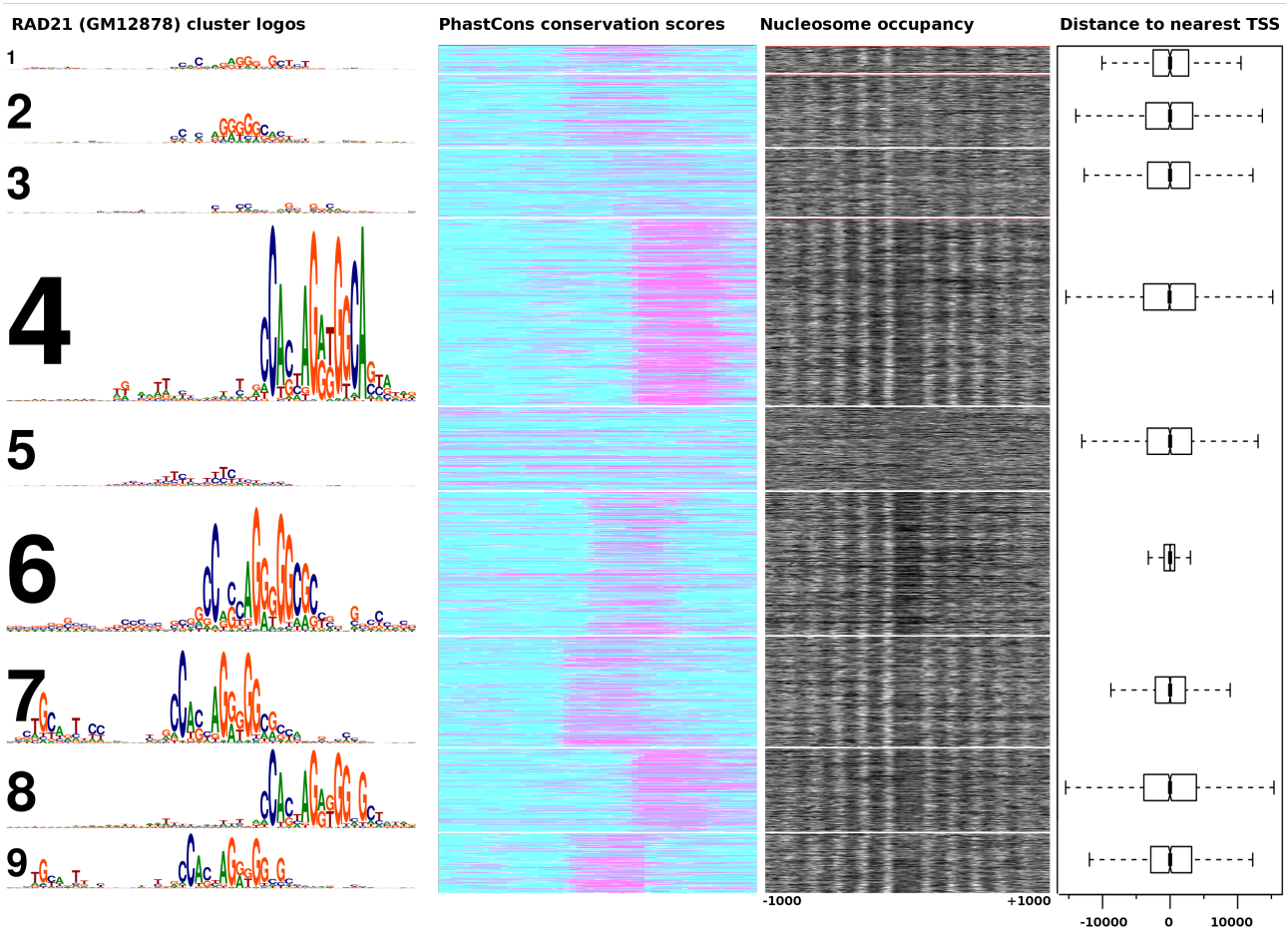

Figure 4: Comparison of sequence clusters of RAD21 with nucleosome occupancy (colour scale: black = 0, white = 5+), PhastCons conservation score (colour scale:blue=0, pink=1), and distance from nearest TSS, suggesting connections between the motif structure in different sequence clusters, biological function, and evolutionary conservation pressure.

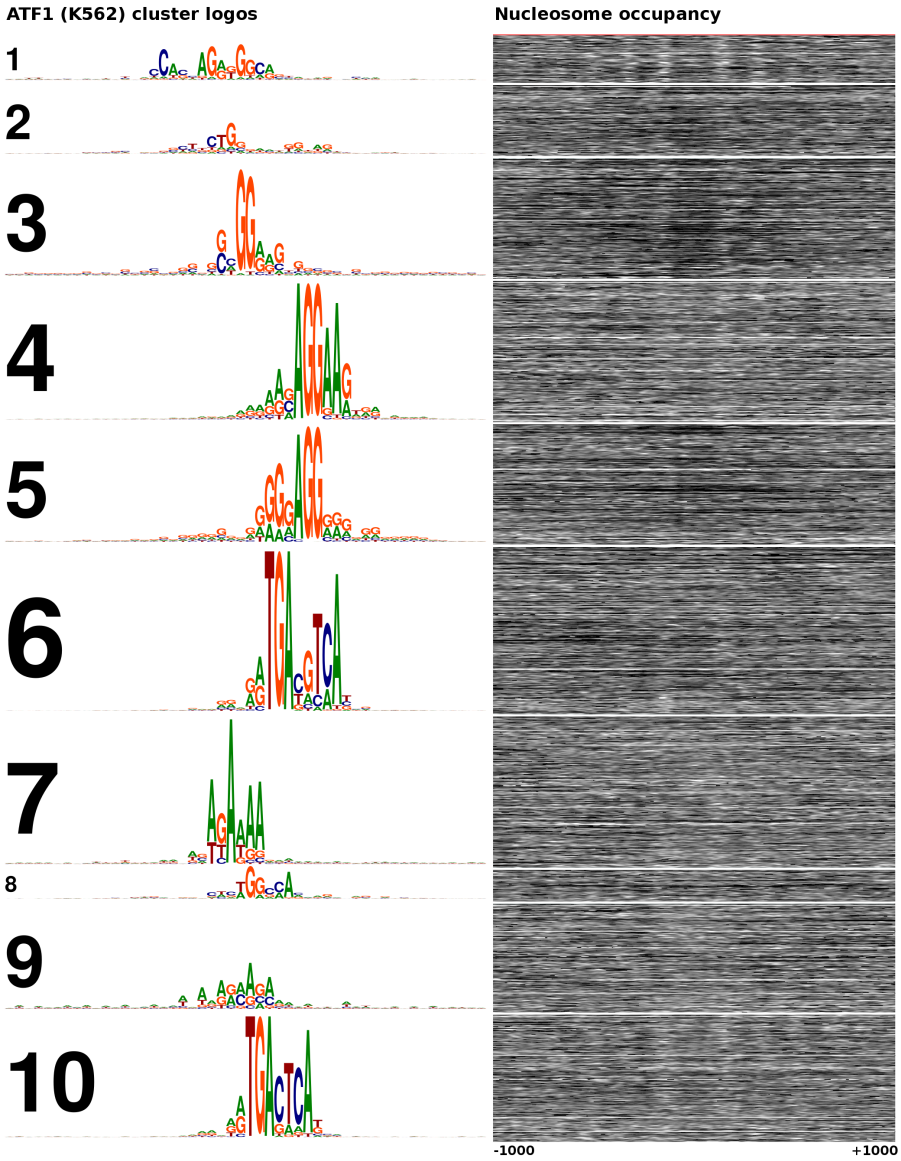

Figure 5: Comparison of sequence clusters of ATF1 with nucleosome occupancy (colour scale: black=0, white = 5+). The cluster with a CTCF-like motif shows significantly more correlated nucleosome positioning.

out resorting to training on a small subset of the sequences (as Weeder does). The clustering algorithm is heuristic and stochastic, but is essentially similar to an iterated K-means clustering with $K = 2$, with significance criteria to avoid spurious splits. Instead of invoking pairwise distances and calculating a centroid, however, we calculate multinomial likelihoods correctly within the limitations of the PWM assumption. The clustering algorithm and wide-window approach ensures that little or no prior information is required to run the program: significant short motifs can be found inside longer windows by eyeballing, but other relevant sequence features can be picked up too.

At the moment, THiCweed runs on a single CPU core, but significant speedups should be possible by parallelization.

Finally, in cases where there is a profusion of similar but slightly different motif patterns (as in the RAD21/CTCF case), it appears that the differences may have biological significance, as reflected by nucleosome positioning and phylogenetic conservation. We plan to explore this further in a future work.

Acknowledgements

We thank Snehal V Sambare for assistance in the benchmarking, and Vishaka Datta and Arvind Shankar for useful discussions.

Funding

LN was supported by a Wellcome Trust-DBT India Alliance Early Career fellowship. RS was funded by the PRISM 12th plan project at his institute, under the Department of Atomic Energy, Government of India.

Author contributions

AA, LN and RS conceived the basic algorithm. AA implemented an early prototype. RS implemented the current version in the Julia language. All authors contributed to refinements in the algorithm during development, and to the benchmarks described here. All authors read and approved the final manuscript.

References

Bailey, T. L. and Elkan, C. (1994). Fitting a mixture model by expectation maximization to discover motifs in biopolymers. *Proc Int Conf Intell Syst Mol Biol*, **2**, 28–36.

ENCODE Project Consortium *et al.* (2012). An integrated encyclopedia of dna elements in the human genome. *Nature*, **489**(7414), 57–74.

He, Q., Johnston, J., and Zeitlinger, J. (2015). Chip-nexus enables improved detection of in vivo transcription factor binding footprints. *Nature biotechnology*, **33**(4), 395–401.

Hertz, G. Z., Hartzell, G. W., and Stormo, G. D. (1990). Identification of consensus patterns in unaligned DNA sequences known to be functionally related. *Comput Appl Biosci*, **6**(2), 81–92.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, **2**(1), 193–218.

Hubisz, M. J., Pollard, K. S., and Siepel, A. (2010). Phast and rphast: phylogenetic analysis with space/time models. *Briefings in bioinformatics*, page bbq072.

Hughes, J. D., Estep, P. W., Tavazoie, S., and Church, G. M. (2000). Computational identification of cis-regulatory elements associated with groups of functionally related genes in Saccharomyces cerevisiae. *J Mol Biol*, **296**(5), 1205–1214.

Johnson, D. S., Mortazavi, A., Myers, R. M., and Wold, B. (2007). Genome-wide mapping of in vivo protein-dna interactions. *Science*, **316**(5830), 1497–1502.

Karolchik, D., Baertsch, R., Diekhans, M., Furey, T. S., Hinrichs, A., Lu, Y., Roskin, K. M., Schwartz, M., Sugnet, C. W., Thomas, D. J., *et al.* (2003). The ucsc genome browser database. *Nucleic acids research*, **31**(1), 51–54.

Kulakovskiy, I., Levitsky, V., Oshchepkov, D., Bryzgalov, L., Vorontsov, I., and MAKEEV, V. (2013). From binding motifs in chip-seq data to improved models of transcription factor binding sites. *Journal of bioinformatics and computational biology*, **11**(01), 1340004.

Kulakovskiy, I. V., Boeva, V., Favorov, A. V., and Makeev, V. J. (2010). Deep and wide digging for binding motifs in chip-seq data. *Bioinformatics*, **26**(20), 2622–2623.

Landt, S. G., Marinov, G. K., Kundaje, A., Kheradpour, P., Pauli, F., Batzoglou, S., Bernstein, B. E., Bickel, P., Brown, J. B., Cayting, P., *et al.* (2012). Chip-seq guidelines and practices of the encode and modencode consortia. *Genome research*, **22**(9), 1813–1831.

Levitsky, V. G., Kulakovskiy, I. V., Ershov, N. I., Oshchepkov, D. Y., Makeev, V. J., Hodgman, T., and Merkulova, T. I. (2014). Application of experimentally verified transcription factor binding sites models for computational analysis of chip-seq data. *BMC genomics*, **15**(1), 80.

Machanick, P. and Bailey, T. L. (2011). Meme-chip: motif analysis of large dna datasets. *Bioinformatics*, **27**(12), 1696–1697.

Mathelier, A., Fornes, O., Arenillas, D. J., Chen, C.-y., Denay, G., Lee, J., Shi, W., Shyr, C., Tan, G., Worsley-Hunt, R., *et al.* (2016). Jaspar 2016: a major expansion and update of the open-access database of transcription factor binding profiles. *Nucleic acids research*, **44**(D1), D110–D115.

Millau, J.-F. and Gaudreau, L. (2011). Ctcf, cohesin, and histone variants: connecting the genome. *Biochemistry and Cell Biology*, **89**(5), 505–513.

Mitra, S. and Narlikar, L. (2016). No promoter left behind (nplb): learn de novo promoter architectures from genome-wide transcription start sites. *Bioinformatics*, **32**(5), 779–781.

Narlikar, L. (2013). MuMoD: a Bayesian approach to detect multiple modes of protein-DNA binding from genome-wide ChIP data. *Nucleic Acids Res.*, **41**(1), 21–32.

Narlikar, L. (2014). Multiple novel promoter-architectures revealed by decoding the hidden heterogeneity within the genome. *Nucleic Acids Res.*, **42**(20), 12388–12403.

Pavesi, G., Mereghetti, P., Mauri, G., and Pesole, G. (2004). Weeder web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. *Nucleic acids research*, **32**(suppl 2), W199–W203.

Phillips, J. E. and Corces, V. G. (2009). Ctcf: master weaver of the genome. *Cell*, **137**(7), 1194–1211.

Rhee, H. S. and Pugh, B. F. (2011). Comprehensive genome-wide protein-dna interactions detected at single-nucleotide resolution. *Cell*, **147**(6), 1408–1419.

Roth, F. P., Hughes, J. D., Estep, P. W., and Church, G. M. (1998). Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat Biotechnol*, **16**(10), 939–945.

Sandelin, A., Alkema, W., Engström, P., Wasserman, W. W., and Lenhard, B. (2004). Jaspar: an open-access database for eukaryotic transcription factor binding profiles. *Nucleic acids research*, **32**(suppl 1), D91–D94.

Schmidt, D., Schwalie, P. C., Wilson, M. D., Ballester, B., Gonçalves, Â., Kutter, C., Brown, G. D., Marshall, A., Flicek, P., and Odom, D. T. (2012). Waves of retrotransposon expansion remodel genome organization and ctcf binding in multiple mammalian lineages. *Cell*, **148**(1), 335–348.

Siddharthan, R. (2008). PhyloGibbs-MP: module prediction and discriminative motif-finding by Gibbs sampling. *PLoS Comput Biol*, **4**(8), e1000156.

Siddharthan, R., Siggia, E. D., and van Nimwegen, E. (2005). PhyloGibbs: A Gibbs sampling motif finder that incorporates phylogeny. *PLoS Comput Biol*, **1**(7), e67.

Sloan, C. A., Chan, E. T., Davidson, J. M., Malladi, V. S., Strattan, J. S., Hitz, B. C., Gabdank, I., Narayanan, A. K., Ho, M., Lee, B. T., *et al.* (2016). Encode data at the encode portal. *Nucleic acids research*, **44**(D1), D726–D732.

Stormo, G. D. and Hartzell, G. W. (1989). Identifying protein-binding sites from unaligned DNA fragments. *Proc Natl Acad Sci U S A*, **86**(4), 1183–1187.

Wang, J., Zhuang, J., Iyer, S., Lin, X.-Y., Greven, M. C., Kim, B.-H., Moore, J., Pierce, B. G., Dong, X., Virgil, D., *et al.* (2013). Factorbook. org: a wiki-based database for transcription factor-binding data generated by the encode consortium. *Nucleic acids research*, **41**(D1), D171–D176.

Zambelli, F., Pesole, G., and Pavesi, G. (2014). Using weeder, pscan, and pscanchip for the discovery of enriched transcription factor binding site motifs in nucleotide sequences. *Current Protocols in Bioinformatics*, pages 2–11.