

A New Exhaustive Method and Strategy for Finding Motifs in ChIP-Enriched Regions

Caiyan Jia^{1,2*}, Matthew B. Carson^{3,4}, Yang Wang¹, Youfang Lin¹, Hui Lu^{2,5*}

1 School of Computer and Information Technology & Beijing Key Lab of Traffic Data Analysis, Beijing Jiaotong University, Beijing, China, **2** Department of Bioengineering/Bioinformatics, University of Illinois at Chicago, Chicago, Illinois, United States of America, **3** Center for Healthcare Studies, Institute for Public Health and Medicine, Northwestern University Feinberg School of Medicine, Chicago, Illinois, United States of America, **4** Division of Health and Biomedical Informatics, Department of Preventive Medicine, Northwestern University Feinberg School of Medicine, Chicago, Illinois, United States of America, **5** Shanghai Institute of Medical Genetics, Shanghai Children's Hospital, Shanghai JiaoTong University, Shanghai, China

Abstract

ChIP-seq, which combines chromatin immunoprecipitation (ChIP) with next-generation parallel sequencing, allows for the genome-wide identification of protein-DNA interactions. This technology poses new challenges for the development of novel motif-finding algorithms and methods for determining exact protein-DNA binding sites from ChIP-enriched sequencing data. State-of-the-art heuristic, exhaustive search algorithms have limited application for the identification of short (l, d) motifs ($l \leq 10, d \leq 2$) contained in ChIP-enriched regions. In this work we have developed a more powerful exhaustive method (FMotif) for finding long (l, d) motifs in DNA sequences. In conjunction with our method, we have adopted a simple ChIP-enriched sampling strategy for finding these motifs in large-scale ChIP-enriched regions. Empirical studies on synthetic samples and applications using several ChIP data sets including 16 TF (transcription factor) ChIP-seq data sets and five TF ChIP-exo data sets have demonstrated that our proposed method is capable of finding these motifs with high efficiency and accuracy. The source code for FMotif is available at <http://211.71.76.45/FMotif/>.

Citation: Jia C, Carson MB, Wang Y, Lin Y, Lu H (2014) A New Exhaustive Method and Strategy for Finding Motifs in ChIP-Enriched Regions. PLoS ONE 9(1): e86044. doi:10.1371/journal.pone.0086044

Editor: Ying Xu, University of Georgia, United States of America

Received: June 3, 2013; **Accepted:** December 4, 2013; **Published:** January 24, 2014

Copyright: © 2014 Jia et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Funding: This work was supported in part by National Nature Science Foundation of China (Grant No. 60905029, 61105055, 61105056, 81230086, and 31071167), the Beijing Natural Science Foundation (Grant No. 4112046), and the Fundamental Research Funds for the Central Universities. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: cyjia@bjtu.edu.cn (CJ); huilu.bioinfo@gmail.com (HL)

Introduction

Protein-DNA interactions play key roles in several cellular processes and functions including DNA transcription, packaging, replication, and repair. Identification of regions such as transcription factor binding sites (TFBSs), which are targeted by proteins called transcription factors (TFs), is crucial for a better understanding of transcriptional regulation. Although traditional footprinting assays can accurately identify the precise binding sites of any factor, this low-throughput method is highly technical and can only be used to analyze a single small region (< 1 kilobase pairs (kb)) at a time. Chromatin immunoprecipitation followed by high-throughput deep sequencing (ChIP-seq) enables genome-wide detection of transcription factor binding sites as well as the localization of epigenetic regulatory markers on a genomic scale [1,2]. It typically returns millions of short (35–50 base pairs (bps)) sequence tags mapped onto a reference genome from a sample organism. Putative binding sites with high confidence can be extracted from peak-enriched regions in the genome by peak-calling programs [3]. However, the resolution of binding regions identified from ChIP-seq can be a few hundred base pairs and is one or two orders of magnitude larger than a typical TFBS. By using an exonuclease that trims DNA regions at a precise distance from binding sites, the novel ChIP-seq technique ChIP-exo is able to locate binding sites at high resolution [4]. However, according to the results in Rhee and Pugh [4], binding regions identified

from ChIP-exo experiments may be tens of bps away from the exact binding locations, although some of them at the location indicated by the experiments. Computational methods are still needed to identify the exact binding locations of a TF in ChIP-seq or ChIP-exo data sets.

Binding sites for a specific TF are often highly conserved and have strong evidence for sequence specificity [5]. An actual DNA region interacting with and bound by a single TF usually ranges in size from 8–10 to 16–20 bps. In the past two decades, numerous programs have been developed to identify over-represented DNA sequence motifs from the promoters of co-regulated or homologous genes [6]. These programs can be divided into two groups. The first includes profile-based methods such as CONSENSUS [7], MEME [8], Gibbsampler [9], AlignACE [10], PROJECTION [11], and CRMD [12], each of which attempts to maximize a statistic- or entropy-related score from a profile matrix (also called a position weight matrix (PWM)). The second group is comprised of consensus-based methods, which include SPELLER [13], WEEDER [14,15], MITRA-count [16], Voting [17], PMSprune [18], WINNOWER [19], iTriplet [20], VINE [21], Stemming [22], and RecMotif [23]. These programs are designed to find potential (l, d) motifs within DNA sequences [19], where l is the length of a motif and d is the maximum number of mutations between a predicted binding site and the motif consensus. In most cases, profile-based methods are faster but suffer from lower

accuracy due to their tendency to be trapped in a local optimum. Consensus-based methods are more accurate but slower due to the exponential growth of the search space with increasing values of l and d .

Consensus-based methods can be further divided into two categories: pattern-driven and sample-driven approaches [16]. A pattern-driven approach attempts to enumerate all possible 4^l l -mer motifs with lexical order, while a sample-driven approach tries to test all possible (l, d) motifs generated from real l -mers of input sequences. For the methods mentioned above, SPELLER, WEEDER, and MITRA-count are pattern-driven approaches and Voting, PMPprune, WINNOWER, iTriplet, VINE, Stemming, and RecMotif are sample-driven. By using pattern-driven approaches (with the exception of MITRA-count), one can automatically find planted (l, d) motifs without prior knowledge of the length l . On the contrary, sample-driven approaches require that l be specified for each run. In real applications, the exact length of motifs contained in a set of sequences is usually unknown. The pattern-driven algorithm WEEDER has been successful in real eukaryotic applications [24] but has not been improved upon to the best of our knowledge. In this study, we have developed a more powerful method to extract (l, d) motifs and their binding locations contained in DNA sequences without prior knowledge of motif length and have used this method to identify motifs and their binding locations in ChIP-enriched regions.

The pattern-driven approach MITRA-count builds a mismatch tree for all l -mers first, then traverses search space recursively from the root down in depth-first order. Therefore, the length l of a predicted motif must be specified in advance. SPELLER enumerates all possible motifs in a depth-first manner throughout the search space, then scans and counts all possible instances of the current motif with length i ($i \in \{1, 2, \dots, l\}$) from the suffix tree of input sequences. The algorithm can identify planted (l, d) motifs efficiently when $l \leq 13$ and $d \leq 3$ (see Table 1). In order to increase the speed of SPELLER, WEEDER includes an error ratio ε ($\varepsilon \approx d/l$) for the algorithm that narrows the search space such that for all $i \in \{1, 2, \dots, l\}$ the number of mismatches between the first i nucleotides of a candidate l -mer motif and the first i nucleotides of a valid instance of the motif is at most εi . The algorithm can accelerate SPELLER to some extent, especially when d/l is small (e.g., $d/l \leq 0.25$). Unfortunately, not all motif occurrences satisfy this restriction. WEEDER must lower the occurrence frequency $q \leq N$ to make sure exact motifs will not be missed. However, WEEDER's run time increases dramatically with the decrease of q . For instance, for (15, 4), q should be lowered to half the number of sequences at the OOPS constraint (one occurrence(s) of the motif instance(s) per sequence) to make sure that the true motif will be discovered [14]. However, WEEDER's run time may be even longer than SPELLER under the condition that the two algorithms use the same programming techniques. Thus, a more efficient method is needed to improve the efficiency of pattern-driven algorithms without knowledge of the length of predicted motifs under the ZOMOPS constraint (zero, one or multiple occurrence(s) of the motif instance(s) per sequence).

Additionally, the programs mentioned above are not computationally efficient enough to process a large number of ChIP-seq peaks. In recent years, several programs have been developed to cope with large-scale ChIP-seq data. Some are ChIP-tailored versions of previously-developed software (e.g., ChIP-MEME [25], DREME [26], and GimmeMotifs [27]). These typically restrict motif discovery to a few hundred peaks and usually ignore the remaining unselected sequences. Other programs are faster versions of previous software (e.g., STEME [28], ChIPMunk [29], and HMS [30]). STEME is a faster version of MEME and

Table 1. Comparisons between FMotif and other pattern-driven algorithms on (l, d) samples with $N = 20$, $L = 600$, and $\alpha = 0\%$ noise sequences.

$(l, d); \tau = d/l$	SPELLER	WEEDER(q)	MITRA	FMotif
(10, 2); $\tau = 0.2$	17.16s-1	7.47s (19)-1	1.83s-2	0.59s-1
(11, 2); $\tau = 0.18$	17.73s-1	32.53s (15)-1	1.82s-1	0.59s-1
(12, 3); $\tau = 0.25$	4.42m-1	9.35m (15)-1	21.22s-1	6.77s-1
(13, 3); $\tau = 0.23$	4.42m-1	2.80m (18)-1	21.25s-3	6.73s-1
(14, 4); $\tau = 0.28$	1.05h-1	2.41h (15)-1	3.94m-1	1.32m-1
(15, 4); $\tau = 0.27$	1.05h-1	1.08h (16)-1	3.93m-1	1.31m-1
(15, 5); $\tau = 0.33$	—	—	41.25m-2	15.51m-1
(16, 5); $\tau = 0.31$	—	—	41.19m-1	15.50m-1
(17, 6); $\tau = 0.35$	—	—	6.58h-1	3.17h-1
(18, 6); $\tau = 0.35$	—	—	6.84h-1	3.17h-1

'WEEDER(q)' indicates the execution time of WEEDER given the occurrence frequency threshold q . '—' indicates a run time of over 10 hours. $s, m,$ and h are the units of a run time and denote seconds, minutes, and hours respectively. The number after each run time is the ranking number of a true planted motif among the top 25 predicted motifs. '/' after a run time indicates that the real motifs were not in the top 25.

doi:10.1371/journal.pone.0086044.t001

involves indexing sequences with a suffix tree, which accelerates the expectation-maximization (EM) steps. ChIPMunk combines EM with a greedy approach similar to CONSENSUS and decreases the run time of the optimization procedure. HMS is an improved version of Gibbs Sampler and combines stochastic sampling with deterministic, greedy search steps. Another group of programs integrate other information such as TFBS positional priors [31] or transcription start sites [32] in order to optimize a PWM of ChIP-enriched regions. As mentioned above, these programs still have a local optimum problem. Similar to SPELLER and WEEDER, some of these programs are consensus-based methods (sometimes called word enumeration methods). These include RSAT [33], Cisfinder [34] and POSMO [35]. RSAT is a word enumeration method and has been developed to process whole ChIP-seq peak data sets, but is limited to short (l, d) motifs ($l \leq 10, d \leq 2$). Cisfinder is a word clustering method and combines short k -mer enumeration ($k = 7, 8,$ or 9) with a clustering strategy. POSMO, also a word clustering method, uses TFBS positional bias information along with k -mer enumeration and clustering. However, both Cisfinder and POSMO use clustering methods to group short k -mers and therefore cannot find exact (l, d) motifs contained in sequences. Thus, finding exact (l, d) motifs with larger values of l and d in a large-scale sequence data set is still very difficult.

According to a previous study by Keich and Pevzner [36], real signals may be mixed with spurious motifs contained in background sequences under the OOPS constraint when the degenerative ratio $\tau = d/l > 0.25$. A larger τ makes it more difficult to discriminate between a real motif and spurious motifs. However, some sequences may not contain any occurrence of a motif. As previously mentioned, we have concentrated on a more generalized model (the ZOMOPS constraint). Under this constraint, we have found that, except for the degenerative ratio τ , the ratio of noise sequences $\alpha = (N - Q)/N$, where N is the number of sequences and Q is the number of sequences containing at least one variant of a motif, negatively affects (l, d) motif searches. A larger α leads to more spurious motifs in background sequences. It is suspected that 30% of factor-bound locations in

ChIP-seq data may be false positives [4]. Plus, there may be different versions of DNA-binding motifs for any given TF. A specified motif may only occur in 30% of binding regions. Although false positive rates in ChIP-seq data sets are low enough that statistical conclusions can be drawn in most cases, the noise (plus the diversity of DNA-binding modes) still interrupts the motif-finding process and alters motif-finding results. Thus, this may not be the best way to identify motifs in full-size ChIP-seq data sets. After running a peak-calling program on a raw ChIP-seq data set, peaks along with their ChIP enrichment values, p -values, or false discovery rates (FDRs) can be obtained. False positive peaks are those with low peak enrichment values, p -values, or FDRs. A better method may be to find motifs with a high confidence value (i.e., those that are plentiful enough to draw statistical conclusions) in peak-enriched regions and subsequently scan their binding locations with the degenerative value d in the remaining peak regions that have low peak enrichment values, p -values, or FDRs. This would not only exclude more noise and spurious motifs [37], but it would also take advantage of well-developed motif-finding tools with an acceptable level of scalability. A similar idea was used in MICSA and achieved good performance [38]. However, MICSA used the optimal method MEME (the accuracy of which is limited [12,19]) to get the PWM of a motif for only the first three hundred peak-enriched regions.

In this study we have found that for motifs with length l , both SPELLER and WEEDER have been designed to check each i -mer ($i \leq l$) in the pattern space with depth-first order and count the variants of the i -mer in the suffix tree of sequences from the root to layer i . The suffix tree is scanned one time for each i -mer pattern. Thus, as i increases, the algorithms scan the suffix tree an increasing number of times. In fact, the mismatch information in layer i of a suffix tree can be used to search for $(i+1)$ -mers in the pattern space. For this reason, we constructed a new suffix tree structure with mismatch information (called a mismatched suffix tree) and developed a fast motif enumerative method (FMotif) under the ZOMOPS constraint. Using the newly constructed suffix trees, we incorporated the mismatch information in layer i of the mismatched suffix trees to verify $(i+1)$ -mers in the pattern space. We then updated mismatch information in layer $i+1$ of the mismatched suffix trees. In this way we were able to implement a depth-first search within the pattern space and the mismatched suffix trees simultaneously. To process large-scale ChIP-seq data sets, we integrated the peak detection method MACS [39] with our motif-finding method and ChIP-enriched sampling strategy, which allowed us to locate the exact binding locations in ChIP-seq and ChIP-exo data sets. We chose MACS because it has been shown to perform well when compared to several other peak-calling programs [3].

Results

Experimental Results on Artificial Data Sets

We compared FMotif with the existing pattern-driving methods including SPELLER, WEEDER, and MITRA-count (MITRA for short) on synthetic samples to show the efficiency of our proposed method. All synthetic samples were generated following the method of Pevsner and Sze [19], where Q ($Q \leq N$) variants of an l -length motif were randomly planted into Q sequences selected randomly from a set of N sequences with length L . In this (l, d) model, each planted variant of the motif with length l had exactly d mismatches with the motif itself.

In the first group of experiments, we tested the performance of these algorithms on (l, d) sample sets without noise sequences (i.e., $Q = N$) at standard settings, where the number N and the length L

of sequences are set to 20 and 600, respectively [14,16,19]. These test results are shown in Table 1. 'WEEDER(q)' indicates the execution time of WEEDER given the occurrence frequency threshold q . '-' indicates a run time of over 10 hours. s , m , and h denote seconds, minutes, and hours respectively. The number after each run time is the ranking number of a true planted motif among the top 25 predicted motifs. '/' after a run time indicates that the real motifs were not in the top 25. In the second group of experiments, we first tested the influence of the ratio of noise sequences α ($\alpha = (N - Q)/N$) on (l, d) samples using FMotif with typical settings (i.e., $N = 20$ and $L = 600$). In order to provide a more comprehensive comparison of the calculation speed when noise was added, we compared FMotif to SPELLER and MITRA on (10, 2), (11, 2), (12, 3), and (13, 3) samples. We avoided comparisons over motifs more complicated than (13, 3) because SPELLER lacked computational efficiency on these problems and WEEDER required tuning of parameter q . These test results are shown in Table 2, where α is set at 5%, 10%, 15%, ..., and 40%, '/' indicates that the real motifs were not in the top 25, and the first line for (10, 2), (11, 2), (12, 3) and (13, 3) is the FMotif result, the second line (denoted by 'M..') is the MITRA result, and the third line (denoted by 'S..') is the SPELLER result. We then tested the influence of the noise ratio α on samples with $N = 1000$ and $L = 100$ to simulate ChIP-enriched regions because those regions are usually relatively short and the number of regions is usually large. We subsequently compared FMotif to SPELLER and MITRA on (10, 2), (11, 2), (12, 3), and (13, 3) samples as before. These test results are shown in Table 3, where α is set at 10%, 20%, ..., and 80%. In the third group of experiments, we tested FMotif scalability using two groups of samples to see whether it was suitable for recognizing motifs in large-scale ChIP-enriched regions. The settings of the first group were $L = 100$, $N = 1000, 2000, \dots, 8000$ and no noise sequences ($\alpha = 0\%$). These test results are shown in Table 4. The settings of the second group were $L = 100$, $N = 1000, 2000, \dots, 8000$ and $\alpha = 30\%$ noise sequences in order to mimic ChIP-seq data. These test results are shown in Table 5. All experiments were performed on a computer with an Intel 2.99 GHz processor, 2.00GB of main memory, and the Windows XP operating system.

The results in Tables 1–5 lead to three observations. First, FMotif is a fast and exact algorithm and capable of finding (l, d) motifs in synthetic samples without being given the length l of a predicted motif. It performs faster than SPELLER, MITRA, and WEEDER without sacrificing accuracy. As mentioned above, WEEDER's efficiency suffers significantly (see (14, 4) in Table 1 for an example) when the occurrence frequency threshold q is too low, and MITRA requires that the length l be specified a priori. It should be noted that FMotif ranked all motifs with different lengths together by significance score. For the samples whose true motifs were not ranked in the top 25, the top motifs were usually $(l-1)$ - or $(l-2)$ - substrings of the true motifs with length l . In these cases the true motifs were still in the output list but were ranked below the top 25. Second, noise sequences have a strong effect on the results and the speed of the method. With an increase in α , the run time increases as well. Like the degenerative ratio $\tau = d/l$, the ratio of noise sequences α also weakens motifs, especially when background sequences are long (see Table 2). Spurious motifs in background sequences bury the authentic signals when either τ or α is large. For example, real motifs were difficult to filter by their significance score for the (15, 5) motif in Table 2, even when the noise sequence ratio was set to 5%. In this case, many spurious motifs of length 14–15 with a large significance score were ranked among the top 25. When the length of background sequences was shorter and the number of

Table 2. Results for noise-influenced models on (l, d) samples with $N = 20$, $L = 600$, and $\alpha = 5\%$, 10% , \dots , 40% noise sequences.

(l, d)	5%	10%	15%	20%	25%	30%	35%	40%
(10, 2)	0.78 _s -1	0.95 _s -1	1.06 _s -1	1.19 _s -1	1.30 _s -1	1.44 _s -3	1.63 _s -5	/
M..	2.78 _s -1	3.05 _s -1	3.23 _s -1	3.44 _s -1	3.78 _s -1	4.83 _s -3	5.23 _s -15	/
S..	38.99 _s -1	1.09 _m -1	1.53 _m -1	1.96 _m -1	2.51 _m -1	3.34 _m -3	4.69 _m -21	/
(11, 2)	0.77 _s -1	0.94 _s -1	1.06 _s -1	1.17 _s -1	1.30 _s -1	1.44 _s -1	1.61 _s -1	1.83 _s -1
M..	2.77 _s -1	3.05 _s -1	3.23 _s -1	3.23 _s -1	3.44 _s -1	3.77 _s -1	4.38 _s -1	5.19 _s -1
S..	38.13 _s -1	1.08 _m	1.52 _m -1	1.95 _m -1	2.48 _m -1	3.31 _m -1	4.66 _m -1	6.53 _m -1
(12, 3)	9.16 _s -1	11.69 _s -1	13.92 _s -1	15.88 _s -1	17.61 _s -6	/	/	/
M..	33.13 _s -1	33.23 _s -1	39.28 _s -1	43.45 _s -1	46.64 _s -4	50.36 _s -20	/	/
S..	9.78 _m -1	17.69 _m -1	26.94 _m -1	36.16 _m -1	45.64 _m -2	58.11 _m -8	/	/
(13, 3)	9.17 _s -1	11.64 _s -1	13.95 _s -1	15.88 _s -1	17.63 _s -1	19.50 _s -5	21.83 _s -2	24.80 _s -1
M..	27.23 _s -1	34.38 _s -1	40.22 _s -1	43.44 _s -1	46.66 _s -1	50.58 _s -1	57.14 _s -1	1.12 _m -1
S..	9.80 _m -1	17.71 _m -1	27.07 _m -1	36.12 _m -1	45.80 _m -1	58.33 _m -1	1.30 _h -1	1.79 _h -1
(14, 4)	1.85 _m -1	2.33 _m -1	2.89 _m -9	3.47 _m -5	/	4.52 _m -20	/	/
(15, 4)	1.82 _m -1	2.32 _m -1	2.89 _m -1	3.48 _s -1	4.03 _m -1	4.52 _m -3	5.02 _m -3	5.66 _m -1
(15, 5)	/	/	/	/	/	/	/	/
(16, 5)	22.90 _m	29.52 _m -1	36.05 _m -1	/	/	/	/	/
(17, 6)	/	/	/	/	/	/	/	/
(18, 6)	5.18 _h	7.07 _h -1	/	/	/	/	/	/

The ratio of noise sequences α is set at 5%, 10%, 15%, \dots , and 40%, '/' indicates that the real motifs were not in the top 25. The number after each run time is the ranking number of a true planted motif among the top 25 predicted motifs. The first line for (10, 2), (11, 2), (12, 3) and (13, 3) is the FMotif result, the second line (denoted by 'M..') is the MITRA result, and the third line (denoted 'S..') is the SPELLER result. s, m, h denote seconds, minutes, and hours respectively.

doi:10.1371/journal.pone.0086044.t002

sequences was larger, the signals were stronger and could be easily identified even if a large portion of noise sequences was added (see Table 3). This is consistent with the previous result that false-positives could be reduced by decreasing the sequence length or by adding more sequences to the data set [37]. Third, as shown in Tables 4 and 5, FMotif is capable of operating on a large scale even when there are 30% noise sequences in samples. This allowed us to use FMotif to process peak regions within ChIP-seq and ChIP-exo data sets.

Additionally, we compared FMotif with CisFinder, which uses k -mer enumeration with k -mer clustering to find motifs in large-scale ChIP-seq peak regions. Using both algorithms, we verified the accuracy of FMotif and CisFinder by searching for long (l, d) motifs in synthetic sample sets with 3000 sequences, each of which contained a planted variant of a parent motif. The experimental results are shown in Table 6, where 'Planted Motif' indicates a planted motif consensus in a set of sequences, 'FMotif ($Top-I$)' indicates the top ranked motif consensus found by FMotif in a sample set, 'CisFinder' indicates the closest matching motif consensus (described by IUPAC nucleotide codes) found by CisFinder in a sample set, '#' indicates the number of variants of a reported motif found by FMotif or CisFinder in a sample set, and 'Rank' after '#-' is the ranking number of the reported motif found by Cisfinder in Table 6.

We used the *site-level sensitivity* (sSn) and *positive predictive value* ($sPPV$) metrics described by Tompa [24] to statistically quantify the accuracy of the two methods, where $sSn = sTP / (sTP + sFN)$ and $sPPV = sTP / (sTP + sFP)$, sTP is the number of known sites overlapping predicted sites, sFN is the number of known sites not overlapping predicted sites, and sFP is the number of predicted sites not overlapping known sites. A predicted site overlaps a known site if they share at least a half of the length of known sites. In order to give a more comprehensive comparison of the

accuracy of the two methods on simulated ChIP-seq data sets, we added 30% noise sequences to samples with $N = 3000$ and $L = 100$ and performed the experiments again. These test results are shown in Table 7.

As evident from Tables 6 and 7, FMotif is an exact algorithm. It reported all true motif consensus and their planted variants plus false positive variants in background sequences. CisFinder performed quickly but suffered from low accuracy (due to low *sensitivity*), especially when $\tau = d/l$ was large. FMotif and CisFinder both were robust after 30 noise sequences were added to the samples. It should be pointed out that, although there are various resources on CisFinder's website (<http://lgsun.grc.nia.nih.gov/cisfinder/download.html>), we used only the motif-finding program. There are other programs focused on motif clustering, motif improvement, motif comparison, and other tasks. If all programs were used together, a better motif and more of its binding sites may be identified. However, the CisFinder algorithm [34] was implemented in that motif-finding program and there was no direct way to use all these programs together based on our knowledge.

Experimental Results Using ChIP-seq Data Sets

We tested FMotif using 12 mouse ChIP-seq data sets for 12 DNA-binding TFs (CTCF, cMyc, Esrrb, Klf4, Nanog, nMyc, Oct4, Smad1, Sox2, STAT3, Tcfcp2l1, and Zfx) involved in mouse embryonic stem cell pluripotency and self-renewal [40]. These ChIP-seq data sets have been deposited in the GEO database with ID number GSE11431. We also tested FMotif using four widely used human ChIP-seq data sets for four DNA-binding TFs including CTCF (CCCTC-binding factor [41], named CTCF(h) in the paper), FoxA1 (hepatocyte nuclear factor 3 α [42]), NRSF (neuron-restrictive silencer factor [2]), and STAT1 (signal transducer and activator of transcription protein [1]). The

Table 3. Results for noise-influenced models on (l, d) samples with $N = 1000$, $L = 100$, and $\alpha = 10\%$, 20% , \dots , 80% noise sequences.

(l, d)	$\alpha = 10\%$	$\alpha = 20\%$	$\alpha = 30\%$	$\alpha = 40\%$	$\alpha = 50\%$	$\alpha = 60\%$	$\alpha = 70\%$	$\alpha = 80\%$
(10,2)	6.39 _s -1	7.63 _s -1	10.36 _s -1	12.95 _s -1	13.63 _s -1	14.64 _s -1	23.13 _s -1	24.84 _s -6
M..	12.44 _s -1	12.49 _s -1	19.38 _s -1	31.61 _s -1	31.97 _s -1	32.29 _s -1	1.47 _m -1	1.64 _m -1
S..	1.23 _m -1	2.29 _m -1	6.84 _m -1	10.68 _m -1	12.72 _m -1	15.03 _m -1	1.05 _h -1	1.23 _h -1
(11,2)	6.41 _s -1	7.70 _s -1	10.28 _s -1	12.59 _s -1	13.60 _s -1	14.66 _s -1	23.01 _s -1	24.86 _s -2
M..	12.41 _s -1	12.66 _s -1	18.58 _s -1	31.44 _s -1	31.86 _s -1	32.16 _s -1	1.41 _m -1	1.64 _m -1
S..	1.03 _m -1	2.09 _m -1	5.86 _m -1	10.64 _m -1	12.74 _m -1	15.08 _m -1	1.05 _h -1	1.23 _h -1
(12,3)	1.19 _m -1	1.44 _m -1	1.66 _m -1	2.40 _m -1	2.94 _m -1	3.25 _m -1	3.56 _m -1	/
M..	2.44 _m -1	2.84 _m -1	2.87 _m -1	4.55 _m -1	7.92 _m -1	7.98 _m -1	8.63 _m -1	25.91 _m -1
S..	15.45 _m -1	32.14 _m -1	45.25 _m -1	2.22 _h -1	3.80 _h -1	4.42 _h -1	6.00 _h -1	22.34 _h -1
(13,3)	1.18 _m -1	1.43 _m -1	1.66 _m -1	2.33 _m -1	2.91 _m -1	3.50 _m -1	3.56 _m -1	5.72 _m -10
M..	2.33 _m -1	2.84 _m -1	2.87 _m -1	4.31 _m -1	7.90 _m -1	7.96 _m -1	8.41 _m -1	25.90 _m -1
S..	15.39 _m -1	32.30 _m -1	46.20 _m -1	2.20 _h -1	3.82 _h -1	4.40 _h -1	5.92 _h -1	22.25 _h -1
(14,4)	10.03 _m -1	16.35 _m -1	19.17 _m -1	21.63 _m -1	30.65 _m -1	42.05 _m -1	45.05 _m -1	/
(15,4)	10.07 _m -1	16.36 _m -1	19.19 _m -1	21.78 _m -1	30.67 _m -1	42.02 _m -1	45.04 _m -1	/
(15,5)	1.69 _h -1	2.23 _h -1	3.84 _h -1	4.66 _h -1	5.18 _h -1	7.38 _h -1	10.60 _h -13	/
(16,5)	1.70 _h -1	2.24 _h -1	3.89 _h -1	4.67 _h -1	5.20 _h -1	7.38 _h -1	10.61 _h -3	/

The ratio of noise sequences α is set at 10%, 20%, \dots , and 80%. Row definitions are the same as those in Table 2.
doi:10.1371/journal.pone.0086044.t003

raw sequence of the FoxA1 ChIP-seq data set was downloaded from <http://liulab.dfci.harvard.edu/MACS/Sample.html>. The bed format of the CTCF, NRSF and STAT1 ChIP-seq data sets was downloaded from <http://dir.nhlbi.nih.gov/papers/lmi/epigenomes/sissrs/>. These downloaded short reads were mapped onto the newest version of mouse genome assembly mm10 and human genome assembly hg19, respectively. The peak regions were extracted from these reads using the peak finding program MACS [39] with a false discovery rate (FDR) threshold of 0.2. The reads were ranked by their FDR if a negative control was available, or by p -value otherwise. To prepare the data sets for use with motif discovery algorithms, we mapped the summits of the ChIP-seq peaks and extracted the 100 bps of genomic sequence centered around each peak.

In order to facilitate a fast motif search, avoid the potential influence of false positive peaks, and reduce false positive motifs in background sequences [37], we ran FMotif on the first 3000 ChIP-enriched genomic sequences and then scanned for potential binding locations in the remaining genomic sequences with the degenerative value d . Since binding sites could exist on either DNA strand and CisFinder searched both, we counted the instances of a predicted motif and those of its reverse complement motif. We then compared FMotif with CisFinder and published motifs [2,40,42–44] in literature. The experimental results are shown in Figures 1 and 2, where ‘Nb’ indicates the number of peak-enriched regions predicted by the peak-calling program MACS with an FDR threshold of 0.2 or a p -value threshold of 10^{-5} , ‘FMotif’ and ‘CisFinder’ indicate the closest matching motif logos found by these programs (all motif logos were generated using the web-based tool Weblogo [45]), ‘Literature’ indicates the corresponding motif logos published in literature, ‘#’ indicates the number of binding sites found by either FMotif or CisFinder, and ‘Rank’ after ‘#–’ is the ranking number of a reported motif found by either FMotif or CisFinder.

We compared predicted and published motifs using a motif-level accuracy measure called the *performance coefficient*

$\|U \cap V\| / \|U \cup V\|$, where U is a predicted motif consensus and V is the motif consensus published in literature [19]. As shown in Figures 1 and 2, the motif logos found by FMotif were more accurate compared with published logos from literature than those found by CisFinder. Furthermore, FMotif identified more TFBS locations than CisFinder. As for the 12 mouse TFs DNA-binding logos in [40], Chen et al. used the motif discovery algorithm WEEDER and subsequently extended the motifs using an expectation-maximization method. This second step was necessary because the supplied version of the WEEDER algorithm limited the motif search to a maximum of 12 bps. As discussed in the previous sections, WEEDER operated with low efficiency for long motifs and was difficult to tune for the parameter q .

To estimate the robustness of our sampling strategy, we ran FMotif on the first 500, 1000, 1500, \dots , and 5000 sequences and the full-size ChIP-enriched genomic sequences of TFs n-Myc, Oct4, and NRSF. For all subsets and the full-size data sets, each of the corresponding motifs in Figures 1 and 2 was ranked within the top 25 motifs predicted by FMotif. The ranking number of reported motifs increased with subset size and tended to be stable when the size was greater than 1000. All potential binding sites of reported motifs were obtained from subsets and discovered during the scanning step. Thus, it was not necessary to run a motif-finding algorithm on the whole ChIP-seq data sets, especially when data sets were very large. Additionally, we tested FMotif on N randomly selected sequences ($N = 500, 1000, 1500, \dots$, and 3000). These experiments were repeated 10 times. We then compared these results to those of the first N sequences and those of the last N sequences for TFs n-Myc, Oct4, and NRSF. In general, motif consensus predicted from the first N sequences were the most similar to published motifs and ranked highest in the final output. Those predicted from randomly selected N sequences tended to be ranked second, while those predicted from the last N sequences were usually ranked the lowest. Furthermore, for the same reported motif of a TF, the number of binding sites found in the first N sequences was significantly greater than that

Table 4. A demonstration of FMotif scalability on (l, d) samples for $N = 1000, 2000, \dots, 8000$ sequences, $L = 100$, and no ($\alpha = 0\%$) noise sequences.

(l, d)	1000	2000	3000	4000	5000	6000	7000	8000
(10, 2)	3.34s-1	8.05s-1	11.90s-1	15.80s-1	19.58s-1	23.34s-1	26.83s-1	42.81s-1
(11, 2)	3.36s-1	8.08s-1	11.86s-1	15.61s-1	19.45s-1	23.16s-1	26.84s-1	39.25s-1
(12, 3)	33.41s-1	1.31m-1	1.84m-1	2.38m-1	2.78m-1	3.32m-1	4.03m-1	4.46m-1
(13, 3)	34.62s-1	1.30m-1	1.85m-1	2.40m-1	2.84m-1	3.32m-1	3.82m-1	4.33m-1
(14, 4)	4.51m-1	10.22m-1	15.14m-1	19.93m-1	24.85m-1	29.54m-1	34.25m-1	39.03m-1
(15, 4)	4.52m-1	10.30m-1	15.23m-1	20.05m-1	24.87m-1	29.16m-1	34.38m-1	39.26m-1
(15, 5)	35.06m-1	1.46h-1	2.15h-1	2.68h-1	3.21h-1	3.75h-1	4.27h-1	4.77h-1
(16, 5)	35.02m-1	1.47h-1	2.13h-1	2.70h-1	3.24h-1	3.84h-1	4.27h-1	4.75h-1

The number after each run time is the ranking number of a true planted motif among the top 25 predicted motifs. $s, m,$ and h denote seconds, minutes, and hours respectively.

doi:10.1371/journal.pone.0086044.t004

found in randomly selected N sequences. The number of predicted binding sites found in the last N sequences was the lowest. In some cases there was no corresponding motif in randomly selected N sequences or in the last N sequences when employing the same parameter settings. This situation occurred more often when using the last N sequences. Therefore, we decided that the first N sequences with the lowest p -value or FDR (i.e., the most ChIP-enriched sequences) were the best choice for drawing statistical conclusions about a corresponding motif. This is because, as discussed in the Introduction section, the first N sequences were the least affected by noise. We selected the first 3000 peak regions to be sure that the selected subsets were large enough to account for the specificity of TF-DNA binding and to exclude false positive motifs. The same results may be obtained by running the algorithm on the first 1000–2000 sequences and then scanning potential locations in the remaining sequences.

FMotif Sensitivity

To test the sensitivity of FMotif, we ran it on an NRSF-positive TFBS set (NRSF/qPCR), which was composed of 83 binding sites verified by qPCR [2]. We then ran FMotif on four yeast DNA-binding TFs (Reb1, Gal4, Phd1, and Rap1) and one human TF (CTCF) ChIP-exo data sets. Raw sequence of the five ChIP-exo data sets are available from the NCBI Sequence Read Archive with accession number SRA0044886 [4]. Since it is thought that >98% of ChIP-exo peak regions contain one recognizable

DNA-binding motif within tens of bps away from peak summits, these can be viewed as positive TFBS sets. We used the five ChIP-exo peaks reported in Data S1 from Rhee and Pugh [4]. Similarly, we mapped the summits of ChIP-exo peaks and extracted 50 bps of genomic sequence centered around each peak in yeast genome sacCer3 and human genome hg19, respectively. This allowed us to avoid peak regions overlapping with each other due to some of the summits of ChIP-exo peaks being very close together. Results from CisFinder and published motifs [2,4,43,46–48] in literature are shown for comparison (see Figure 3).

As shown in Figure 3, FMotif was capable of finding more matching motifs and true TF-binding locations when compared to CisFinder. For example, 76 true binding sites of NRSF/qPCR were predicted exactly by FMotif. On the same data set (NRSF/qPCR), MICSA [38] using MEME reported only 55 sites. This highlights the fact that FMotif is capable of identifying TF-binding locations with high sensitivity. It is well-established that specificity is an important consideration for this type of method. However, the ChIP-exo technique is a high-throughput approach, and the resolution of binding regions identified by ChIP-exo may still be tens of base pairs from where the true binding sites of between 8 and 25 base pairs are located. In addition, some of those binding regions are false positives, and it is difficult to say which ones are truly false positives without carefully designed wet-lab experiments. However, we show the specificity (i.e., $sPPV$) of FMotif for artificial samples in Tables 6 and 7. From this information we

Table 5. A demonstration of FMotif scalability on (l, d) samples for $N = 1000, 2000, \dots, 8000$ sequences, $L = 100$, and $\alpha = 30\%$ noise sequences.

(l, d)	1000	2000	3000	4000	5000	6000	7000	8000
(10, 2)	10.33s-1	26.19s-1	45.16s-1	1.10m-1	1.63m-1	2.09m-1	2.70m-1	2.97m-1
(11, 2)	10.33s-1	25.89s-1	45.30s-1	1.09m-1	1.49m-1	1.90m-1	2.43m-1	2.88m-1
(12, 3)	1.66m-1	4.13m-1	7.26m-1	10.70m-1	14.07m-1	17.52m-1	21.40m-1	25.76m-1
(13, 3)	1.66m-1	4.15m-1	7.27m-1	10.72m-1	14.06m-1	17.56m-1	21.51m-1	25.94m-1
(14, 4)	19.14m-1	45.85m-1	1.28h-1	1.90h-1	2.59h-1	3.24h-1	3.88h-1	4.52h-1
(15, 4)	19.16m-1	45.81m-1	1.29h-1	1.91h-1	2.59h-1	3.25h-1	3.89h-1	4.54h-1
(15, 5)	3.86h-1	8.83h-1	14.70h-1	21.28h-1	28.40h-1	35.48h-1	43.08h-1	50.36h-1
(16, 5)	3.87h-1	8.61h-1	14.64h-1	20.81h-1	28.52h-1	35.15h-1	43.35h-1	51.22h-1

Row and column definitions are the same as those in Table 4.

doi:10.1371/journal.pone.0086044.t005

Table 6. Comparisons between FMotif and CisFinder on large (*l, d*) samples with $L = 100$, $N = 3000$, and no ($\alpha = 0\%$) noise sequences.

<i>(l, d)</i>	Planted Motif	FMotif (<i>Top-1</i>)	CisFinder (#-Rank)
		(#)-(sSn)-(sPPV)	(#-Rank)-(sSn)-(sPPV)
(10, 2)	CACGAGAACC	CACGAGAACC (3108)-(1.0)-(0.97)	CACGANAACC (66-1)-(0.02)-(0.98)
(11, 2)	TTGACAAGGAT	TTGACAAGGAT (3026)-(1.0)-(0.99)	TTVACAASGA (186-1)-(0.06)-(0.96)
(12, 3)	TCCATTAGGTGG	TCCATTAGGTGG (3089)-(1.0)-(0.97)	CCWMCTAABKGAMC (80-1)-(0.02)-(0.93)
(13, 3)	CGATAGGTCTATG	CGATAGGTCTATG (3026)-(1.0)-(0.99)	ATAGKYCTA (148-1)-(0.05)-(0.96)
(14, 4)	AGCTATCTATTTAA	AGCTATCTATTTAA (3161)-(1.0)-(0.95)	TAAANWGATA (75-1)-(0.02)-(0.85)
(15, 4)	GATCACACGGAAACC	GATCACACGGAAACC (3022)-(1.0)-(0.99)	CACACGGAAAC (109-3)-(0.04)-(0.98)
(15, 5)	GGTGGGGCGGCGAT	GGTGGGGCGGCGAT (3371)-(1.0)-(0.89)	CMGGYYGGKCG (40-1)-(0.01)-(0.77)
(16, 5)	GAGGCTTGTAACGTT	GAGGCTTGTAACGTT (3062)-(1.0)-(0.98)	GGMKGTAAMGTTKC (59-1)-(0.02)-(0.85)

'Planted Motif' indicates a planted motif consensus in a set of sequences, 'FMotif (*Top-1*)' indicates the top ranked motif consensus found by FMotif in a sample set, 'CisFinder' indicates the closest matching motif consensus (described by IUPAC nucleotide codes) found by CisFinder in a sample set, '#' indicates the number of variants of a reported motif found by FMotif or CisFinder in a sample set, and 'Rank' after '#' is the ranking number of the reported motif found by Cisfinder. The *site-level sensitivity (sSn)* and *positive predictive value (sPPV)* metrics described by Tompa [24] were used to statistically quantify the accuracy of the two methods, where $sSn = sTP / (sTP + sFN)$ and $sPPV = sTP / (sTP + sFP)$, *sTP* is the number of known sites overlapping predicted sites, *sFN* is the number of known sites not overlapping predicted sites, and *sFP* is the number of predicted sites not overlapping known sites. A predicted site overlaps a known site if they share at least a half of the length of known sites.

doi:10.1371/journal.pone.0086044.t006

Table 7. Comparisons between FMotif and CisFinder on large (*l, d*) samples with $L = 100$, $N = 3000$, and $\alpha = 30\%$ noise sequences.

<i>(l, d)</i>	Planted Motif	FMotif (<i>Top-1</i>)	CisFinder
		(#)-(sSn)-(sPPV)	(#-Rank)-(sSn)-(sPPV)
(10, 2)	TGACCCACG	TGACCCACG (2192)-(1.0)-(0.96)	YHGAYCHMACGSM (65-2)-(0.03)-(0.89)
(11, 2)	GCGGTGTACCA	GCGGTGTACCA (2130)-(1.0)-(0.99)	GCGGTNTACC (120-2)-(0.06)-(0.99)
(12, 3)	CACGGGCCTTAG	CACGGGCCTTAG (2182)-(1.0)-(0.96)	CAKSGGCCBBAG (61-2)-(0.03)-(0.85)
(13, 3)	TTCAGTAAGCACG	TTCAGTAAGCACG (2124)-(1.0)-(0.99)	TTCRGTAARCAYG (99-1)-(0.05)-(0.96)
(14, 4)	GCAAGTCACCGTGT	GCAAGTCACCGTGT (2167)-(1.0)-(0.97)	RVAAGTVVBNGTGT (42-2)-(0.02)-(0.90)
(15, 4)	AAGGTGTTGGTATGG	AAGGTGTTGGTATGG (2137)-(1.0)-(0.98)	AARGTGTGGTATGGG (70-2)-(0.03)-(0.90)
(15, 5)	AATACTGTGCATGGA	AATACTGTGCATGGA (2272)-(1.0)-(0.92)	AATWCTGTSCA (27-1)-(0.01)-(0.70)
(16, 5)	AGCTTGCCAGCGACGT	AGCTTGCCAGCGACGT (2145)-(1.0)-(0.98)	VGCTSKCCAGCWACGT (51-1)-(0.02)-(0.90)

Column definitions are the same as those in Table 6.

doi:10.1371/journal.pone.0086044.t007

Data Set (Nb)	FMotif (#-Rank)	CisFinder (#-Rank)	Literature [40]
c-Myc (2840)	(3211-1)	(277-8)	
CTCF (46872)	(37308-1)	(10011-2)	
Esrrb (51784)	(34789-1)	(16485-1)	
Klf4 (17409)	(14800-1)	(277-1)	
Nanog (8367)	(4818-1)	(291-2)	
n-Myc (8032)	(8113-1)	(577-15)	
Oct4 (3775)	(2675-1)	(410-1)	
Smad1 (1188)	(997-2)	(72-1)	
Sox2 (4593)	(2133-1)	(419-1)	
STAT3 (2272)	(1694-1)	(273-1)	
Tcfcp2l1 (24635)	(8165-1)	(1724-1)	
Zfx (19580)	(34059-1)	(828-1)	

Figure 1. Motifs in 12 mouse ES Cell ChIP-seq data sets. FMotif was tested using mouse ChIP-seq data sets for 12 DNA-binding TFs (CTCF, cMyc, Esrrb, Klf4, Nanog, nMyc, Oct4, Smad1, Sox2, STAT3, Tcfcp2l1, and Zfx) involved in mouse embryonic stem cell pluripotency and self-renewal [40]. Results from CisFinder and published motifs in literature are shown for comparison. ‘Nb’ indicates the number of peak-enriched regions predicted by the peak-calling program MACS with an FDR threshold of 0.2 or a *p*-value threshold of 10^{-5} , ‘FMotif’ and ‘CisFinder’ indicate the closest matching motif logos found by these programs (all motif logos were generated using the web-based tool Weblogo [45]), ‘Literature’ indicates the corresponding motif logos published in literature, ‘#’ indicates the number of binding sites found by either FMotif or CisFinder, and ‘Rank’ after ‘#-’ is the ranking number of a reported motif found by either FMotif or CisFinder.
doi:10.1371/journal.pone.0086044.g001

Data Set (Nb)	FMotif (#-Rank)	Cisfinder (#-Rank)	Literature
CTCF(<i>h</i>) (27112)	(23026-1)	(3669-1)	[43]
FoxA1 (14455)	(15940-1)	(1167-1)	[42, 44]
NRSF (5014)	(2081-1)	(1012-1)	[2]
STAT1 (42855)	(16621-1)	(3889-1)	[43]

Figure 2. Motifs in 4 human TF ChIP-seq data sets. FMotif was tested with four widely used human ChIP-seq data sets for four DNA-binding TFs including CTCF (CCCTC-binding factor [41], named CTCF(*h*)), FoxA1 (hepatocyte nuclear factor 3 α [42]), NRSF (neuron-restrictive silencer factor [2]), and STAT1 (signal transducer and activator of transcription protein [1]). Results from CisFinder and published motifs in literature are shown for comparison. Column definitions are the same as those in Figure 1.
doi:10.1371/journal.pone.0086044.g002

conclude that FMotif has both a higher sensitivity and a higher specificity than CisFinder.

Discussion

In this study, we have proposed a new and fast heuristic enumeration method, FMotif, for extracting motifs from sequences. We have used this method to identify motifs and their binding locations in widely-used large-scale ChIP-seq and ChIP-exo data sets by combining FMotif with a peak-enriched sampling strategy. Our empirical studies have shown that this algorithm is fast and exact when searching for motifs in (*l, d*) samples and has achieved good performance when identifying motifs in ChIP-enriched regions. In addition, the ChIP-enriched sampling strategy worked well on large-scale ChIP-seq and ChIP-exo data sets. It not only allowed us to exclude both noise occurring in lower ChIP-enriched peak regions and false positive motifs contained in background sequences, but also let us take advantage of well-developed motif-finding tools with low-level scalability. However, it should be pointed out that, in general, no method can outperform others under all conditions. FMotif performed faster than SPELLER, WEEDER, and MITRA but used more memory to store mismatched information in suffix trees, and FMotif was much more accurate but much slower than CisFinder. FMotif does, however, provide a good trade-off between time, space, and accuracy.

Motif discovery has been a popular area of study for more than two decades. Many successful motif-finding programs have been developed. The programs are ideal for finding motifs in tens or

hundreds of promoters of co-regulated or homologous genes and for extracting motifs in genome-wide ChIP-enriched regions contained in large-scale ChIP-chip, ChIP-seq, and ChIP-exo data sets. Still, the problem is far from solved due to diversity in gene expression/regulation and the low specificity of binding sites. With the advance of high-throughput and high-resolution sequencing techniques like ChIP-exo, researchers have an increasing number of tools for studying gene regulation on a genomic scale. This will make the motif-finding problem easier to solve. Using advanced techniques such as ChIP-exo, it is possible to acquire new knowledge of regulatory binding sites. This will not only be beneficial for understanding the mechanisms of gene regulation, but also for creating a proper computational model that will replace (*l, d*) models and PWM matrix profiles for motif representation.

Materials and Methods

The (*l, d*) Motif Search and Suffix Tree

A transcription factor binds to specific DNA sequences and is involved in controlling the transcription of genetic information from DNA to mRNA. The actual DNA regions bound by a TF usually range in size from 8–10 to 16–20 bps and display a short motif, but differ by a few nucleotides from one another. The computational problem is to determine such a motif by analyzing a set of sequences that contain instances of the motif.

In current literature, there are two main approaches to motif representation. The first involves using a motif profile character-

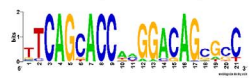




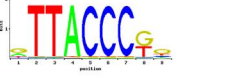
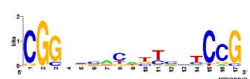
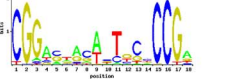


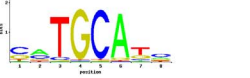
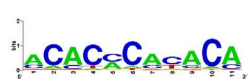

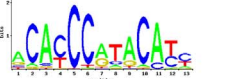



Data Set (Nb)	FMotif (#-Rank)	CisFinder (#-Rank)	Literature
NRSF/qPCR (83)	 (76-1)	 (54-1)	 [2]
Reb1 (1776)	 (1579-1)	 (200-1)	 [4, 46, 47]
Gal4 (15)	 (14-1)	No-Motif-Found	 [4, 47, 48]
Phd1 (967)	 (1610-1)	 (40-1)	 [4, 47]
Rap1 (576)	 (896-1)	 (113-1)	 [4, 47]
CTCF(<i>h</i> -exo) (35161)	 (12919-2)	 (1633-1)	 [43]

Figure 3. FMotif sensitivity. FMotif sensitivity was measured using an NRSF-positive TFBS set (NRSF/qPCR), which was composed of 83 binding sites verified by qPCR [2], four yeast DNA-binding TFs (Reb1, Gal4, Phd1, and Rap1), and one human TF (CTCF) ChIP-exo data sets. Results from CisFinder and published motifs in literature are shown for comparison. Column definitions are the same as those in Figure 1. doi:10.1371/journal.pone.0086044.g003

ized by a PWM $[p_{j,k}]_{l \times 4}$. The PWM records the probability of an observed nucleotide k ($k \in \{A, C, G, T\}$) at position j ($j = \{1, 2, \dots, l\}$) for all aligned sites, where l is the length of the motif. Numerous programs have been developed to maximize the score of a PWM by measuring, for example, the information content of a PWM:

$$IC = - \sum_{i,j} p_{i,j} \cdot \log(p_{i,j}/p_b)$$

where p_b is the background frequency for the nucleotide, which measures motif conservation [7]. Using the second approach, one can characterize a motif as an l -length consensus string and describe it using the most frequent nucleotide in each position of all aligned sites under the assumption that each sequence contains zero or one motif instance with up to d or exactly d mutations within the motif. Finding (l, d) motifs with exactly d mutations is more challenging than finding (l, d) motifs with up to d mutations, and algorithms designed for the former can usually be directly used to find the latter. In addition, profile-based optimization methods, e.g., CONSENSUS and MEME, have failed to find (l, d) motifs such as (15, 4), where a 15-bp motif is planted into 20 sequences, each 600 bps in length with exactly 4 mismatches [19]. Thus, in this study we focus on designing a fast and exact algorithm to find (l, d) motifs with exactly d mutations on (l, d) samples.

When searching for the exact motifs contained in an (l, d) sample, it is customary to perform an exhaustive search for all potential l -mers and verify their occurrence in the entire sample set. When using SPELLER and WEEDER to perform fast l -mer substring searching in a sequence set, a suffix tree structure is used to index sequences. A suffix tree presents the suffixes of a given string or a given set of strings in a way that allows for a very fast implementation of string operations. An example of a classic suffix tree for the string GAGAC is shown in Figure 4a. When the suffix tree of a string with length L is constructed, searching for a substring of length l ($l \leq L$) in the string only requires time proportional to $O(l)$ instead of $O(L)$.

Nevertheless, it is still time consuming to perform an exhaustive motif search in a suffix tree of sequences because the search space (shown as a four-branch tree in Figure 4b) can be up to 4^l in size. With the increase in degenerative value d , the valid instances of a motif in the suffix tree will increase dramatically. Therefore, SPELLER can handle only short (l, d) motifs with $l \leq 13$ and $d \leq 3$. In order to increase the speed of SPELLER, WEEDER introduces an error ratio ε ($\varepsilon \approx d/l$) to narrow the search space such that for all $i \in \{1, 2, \dots, l\}$, the number of mismatches between the first i nucleotides of a candidate l -mer motif and the first i nucleotides of a valid instance of the motif is at most εi . The strategy can quickly discard l -mers in the search space that do not satisfy this restriction. However, not all motif occurrences satisfy this restriction, and therefore the real motif may be missed by the algorithm. WEEDER lowers the occurrence frequency $q \leq Q \leq N$ to make sure that the true motif will not be missed. Still, WEEDER is an almost exact algorithm. What's more, with the decrease of q , WEEDER's run time will increase dramatically [14]. Therefore, a fast and exact (l, d) motif search method is needed.

Mismatched Suffix Trees and FMotif

SPELLER and WEEDER use a depth-first search to scan the entire pattern space. If an i -mer along the pattern tree has enough instances in the suffix tree of sequences, the i -mer can grow up to 4

$(i+1)$ -mers in the next layer of the pattern tree (see Figure 4b). Otherwise, the end node of an i -mer in the pattern tree will not be allowed to grow and the sibling nodes of the i -mer will be checked. If any of the sibling nodes can grow to the $(i+1)$ -th layer in the pattern tree, the search process will go down to the $(i+1)$ -th layer of the pattern tree in a depth-first manner. Otherwise, it will backtrack to the uncle nodes (the siblings of parent nodes) of the i -mer in the $(i-1)$ -th layer of the pattern tree and so forth. The algorithms will end at the longest l -mer or l -mers in the pattern tree. The difference between SPELLER and WEEDER is that WEEDER reduces the number of possible instances of a motif by restricting its mutation locations such that the valid paths on the pattern tree are sharply reduced.

We discovered that for finding motifs with length l , both SPELLER and WEEDER must check each i -mer ($i \leq l$) in the pattern space with depth-first order and count the variants of the i -mer in the suffix tree of sequences from the root to layer i . The suffix tree is scanned one time for each i -mer pattern. Thus, the algorithms scan the suffix tree an increasing number of times with the increase of i . Actually, the mismatch information in layer i of a suffix tree can be used to search $(i+1)$ -mers in the pattern space. In this work we constructed a new suffix tree structure with mismatch information, called a mismatched suffix tree, for each sequence. Using these trees, we took advantage of the mismatch information in the i -th layer of the trees to verify $(i+1)$ -mers in the pattern space and then updated the mismatch information in the $(i+1)$ -th layer. In this way we were able to implement a depth-first search on the pattern space and mismatched suffix trees simultaneously, which avoided a large number of repeated scans on the suffix trees of sequences.

For instance, when searching occurrences of (4, 1) motifs in the sequence GAGAC, we started from the root of the pattern tree represented as $P_0 = \ominus$ in Figure 5a and initialized the mismatched suffix tree for the sequence GAGAC. We then checked the occurrences of pattern $P_1 = A$ with up to 1 mismatch in the mismatched suffix tree and found that all nodes in the first layer have 0 or 1 mismatch(es) with P_1 . Next, we updated the mismatch value along the valid nodes and linked all of these nodes by points (see Figure 5b). We subsequently performed a depth-first search again and arrived at the pattern $P_2 = AA$. We updated mismatch information for all child nodes of the nodes in the link set in the first layer by using the mismatch information of those nodes in the link set and found all nodes in the second layer had 1 mismatch with the pattern $P_2 = AA$. We updated the mismatch value along the valid nodes in the second layer and linked all of these nodes by points to form a new link set (see Figure 5c). Then, we moved to the pattern $P_3 = AAA$ in a depth-first manner and updated mismatch information of all child nodes of the nodes in the newly generated link set by using the mismatch information of those nodes in the new link set. We found that only the child node A, representing the 3-mer AGA from the root to node A in the third layer of the suffix tree, had 1 mismatch with the pattern $P_3 = AAA$ (see Figure 5d). Other nodes with 2 mismatches did not need to be updated and checked for the longer pattern $AAAA$. We found that the child node of the node A in the third layer did not satisfy the 1-mismatch restriction with the pattern $AAAA$, so we looked at the pattern $P_4 = AAAC$ and found a (4, 1) occurrence of P_4 (see Figure 5e). We then went to the patterns $AAAG$ and $AAAT$ and found no occurrence of these patterns in the sequence GAGAC. We backtracked to the pattern $P_3 = AAC$ and updated the mismatch information in the third layer by using the mismatch information of their parent nodes in link set of the second layer. There we found that only node C in the third layer satisfies the restriction (see Figure 5f), but that node C has no child. We then

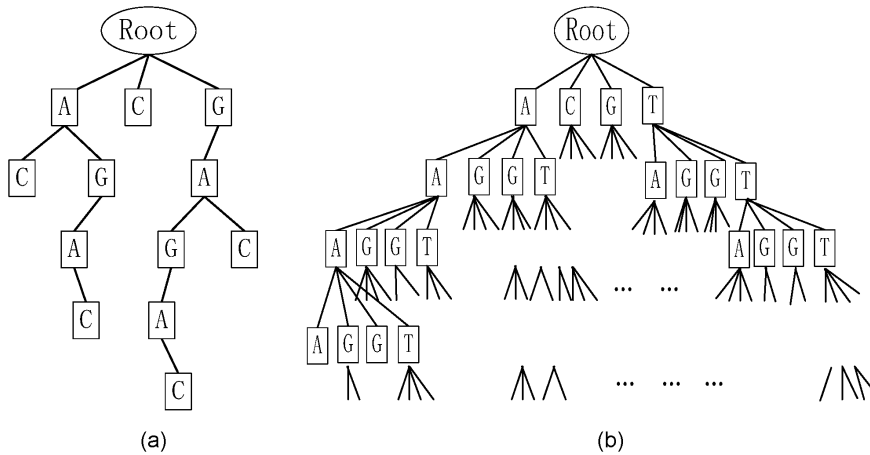


Figure 4. An example of a suffix tree and a tree representation of pattern space. (a) The suffix tree of the sequence GAGAC. (b) A tree representation of pattern space in the search for an (l, d) motif. doi:10.1371/journal.pone.0086044.g004

backtracked to pattern *AAG* and continued the process as before until we found all occurrences for each $(4, 1)$ motif. The details of the pattern search and mismatched suffix tree construction are shown in the subroutine $PatternOnTree(P_k, d, T, List(P_{k-1}, T))$, where T is the mismatched suffix tree for a sequence, P_k is the node currently being processed (representing a k -mer pattern) in the k -th layer of the pattern tree, $List(P_{k-1}, T)$ is the link set representing all valid occurrences of the $(k-1)$ -mer pattern represented by the node P_{k-1} in the $(k-1)$ -th layer of the pattern tree, and $MMC(n_{jk})$ is mismatch value of the pattern represented by P_k compared with the substring represented by the node n_{jk} in the tree T .

```

PatternOnTree( $P_k, d, T, List(P_{k-1}, T)$ )
  Initialize  $List(P_k, T) = \emptyset$ ;
  for  $n_{i,k-1} = \text{head node to tail node of } List(P_{k-1}, T)$  do
    for each  $n_{jk} \in NS$  do ( $NS$  is the child node set of the node  $n_{i,k-1}$ )
      if  $n_{jk} = P_k$ , then
         $MMC(n_{jk}) = MMC(n_{i,k-1})$ 
      else
         $MMC(n_{jk}) = MMC(n_{i,k-1}) + 1$ ;
      if  $MMC(n_{jk}) \leq d$  then
        add  $n_{jk}$  to  $List(P_k, T)$ ;
  return  $List(P_k, T)$ ;

```

```

MotifFinding( $P_k, TS, List(P_k, TS), q, d, l_{max}$ )
  Initialize  $str = ACGT$ ;
  if  $k > l_{max}$  or  $List(P_k, TS) = \emptyset$  then
    return;
  for  $j = 1 \dots 4$  do
     $P_{k+1} = P_k + str[j]$ ;
     $FailureCount = 0$ ;
    for each tree  $T_i$  in the tree set  $TS$  do
       $List(P_{k+1}, T_i) = PatternOnTree(P_{k+1}, d, T_i, List(P_k, T_i))$ ;
      if  $List(P_{k+1}, T_i) = \emptyset$  then
         $FailureCount++$ ;
    if  $FailureCount > |TS| - q$  then
      break;
    if  $FailureCount \leq |TS| - q$  then
      Output( $P_{k+1}$ );
  MotifFinding( $P_{k+1}, TS, List(P_{k+1}, TS), q, d, l_{max}$ )

```

For each set of sequences, we counted the number of occurrences of a potential pattern in all sequences instead of just one sequence shown in subroutine $PatternOnTree(P_k, d, T, List(P_{k-1}, T))$. If the number of occurrences was larger than the threshold of occurrence frequency q , it was reported as a potential pattern. The subroutine for counting occurrences of a $(k+1)$ -mer pattern, represented by the node P_{k+1} in the $(k+1)$ -th layer of the pattern tree, is shown in $MotifFinding(P_k, TS, List(P_k, TS), q, d, l_{max})$, where l_{max} is the maximum length allowed for a motif, TS is the set of mismatched suffix trees for all sequences, $List(P_k, TS) = \cup_{i=0}^N List(P_k, T_i)$, and N is the number of total sequences.

The entire process of finding motifs with at least q occurrences in a set of sequences is shown below. Additionally, since there may be many motifs that satisfy the quorum restriction q , we sorted all potential motifs according to their statistical significance using the method in [14,15]. We reported the top n significant motifs and their occurrences as output, where (i, j) indicates an instance of a motif starting at the j -th position of the i -th sequence s_i .

The FMotif Algorithm

- 1) Initialize a mismatched suffix tree T_i like the one shown in Figure 5a, $i = 1, 2, \dots, N$;
- 2) Initialize $List(\emptyset, T_i) = \&tempnode$, where $\&tempnode$ is the pointer of a temporary node, $i = 1, 2, \dots, N$;
- 3) Input q, d, l_{max}, n ;
- 4) $MotifFinding(\emptyset, TS, List(\emptyset, TS), q, d, l_{max})$;
- 5) Rank the found motifs according to their significance scores;
- 6) Output the top n motifs, their instances, and the positions (i, j) of these instances.

According to our empirical study, FMotif is capable of increasing the speed of the algorithms SPELLER and WEEDER without loss of accuracy. In addition, we used the WEEDER strategy to further decrease the search space by allowing mismatches occurring at most ϵi times with an increase in i . This strategy decreased FMotif's run time but caused problems during the tuning of the parameter q and resulted in a loss of accuracy.

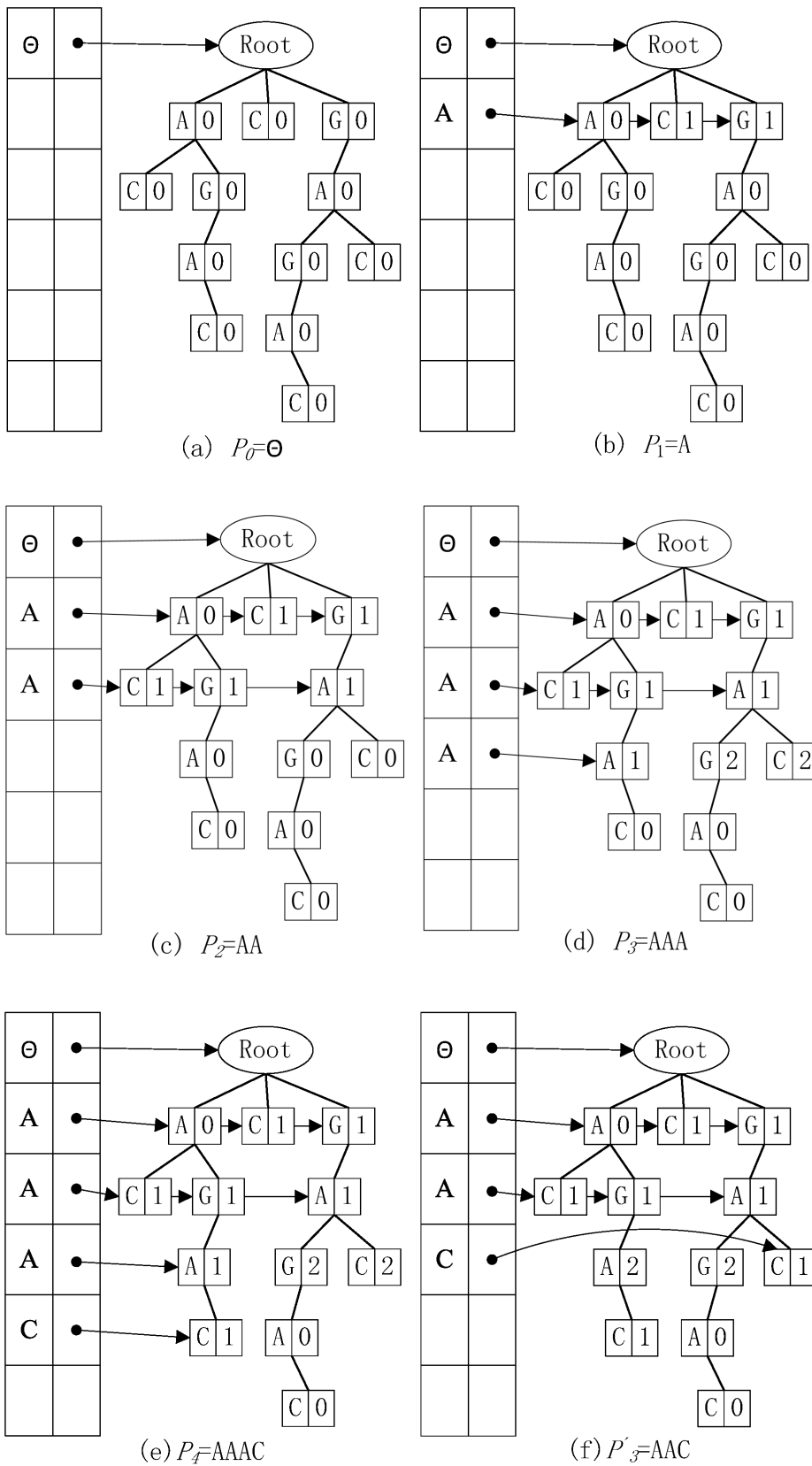


Figure 5. An example of a (4, 1) motif search using FMotif. Figures (a)–(f) illustrate the search process of (4, 1) motifs on the mismatched suffix tree of the sequence GAGAC.
doi:10.1371/journal.pone.0086044.g005

Author Contributions

Conceived and designed the experiments: CJ YW. Performed the experiments: CJ YW. Analyzed the data: CJ MBC. Contributed reagents/materials/analysis tools: YL HL. Wrote the paper: CJ MBC.

References

- Robertson G, Hirst M, Bainbridge M, Bilenky M, Zhao Y, et al. (2007) Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nat Methods* 4: 651–657.
- Johnson DS, Mortazavi A, Myers RM, Wold B (2007) Genome-wide mapping of in vivo protein-DNA interactions. *Science* 316: 1497–1502.
- Wilbanks EG, Facciotti MT (2010) Evaluation of algorithm performance in ChIP-seq peak detection. *PLoS One* 5: e11471.
- Rhee, S H, Pugh BF (2011) Comprehensive genome-wide protein-DNA interactions detected at single-nucleotide resolution. *Cell* 147: 480–483.
- Zhao Y, Stormo GD (2011) Quantitative analysis demonstrates most transcription factors require only simple models of specificity. *Nat Biotechnol* 29: 1408–1419.
- Zambelli F, Pesole G, Pavese G (2013) Motif discovery and transcription factor binding sites before and after the next-generation sequencing era. *Brief Bioinform* 14: 225–237.
- Hertz GZ, Stormo GD (1999) Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics* 15: 563–577.
- Bailey TL, Elkan C (1994) 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*, 1994, Menlo Park, CA. 28–36.
- Lawrence CE, Altschul SF, Boguski MS, Liu JS, Neuwald AF, et al. (1993) Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science* 262: 208–214.
- Hughes JD, Estep PW, Tavazoie S, Church GM (2000) Computational identification of cisregulatory elements associated with groups of functionally related genes in *Saccharomyces cerevisiae*. *Journal of Molecular Biology* 296: 1205–1214.
- Buhler J, Tompa M (2002) Finding motifs using random projections. *Journal of Computational Biology* 9: 225–242.
- Gang L, Chan TM, Leung KS, Lee KH (2010) A cluster refinement algorithm for motif discovery. *IEEE/ACM Trans on Computational Biology and Bioinformatic* 7: 654–668.
- Sagot MF (1998) Spelling approximate repeated or common motifs using a suffix tree. *Proceedings of LATIN'98: Theoretical Informatics, LNCS 1380*: 111–127.
- Pavese G, Mauri G, Pesole G (2001) An algorithm for finding signals of unknown length in DNA sequences. *Bioinformatics* 17: 207–214.
- Pavese G, Mereghetti P, Mauri G, Pesole G (2004) Weeder Web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. *Nucleic Acids Research: Web Server Issue* 32: W199–W203.
- Eskin E, Pevzner P (2002) Finding composite regulatory patterns in DNA sequences. *Bioinformatics* 18: 354–363.
- Chin YL, Leung CM (2005) Voting algorithms for discovering long motifs. In: *Proceedings of the Third Asia-Pacific Bioinformatics Conference*, 2005, Singapore. 261–271.
- Davila J, Balla S, Rajasekaran S (2007) Fast and practical algorithms for planted (*l, d*) motif search. *IEEE/ACM Trans On Computational Biology and Bioinformatics* 4: 544–552.
- Pevzner P, Sze S (2000) Combinatorial approaches to finding subtle signals in DNA sequences. In: *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, 2000, California, USA. 269–278.
- Ho ES, Jakubowski CD, Gunderson SI (2009) iTriplet, a rule-based nucleic acid sequence motif finder. *Algorithms Molecular Biology* 4: doi:10.1186/1748-7188-4-14.
- Huang CW, Lee WS, Hsieh SY (2010) An improved heuristic algorithm for finding motif signals in DNA sequences. *IEEE/ACM Trans On Computational Biology and Bioinformatics* 8: 959–975.
- Kuksa PP, Pavlovic V (2010) Efficient motif finding algorithms for large-alphabet inputs. *BMC Bioinformatics* 1: S1.
- Sun HQ, Low MYH, Hsu WJ, Rajapakse JC (2010) RecMotif: a novel fast algorithm for weak motif discovery. *BMC Bioinformatics* 11: S8.
- Tompa M (2005) Assessing computational tools for the discovery of transcription factor binding sites. *Nature Biotechnology* 23: 137–144.
- Machanic P, Bailey TL (2011) MEME-ChIP: motif analysis of large DNA datasets. *Bioinformatics* 27: 1696–1697.
- Bailey TL (2011) DREME: motif discovery in transcription factor ChIP-seq data. *Bioinformatics* 27: 1653–1659.
- Heeringen SJV, Veenstra GJ (2011) GimmeMotifs: a de novo motif prediction pipeline for ChIPsequencing experiments. *Bioinformatics* 27: 270–271.
- Reid JE, Wernisch L (2011) STEME: efficient EM to find motifs in large data sets. *Nucleic Acids Res* 38: e126.
- Kulakovskiy IV, Boeva VA, Favorov AV, Makeev VJ (2010) Deep and wide digging for binding motifs in ChIP-seq data. *Bioinformatics* 26: 2622–2623.
- Hu M, Yu J, Taylor JM, Chinnaiyan AM, Qin ZS (2010) On the detection and refinement of transcription factor binding sites using ChIP-seq data. *Nucleic Acids Res* 38: 2154–2167.
- Guo Y, Mahony S, Gifford DK (2012) High resolution genome wide binding event finding and motif discovery reveals transcription factor spatial binding constraints. *PLoS Comput Biol* 8: e1002638.
- He Y, Zhang Y, Zheng G, Wei C (2012) CTF: a CRF-based transcription factor binding sites finding system. *BMC Genomics* 13: S18.
- Thomas-Chollier M, Herrmann C, Defrance M, Sand O, Thieffry D, et al. (2012) RSAT peak-motifs: motif analysis in full-size ChIP-seq datasets. *Nucleic Acids Res* 40: e31.
- Sharov AA, Ko MSH (2009) Exhaustive search for over-represented DNA sequence motifs with CisFinder. *DNA Research* 16: 261–273.
- Ma X, Kulkarni A, Zhang Z, Xuan Z, Serfling R, et al. (2012) A highly efficient and effective motif discovery method for ChIP-seq/ChIP-chip data using positional information. *Nucleic Acids Res* 40: e50.
- Keich U, Pevzner P (2002) Subtle motif: defining the limits of finding algorithms. *Bioinformatics* 18: 1382–1390.
- Zia A, Moses AM (2012) Towards a theoretical understanding of false positives in DNA motif finding. *BMC Bioinformatics* 13: doi: 10.1186/1471-2105-13-151.
- Boeva V, Surdez D, Guillon N, Tirode F, Fejes A, et al. (2010) De novo motif identification improves the accuracy of predicting transcription factor binding sites in ChIP-Seq data analysis. *Nucleic Acids Res* 38: e126.
- Zhang Y, Liu T, Meyer CA, Eickhout J, Johnson D, et al. (2008) Model-based analysis of ChIPSeq (MACS). *Genome Biology* 9: R137.
- Chen X, Xu H, Yuan P, Fang F, Huss M, et al. (2008) Integration of external signaling pathways with the core transcriptional network in embryonic stem cells. *Cell* 133: 1106–1117.
- Barski A, Cuddapah S, Cui K, Roh T, Schones D, et al. (2007) High-resolution profiling of histone methylations in the human genome. *Cell* 129: 823–837.
- Lupien M, Eickhout J, Meyer CA, Wang Q, Zhang Y, et al. (2008) FoxA1 translates epigenetic signatures into enhancer-driven lineage-specific transcription. *Cell* 132: 958–970.
- Jothi R, Cuddapah S, Barski A, Cui K, Zhao K (2008) Genome-wide identification of in vivo protein-DNA binding sites from ChIP-Seq data. *Nucleic Acids Res* 36: 5221–5231.
- Jothi R, Cuddapah S, Barski A, Cui K, Zhao K (2012) Breast cancer risk-associated SNPs modulate the affinity of chromatin for FOXA1 and alter gene expression. *Nat Genet* 44: 1191–1198.
- Crooks GE, Hon G, Chandonia JM, Brenner SE (2004) WebLogo: a sequence logo generator. *Genome Research* 14: 1188–1190.
- Badis G, Chan ET, van Bakel H, Pena-Castillo L, Tillo D, et al. (2008) A library of yeast transcription factor motifs reveals a widespread function for Rsc3 in targeting nucleosome exclusion at promoters. *Mol Cell* 32: 878–887.
- Spivak AT, Stormo GD (2012) ScerTF: a comprehensive database of benchmarked position weight matrices for *Saccharomyces* species. *Nucleic Acids Res* 40: D162–168.
- Pachkov M, Erb I, Molina N, van Nimwegen E (2007) SwissRegulon: a database of genome-wide annotations of regulatory sites. *Nucleic Acids Res* 35: D127–131.