ThIEF: Finding Genome-wide Trajectories of Epigenetics Marks*

Anton Polishko[1], Md. Abid Hasan[2], Weihua Pan[3], Evelien M. Bunnik[4], Karine Le Roch[5], and Stefano Lonardi[6]

1 **Department of Computer Science, University of California, Riverside, CA, USA**
2 **Department of Computer Science, University of California, Riverside, CA, USA**
3 **Department of Computer Science, University of California, Riverside, CA, USA**
4 **University of Texas Health Science Center, San Antonio, TX, USA**
5 **Department of Cell Biology, University of California, Riverside CA, USA**
6 **Department of Computer Science, University of California, Riverside, CA, USA**

Abstract

We address the problem of comparing multiple genome-wide maps representing nucleosome positions or specific histone marks. These maps can originate from the comparative analysis of ChIP-Seq/MNase-Seq/FAIRE-Seq data for different cell types/tissues or multiple time points. The input to the problem is a set of maps, each of which is a list of genomics locations for nucleosomes or histone marks. The output is an alignment of nucleosomes/histone marks across time points (that we call *trajectories*), allowing small movements and gaps in some of the maps. We present a tool called ThIEF (TrackIng of Epigenetic Features) that can efficiently compute these trajectories. ThIEF comes into two "flavors": ThIEF:Iterative finds the trajectories progressively using bipartite matching, while ThIEF:LP solves a k-partite matching problem on a hyper graph using linear programming. ThIEF:LP is guaranteed to find the optimal solution, but it is slower than ThIEF:Iterative. We demonstrate the utility of ThIEF by providing an example of applications on the analysis of temporal nucleosome maps for the human malaria parasite. As a surprisingly remarkable result, we show that the output of ThIEF can be used to produce a supervised classifier that can accurately predict the position of stable nucleosomes (i.e., nucleosomes present in all time points) and unstable nucleosomes (i.e., present in at most half of the time points) from the primary DNA sequence. To the best of our knowledge, this is the first result on the prediction of the dynamics of nucleosomes solely based on their DNA binding preference. Software is available at `https://github.com/ucrbioinfo/ThIEF`

1998 ACM Subject Classification G.2.1 Combinatorics, I.1.2 Algorithms, J.3 Life and Medical Sciences

Keywords and phrases Nucleosomes, Histone Marks, Histone Tail Modifications, Epigenetics, Genomics

Digital Object Identifier 10.4230/LIPIcs.WABI.2017.19

* This work was partially supported by NSF IIS-1302134 to SL and KLR and NSF III-1526742 to SL.

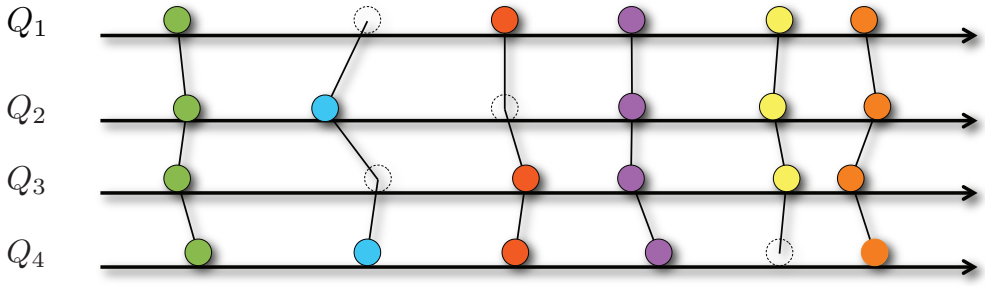

Figure 1 An illustration of the problem of aligning epigenetic features on four maps Q_1, Q_2, Q_3, Q_4; the input is a set of locations for the feature of interest (e.g., CHiP-seq peaks for nucleosome or specific histone marks) illustrated as circles here; the output is an assignment of features to trajectories, in a way that most parsimoniously explain the data; dotted circles indicate gaps or "missing features"

1 Introduction

Advancements in high-throughput DNA sequencing technology has enabled life scientists to carry out increasingly large-scale experiments. In genomics and epigenetics, it is relatively common to run multiple genome-wide experiments: for example, one can use ChIP-Seq/MNase-Seq/FAIRE-Seq/NOMe-Seq to obtain nucleosome levels or specific histone tail post-translational modifications (e.g., methylation, acetylation, phosphorylation, ubiquitination) at different cell types/tissues (which allows to understand cell type-specific marks), or at different time points for a particular cell process cycle (which allows to explore the dynamics of epigenetic marks). The ENCODE project [8], modENCODE [10, 41], phychENCODE [1], and the NIH Roadmap Epigenomics Project [2] are notable examples of these efforts. Similar comparative analysis of epigenetic marks could be carried out for a set of closely related organisms (e.g., human-chimp) if the correspondences between the genomes are known (e.g., using liftover by [20]).

Epigenetic maps are usually obtained by (i) sequencing a DNA sample enriched for a feature of interest, (ii) mapping short reads to the reference genome and (iii) running a peak-calling algorithm (e.g., MACS/MACS2 [50], NOrMAL [37], Puffin [36]). In this paper we focus on the fundamental question on how to compare multiple genome-wide epigenetic maps, when features are expected to shift or be missing. Our model allows the possibility of the nucleosome of physically sliding along the genome or compensate for the noise in the peak detection process. In the example in Figure 1, the objective is to align four maps. Circles represent features to align; dashed circle mark the gap; trajectories are indicated with solid lines, which represent the most "likely" explanation of the data.

We propose to compare epigenetic maps by aligning them in a similar way we align DNA sequences. We arrange multiple nucleosome/feature maps on top of each other, with the objective to build a *trajectory* for each individual nucleosomes/feature across time points or cell types, in a way that a total cost (i.e., total "traveled distance" in the case of nucleosomes) is minimized. We call such a set of trajectories an *alignment*: similarly to multiple sequence alignment we allow "insertion" or "deletions" of nucleosomes/features at specific time points or cell types. For instance, Figure 2 illustrates the output of ThIEF in the IGV browser for a region of chromosome 10 of *P. falciparum*. ThIEF aligned eight nucleosome maps over multiple time points, where each trajectory is assigned a unique ID, using alternating colors. Observe that in some trajectories (e.g., the one with ID 10_4546), nucleosomes are stable, while in others (e.g., the one with ID 10_4547) nucleosomes are evicted then bind later to

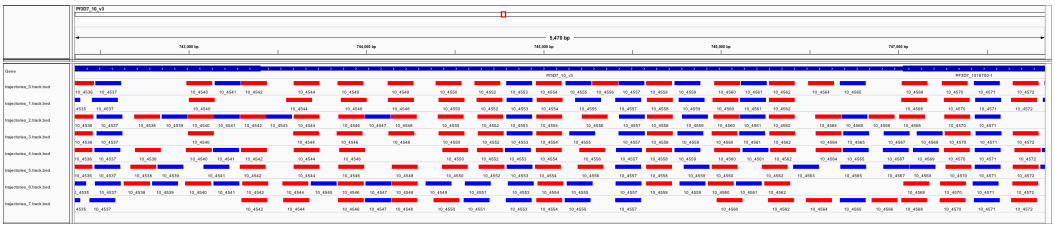

Figure 2 IGV snapshot of nucleosomes (red and blue rectangles – 147bps long) in region [742,356-747,829] of chromosome 10 of *P. falciparum* at eight time points; ThIEF assigns nucleosomes to trajectories which can be identified by an integer ID (zoom in to see them), and displayed in alternating colors

the same location. We should note that it not easy to tag individual nucleosomes and image them under microscopes (see, e.g., [47]), so there is no way with current technology to obtain the "true" trajectories of nucleosomes genome-wide.

2 Problem definition

We define a *epigenetic map* $Q = \{f_1, \ldots, f_n\}$ as a set of n epigenetic features f_i, where a feature could be a nucleosome or a specific histone mark. Each feature $f \in Q$ is described by vector $f = (\mu, a_1, \ldots, a_l)$ where μ is the genomic coordinate of f in the genome (e.g., chromosome number and position in the chromosome) and each a_j, $j = 1, \ldots, l$ is an *attribute* of that feature (e.g., confidence score of a nucleosome, strength of a histone mark, p-value, peak width, "fuzziness" of a nucleosome, etc.). For convenience, we assume that features in Q are ordered by their position μ.

Given k epigenetic maps $Q_1, Q_2, \ldots, Q_k$, the goal is to align them so that the total alignment cost is minimized. For instance when $k = 2$, we either match feature $f \in Q_1$ to a feature $g \in Q_2$ so that we minimize a cost $\Delta(f, g)$ (e.g., some distance between the vectors $f = (\mu^f, a_1^f, \ldots, a_l^f)$ and $g = (\mu^g, a_1^g, \ldots, a_l^g)$) or report that feature $f \in Q_1$ has no match in Q_2 (insertion/deletion). In this latter case we will say that we have an alignment *gap*. For k maps, the cost function is $\Delta(q_1, q_2, \ldots, q_k)$ where q_i is a feature in the i-th map. Note that $\Delta(q_1, q_2, \ldots, q_k)$ is supposed to be defined for all combinations of $q_1, q_2, \ldots, q_k$.

While our framework allows features with $l \geq 0$ attributes, in the rest of the paper we only use the genomic coordinate (i.e., $l = 0$). In that case $\Delta(q_1, q_2, \ldots, q_k)$ is simply the *absolute deviation*, i.e., the sum of absolute distance between the coordinate of each feature and the mean of those k coordinates. As it is done in sequence alignment, we will not allow swaps in the order of genomic features. In other words, we do not allow trajectories to cross each other.

3 Previous work

The comparative analysis of a set of genome-wide epigenetic maps can be challenging, in particular when the number of maps is large. Some bioinformatic tools are available to help users make sense of the data. For instance, BEDTools allows users to compares multiple maps by determining which peaks are overlapping, non-overlapping, or have a minimal fraction of overlap [40]; Galaxy [12, 3, 11] offers a set tools for working with genomic intervals under the "Operate on Genomic Intervals" section; ChromHMM can be also used to summarize multiple epigenetic maps. In ChromHMM, a multivariate hidden Markov Model

allows users to determine *chromatin states* (e.g., active enhancer, heterochromatin, repressed PolyComb, etc.) from multiple histone marks, DNA methylation, DNase I hypersensitive sites, and gene expression [9]. Similarly, Segway uses a dynamic Bayesian network model to identify chromatin patterns associated with transcription start sites, gene ends, enhancers, transcriptional regulator CTCF-binding regions and repressed regions [14, 15]. Recently, EpiCSeg was introduced to address shortcomings of ChromHMM and Segway [30]. In terms of motivations, DGW is the closest work to ours. DGW uses dynamic time warping to align the profiles of ChIP-seq enrichment and to cluster them into groups which share similar shapes [28].

The problem of aligning genomic features from multiple maps is similar to the *multiple sequence alignment* (MSA) problem (see, e.g., [6]). MSA is a central problem in bioinformatics with a wide range of applications (see, e.g., [42, 39, 23, 31]). A very large corpus of literature has been published on MSA and its applications. A direct solution for MSA uses dynamic programming to identify the globally optimal alignment [44, 32]. The majority of efficient methods employ sophisticated heuristics, since the global optimization problem of aligning long sequences is computationally costly (the problem is *NP*-complete [7, 49, 18]). Heuristic methods include progressive alignment construction [13, 46, 45, 25], iterative methods, hidden Markov models, genetic algorithms [34, 33], simulated annealing [21, 16], among others. The major difference between MSA on DNA/proteins and the problem discussed here is that instead of a substitution matrix, we use a cost function Δ defined between all possible combinations of features in the k maps.

The problem of aligning genomic features is also related to the *multi-target tracking* problem or *data association* problem. These latter problems have been known for decades and are extensively studied. These problems arise when there is a need to track features on video sequences, radar scans, etc. The main computational challenge is to overcome scalability when dealing with a large number of snapshots and a relatively small number of objects to track. In our case, we are interested in dealing with a relatively small number of maps and a large number of objects.

4 Methods

We first describe the naïve (greedy) approach and explain its limitations, then we provide two improved algorithms. The first builds iterative approximations of the optimal alignment, the second computes the solution via integer linear programming [4]. To make the second approach feasible to real-world-sized data we applied branch-and-bound technique to reduce the size of the problem. Then, we relax the problem to a linear program, which we solve using the off-the-shelf solver GLPK [29].

4.1 Naïve (greedy) approach

In the naïve algorithm, we consider each feature (at location μ) in the first map and check whether we can match it to features on the other maps such that all these features are located within a window of predefined length w, centered at μ. If a map does not have a feature in the window $[\mu - w/2, \mu + w/2]$, we introduce a gap. When a feature is assigned to a trajectory, we mark it so that it cannot be used for other trajectories. After considering all the features in the first map, we check whether the next map contains any feature that was not marked; if any of the feature is still unmatched the same sliding window approach is employed. We repeat this process until all the features are marked.

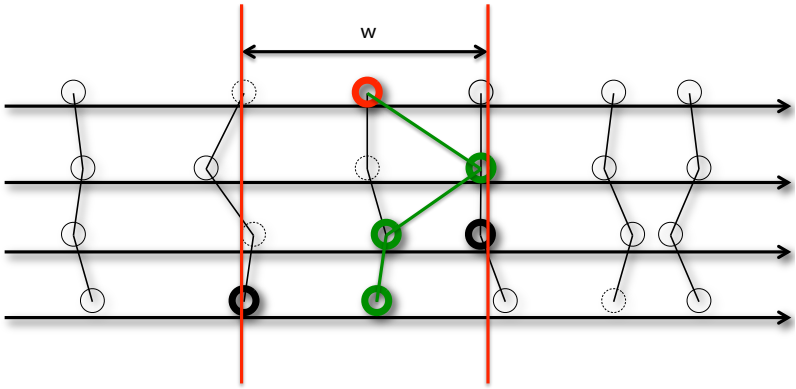

Figure 3 The naïve (greedy) method can create sub-optimal alignments (see text).

The naïve algorithm is fast, but it does not guarantee to produce an optimal solution. First, it relies on the assumption that the alignment score of three or more features is decomposable into the combination of pairwise alignments. Second, the algorithm can make wrong "choices" depending on window size w. Figure 3 illustrates an example of what could go wrong. Circles represent features to align, where a solid circle means that a feature is present, dashed circle means that a feature is missing. The black lines represent the trajectories that we are expected to recover. When the naïve algorithm considers the "red" feature, it will to match it with features marked with thick-line circles. The resulting alignment is shown in green, which is sub-optimal globally.

4.2 ThIEF:Iterative

ThIEF:Iterative iteratively constructs alignments by processing input maps pairwise. The algorithm starts by aligning the first two maps by solving the *weighted bipartite matching* problem. Then, ThIEF:Iterative aligns the resulting alignment between the first two maps to the third map. At each iteration the algorithm solves a bipartite matching problem between the current alignment and the next epigenetic map.

Each node in the bipartite graph is assigned a genomic location. When a genomic feature f is mapped to a node $v_f \in V$, then its location is μ_f. Since the algorithm needs to align alignment to maps, we use the average of the coordinates of the features that belong to an alignment as the "location" of that alignments. In this way, nodes corresponding to partial alignments will have a location attribute and the cost function can be evaluated. Each edge (w, v) is assigned weight $|\mu_w - \mu_v|$.

The bipartite graph must have an equal number of vertices in each feature map (partition), so for every vertex in one partition we introduce a "dummy" vertex in the other one. The presence of dummy nodes also naturally allows gaps: when a feature is matched against a dummy node we are implicitly introducing a gap. The cost of edges connecting "dummy" node to features is the gap penalty δ, and edges "dummy"-"dummy" are not allowed. To solve the $n - to - n$ assignment problem we use the *Hungarian algorithm* [17], which has $\Theta(n^3)$ time complexity.

The total running time of this approach is $\Theta(km^3)$, where k is the number of maps and m is an upper bound on the number of alignments. In the worst case, if we align maps such that features on each map are aligned with corresponding gaps then we could have $m \in O(2^{k-1}n)$. The other disadvantage of this approach is that it does not guarantee optimality, unless the

cost function Δ is decomposable into sum of pairwise costs. In general, the order in which maps are processed can give rise to different (sub-optimal) solutions.

4.3 ThIEF:LP

ThIEF:LP casts the alignment problem as a *weighted k-partite matching* problem. Our problem is slightly more general than the k-partite matching problem because we need to deal with gaps. First ThIEF:LP builds a hyper-graph $H = (V, E)$, where each vertex $v \in V$ represent a genomic feature, and an hyper-edge $e \in E$ connects a subset of vertices (i.e., $e \subseteq V$) representing a possible alignment. By construction, the graph is k-partite: each hyper-edge contains at most one vertex from each partition (feature map). Hyper-edges can skip a partition to model gaps in the alignment.

We build the graph H iteratively. First, we create edges between the nodes in the first two maps (according to the criteria below), then extend them as hyper-edges to the other maps. Given a current (hyper)edge $e \in E$, in order to limit the size of H we extend it to a vertex v in the new partition only if (i) v has a position within $[-\delta, +\delta]$ of the position of the hyper-edge (computed as the average of the position of the nodes that belong to e) and (ii) it does not cross other hyper-edges. Once H is built, we solve the weighted k-partite matching via an integer linear program (ILP) [4], as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{|E|} w_i x_i \\ \text{subject to} \quad & \sum_{i \in S_j} x_i = 1 \qquad j = 1, \ldots, |V| \\ & x_i \in \{0, 1\} \qquad i = 1, \ldots, |E| \end{aligned}$$

where binary variable x_i is associated to hyper-edge $e_i \in E$, S_j is the set of hyper-edges incident to node $v_j \in V$, and weight w_i is the absolute deviation of hyper-edge e_i, i.e., the sum of absolute distance between the coordinate of each feature in e_i and their mean.

The integer program is relaxed to a linear program by allowing each variable to take values in the interval $[0, 1]$. The linear program is solved using the off-the-shelf solver GLPK [29].

5 Experimental results on synthetic data

A synthetic dataset can be described by five parameters, namely (i) the number k of maps, (ii) the number n of features, (iii) the minimum distance d between features, (iv) the probability p of a gap, and (v) the allowed "movement" σ. We generate first a "master" map Q where the features are placed at random locations so that the average distance between adjacent features is uniformly distributed in $[d, 2d]$. We generate the k maps from the initial map Q, as follows. We shift the location of each feature $f \in Q$ by a random quantity drawn from Gaussian distribution with parameters $(0, \sigma)$, then assign f it to a map at the perturbed location. Finally, we replace a feature with a gap with a small probability p. Observe that this procedure might produce alignments that do not satisfy the assumption that trajectories cannot cross.

5.1 Performance analysis

We analyzed the performance of ThIEF:LP and ThIEF:Iterative against the naïve (greedy) approach on several synthetic datasets. We generated a large number of datasets

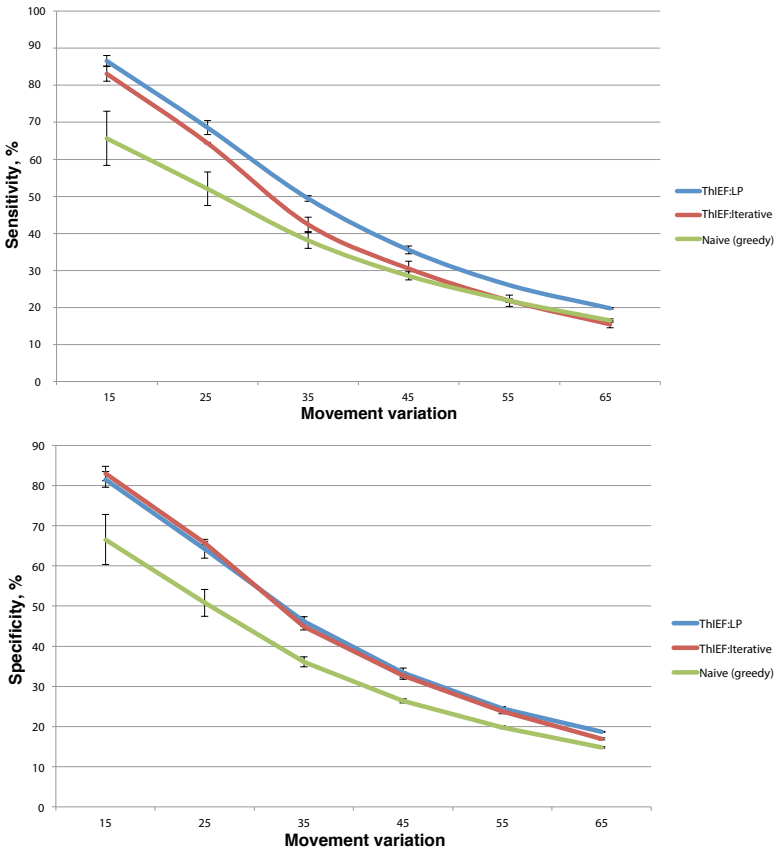

Figure 4 Sensitivity (TOP) and specificity (BOTTOM) of ThIEF:LP and ThIEF:Iterative, and the naïve for several choices of parameter $\sigma = [15, 65]$.

consisting of $k = 3$ maps and $n = 1000$ features, using different values for minimum distance d between features, gap probability p and movement variation σ.

We compared the alignments produced by these tools against the "ground-truth" and measured sensitivity and specificity. Figure 4-LEFT shows the average sensitivity as a function of parameter σ. Observe that ThIEF:LP outperforms the other approaches and ThIEF:Iterative's performance is better then naïve approach (as expected). Also observe that as σ increases, the performance decreases: this can be explained by the fact that with more variability in the location of the features it becomes more likely to have crossing of trajectories, which none of the tools is designed to capture. Specificity analysis shows a similar behavior (see Figure 4-RIGHT). Here ThIEF:LP and ThIEF:Iterative have almost identical performance, which is better than the naïve.

5.2 Execution time

To study the speed of the two algorithms we measured the total execution time on a variety of input datasets. ThIEF:Iterative's time complexity is $\Theta(km^3)$, where k is the number of maps and m is an upper bound on the total number of alignments. The functional dependency between m and n is data-dependent, specifically on how many new alignments are introduced

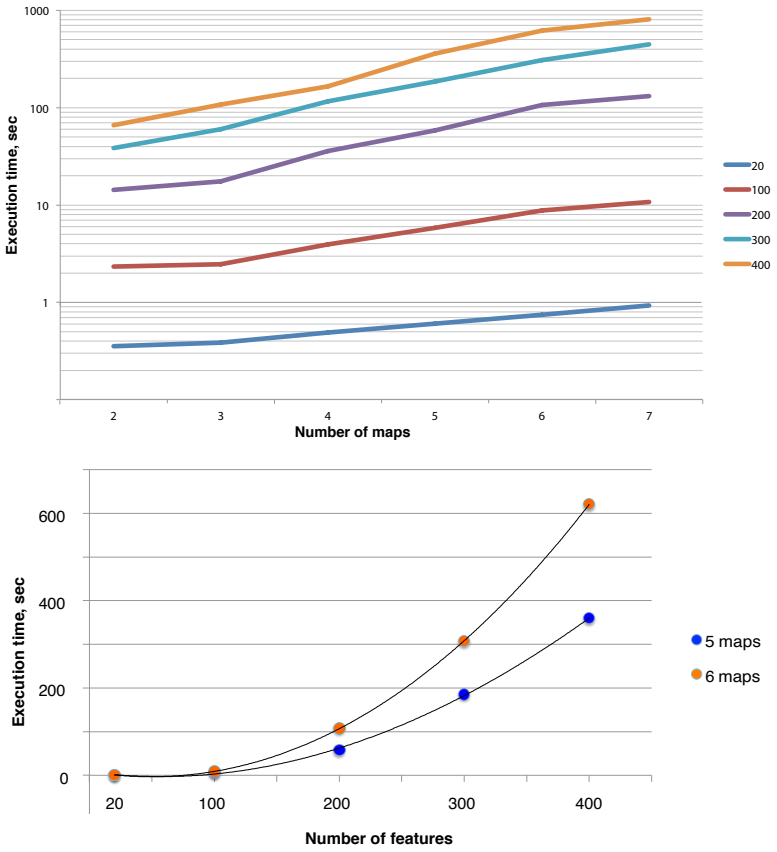

Figure 5 Execution time of ThIEF:Iterative. (LEFT) As a function of the number $k = 2, \ldots, 7$ of maps to align, for different choices of the number n of features (20, 100, 200, 300, 400); (RIGHT) As a function of the number of features (for $k = 5, 6$ maps).

at every iteration (instead of extending existing ones). The worst case is $m \in O(2^{k-1}n)$. As a result, the total worst-case time-complexity could be as bad as $O(2^k n^3)$. Figure 5-LEFT shows the experimental dependency between the execution time of ThIEF:Iterative and the number of maps k. Observe that since the Y axis is log-scale, these experiments confirm that the actual running time is exponential. Figure 5-RIGHT shows the dependency between the execution time of ThIEF:Iterative and the number of features n, for $k = 5$ and $k = 6$ maps. The solid lines are cubic functions of n fitted to the data. These experiments confirm the cubic dependency on number of features in the input.

ThIEF:LP's running time is dominated by the cost of solving a linear program, which in the case of the simplex algorithm, has exponential worst-case running time (although in practice, for the large majority of the instances the time-complexity of simplex is polynomial). The size of the linear program depends on the size of the hyper-graph: in our implementation, the size of the graph can be exponential in k. Figure 6-LEFT shows the dependency between execution time and the number of maps k. Each curve shows a linear trend (note that Y axis is in log-scale). The different shape of the blue curve (twenty features per map) could be explained by the fact that with small inputs the I/O overhead of transferring the data to the GLPK solver dominates the execution time. When we consider the blue curve for at

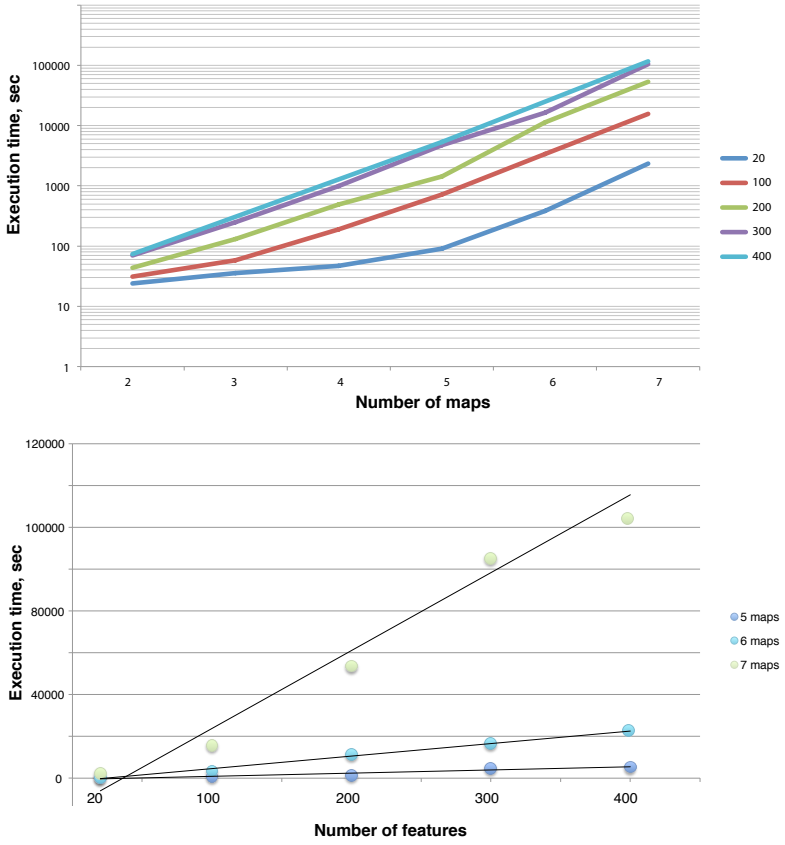

Figure 6 Execution time of ThIEF:LP. (TOP) As a function of the number $k = 2, \ldots, 7$ of maps to align, for different choices of the number n of features (20, 100, 200, 300, 400); (BOTTOM) As a function of the number of features (for $k = 5, 6, 7$ maps).

least five maps the size of the hyper-graph becomes big enough so that solving the linear program dominate the execution time. These experiments support the claim of exponential complexity on the number of maps. Figure 6-RIGHT shows a linear dependency between the execution time and the number of features to track, as expected from the theoretical analysis.

6 Experimental results on real data

As mentioned previously, current imaging technology does not allow one to track nucleosomes or specific histone marks genome-wide over time. Since real data has no "ground-truth" about trajectories that would allows us to objectively evaluate our tool, nor we have found other tools that solve the same problem to which we could compare ThIEF, we decided to demonstrate the utility of ThIEF by developing a supervised classifier that can accurately predict the position of stable and unstable nucleosomes from the primary DNA sequence. We define a nucleosome to be *stable* when it appears in approximately the same position in the genome at all time points of an experiment. We define a nucleosome to be *unstable* when it appears in approximately the same position in at most half of the time points of an experiment. Nucleosomes in the second category are expected to be the most informative

for investigating how chromatin structure affects gene expression. While there is significant amount of work on predicting the binding of nucleosomes from the DNA primary sequence (e.g., [22, 51, 26, 52, 27, 48, 35, 43]), we are not aware of any work that has addressed the problem of predicting whether a nucleosome is expected to be stable or unstable from the DNA sequence.

6.1 Detecting stable and unstable nucleosomes

As an example of application of ThIEF, we analyzed eight nucleosome maps from a study on the human malaria parasite [19]. In this study, synchronized *P. falciparum* parasites were collected at eight different stages of intra-erythrocytic development with five hour intervals (T5-T40). The eight samples were digested to enrich for nucleosome-bound DNA using the MNase protocol, then the digested samples were paired-end sequenced on an Illumina sequencing instrument. Raw reads provided by [19] were mapped to *P. falciparum* 3D7 genome v13.0 (available from `www.plasmoDB.org`) using Bowtie2 [24] allowing a maximum of one mismatch per read. Reads that mapped to multiple locations in the genome, reads that were PCR duplicates, and reads that mapped to ribosomal RNA or transfer RNA were discarded. The final datasets contained about 409 M mapped paired-end reads, with an average read length of 100 bp.

We used PuFFIN [36] to generate nucleosomes positions (for all eight time points) from the aligned reads. Then, we used ThIEF:LP to produce nucleosome trajectories by solving 14 independent optimization problems (one for each chromosome of *P. falciparum*). PuFFIN detected a total of 770,238 nucleosomes, with an average of 96,280 nucleosomes per time point. Nucleosome positions were consistent with the one reported in [19].

ThIEF:LP reported a total of 141,363 trajectories (average of 10,097 trajectories for chromosome). A trajectory generated by ThIEF:LP was considered *stable* if it contained nucleosomes at each time point (i.e., no missing nucleosomes). Trajectories with no more than four nucleosomes (out of eight) were considered as *unstable* (i.e., at least three missing nucleosomes). According to this definition we detected 43,122 stable trajectories (about 31% of the total) and 50,783 unstable trajectories (about 36% of the total). Trajectories that were not stable or unstable were not used in the experiments (about 33% of the total). Chromosomes by chromosomes, the percentage of stable trajectories ranged from 33.23% to 35.39% of the total number of trajectories. The percentage of unstable trajectories for all chromosomes ranged from 33.5% to 40.4%. The distribution of stable and unstable trajectories along the *P. falciparum* chromosomes is shown in Figure 7. Observe that there are regions devoid of stable nucleosomes, and regions very enriched for unstable nucleosomes (e.g., the telomere of chromosome 13).

6.2 A classifier for stable and unstable nucleosomes binding sites

First, we extracted the binding sites from the genome of *P. falciparum* for each trajectory. For each trajectory t labeled stable or unstable, we computed the average position of the nucleosomes in t, then selected 147 bp centered at that position from the malaria genome. Recall that a nucleosome consists of approximately 146-147 bp of DNA wrapped in superhelical turns around a histone octamer complex. Using this procedure we produced a training set composed of 43,122 147bp-long sequences representing stable nucleosomes, and 50,783 147bp-long sequences representing unstable nucleosomes.

We chose a Support Vector Machine (SVM) with radial basis function (RBF) kernel as our binary classifier. We explored many features to use in the classification, including k-mer

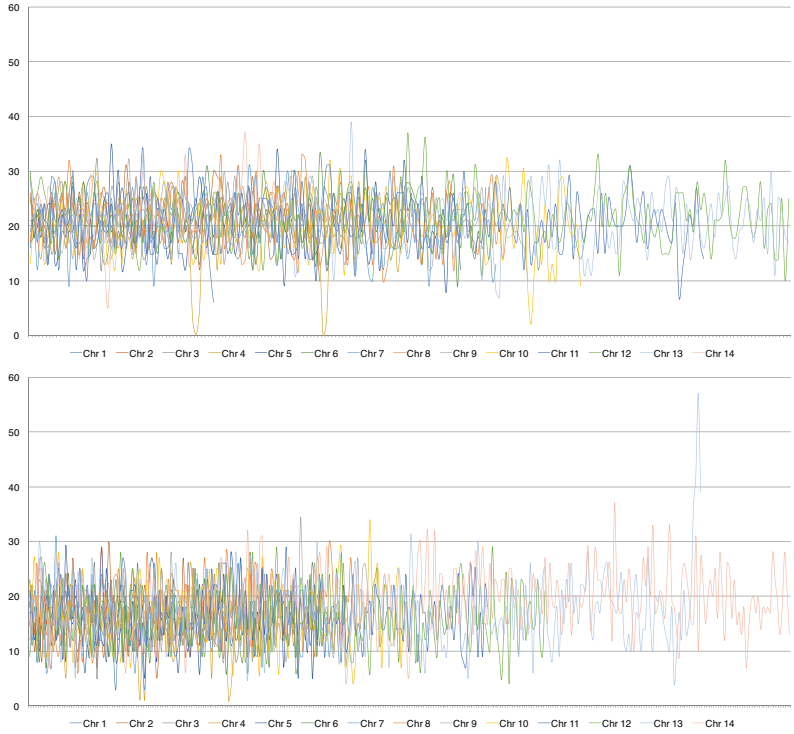

Figure 7 Distribution of stable (TOP) and unstable (BOTTOM) nucleosome trajectories along the 14 human malaria chromosomes (counts in a sliding window of 50Kbp).

distributions for $2 \leq k \leq 5$ and other physico-chemical properties of DNA (e.g., purine-pyrimidine, amino-keto, strong-weak H-bond). By running an extensive set of cross-validation experiments we determined that the five features described next were the most informative.

The first four features were obtained from the 3-mer distribution. For each sequence we collected its 3-mer frequencies, then computed the eigenvalues of four 4×4 matrices corresponding to 3-mers that have a middle nucleotide being `A`, `C`, `G`, and `T`, respectively. We represented each matrix with its leading eigenvalue, for a total of four features. The fifth feature was the DNA stability ΔG which is expressed by in terms of free energies of di-nucleotide stacks ΔG_{KL}, as described in [38]. The stability ΔG is measured by

$$\begin{aligned} \Delta G &= \frac{x_{GC}^2}{4}\left[\sum_{\mathcal{A}} \Delta G_{KL} + \sum_{\mathcal{B}} \Delta G_{KL} - \sum_{\mathcal{C}} \Delta G_{KL}\right] \\ &+ \frac{x_{GC}}{2}\left[\frac{1}{2}\sum_{\mathcal{C}} \Delta G_{KL} - \sum_{\mathcal{B}} \Delta G_{KL}\right] + \frac{1}{4}\sum_{\mathcal{B}} \Delta G_{KL} \end{aligned} \tag{1}$$

where $\mathcal{A} = \{GG, CC, GC, CG\}$, $\mathcal{B} = \{AA, TT, AT, TA\}$ and $\mathcal{C} = \{GA, AG, CT, TC, GT, TG, CA, AC\}$; ΔG_{KL} is the standard melting free energy parameter where di-nucleotide stacks are calculated from stacking free energy parameters ΔG_{KL}^{ST}. Table 1 lists the value of ΔG_{KL}^{ST} and ΔG_{KL}, obtained from [38]. We z-normalized the feature vectors before training the SVM.

Table 2 shows the classification results (precision, accuracy, recall, and area under the ROC curve) when training the SVM on the DNA binding sites of stable/unstable trajectories

Table 1 (LEFT) Stacking free energy parameters ΔG^{ST}_{KL}, (RIGHT) Standard melting free energy parameters ΔG_{KL}.

KL	A	T	G	C
A	-1.11	-1.34	-1.06	-1.81
T	-0.19	-1.11	-0.55	-1.43
G	-1.43	-1.81	-1.44	-2.17
C	-0.55	-1.06	-0.91	-1.44

KL	A	T	G	C
A	-1.04	-1.27	-1.29	-2.04
T	-0.12	-1.04	-0.78	-1.66
G	-1.66	-2.04	-1.97	-2.70
C	-0.78	-1.29	-1.44	-1.97

Table 2 Classification results on the stable/unstable dataset for *P. falciparum*, by training the SVM on the odd-numbered chromosomes, and testing on the even-numbered chromosomes; AUC is the area under the ROC curve.

γ	C	*precision*	*accuracy*	*recall*	*F-score*	AUC
0.5	2	91.32%	79.60%	70.50%	0.795687	0.905418
0.5	8	92.48%	79.18%	68.64%	0.787988	0.908831
2	0.125	92.40%	79.97%	70.23%	0.798028	0.91509
2	0.5	92.59%	79.37%	68.91%	0.79015	0.913866
2	2	93.23%	79.28%	68.19%	0.787704	0.91498
8	0.03125	95.17%	80.32%	68.55%	0.797005	0.924962
8	0.125	94.08%	79.33%	67.58%	0.786582	0.921428
8	0.5	93.95%	79.14%	67.32%	0.784348	0.919447
32	0.007812	96.31%	79.79%	66.70%	0.788176	0.926078
32	0.03125	96.10%	79.39%	66.11%	0.7833	0.929091
32	0.125	94.65%	79.27%	67.00%	0.784592	0.92311
128	0.125	94.72%	79.23%	66.88%	0.78399	0.923386
512	0.125	94.71%	79.03%	66.51%	0.781401	0.923546
2048	0.03125	95.26%	79.04%	66.10%	0.780456	0.926556
8192	0.03125	94.99%	79.07%	66.36%	0.781336	0.925219

for even-numbered chromosomes of *P. falciparum*, and testing it on the odd-numbered chromosomes, for several choices of the penalty parameter C and kernel parameter γ in libSVM [5]. If TP, FP, TN and FN are true positive, false positive, true negative, and false negative, respectively, *precision* is defined as $TP/(TP+FP)$, *accuracy* is $(TP+TN)/(TP+TN+FP+FN)$, *recall* is $TP/(TP+FN)$, and the F-Score (also known as the F_1-score) is $(2\ precision\ \ recall)/(precision\ +\ recall)$. Observe that the SVM classifier makes a prediction for 66%-70% of tested nucleosome binding sites, and the precision of the prediction is very high. In 91%-95% of the predictions, the classifier can correctly determine solely from the DNA sequence whether the nucleosome will be stable or unstable. This is a surprisingly remarkable result.

7 Discussion and conclusion

We described the general problem of aligning multiple genome-wide epigenetic maps. We proposed two novel algorithms for this problem, namely ThIEF:LP and ThIEF:Iterative. The former finds a global optimal solution by constructing a hyper-graph representing the problem and solves it via linear programming. The latter reconstructs the final alignments by computing pair-wise alignments using the Hungarian algorithm. We determined that ThIEF:LP has slightly better sensitivity than ThIEF:Iterative, however the latter ap-

proach is more suitable for aligning a large number of maps. Both tools perform significantly better than the naïve (greedy) approach.

We have also demonstrated how ThIEF can be used in downstream analyses. In our example we used ThIEF to generate nucleosome trajectories for a time-course dataset on *P. falciparum*. The output of ThIEF can be used directly, and for the first time, to label nucleosome trajectories as stable or unstable. As proof of utility, we used the nucleosome-bound DNA sequence (labeled stable or unstable) to train a SVM classifier. Surprisingly, the classifier was able to predict with high accuracy and precision whether a particular 147 bp-long sequence is likely to contain a stable or unstable nucleosome. To the best of our knowledge, this is the first result on the prediction of the dynamics of nucleosomes solely based on their DNA binding preference.

Acknowledgements. The authors thank Christian Shelton (UCR) and Suhn Kyong Rhie (USC) for helpful discussions on this manuscript, and Xueqing (Maggie) Lu (UCR) for providing the nucleosome datasets.

References

1 Schahram Akbarian, Chunyu Liu, et al. The PsychENCODE project. *Nature Publishing Group*, 2015.

2 Bradley E. Bernstein, John A. Stamatoyannopoulos, Joseph F. Costello, Bing Ren, Aleksandar Milosavljevic, Alexander Meissner, Manolis Kellis, Marco A. Marra, Arthur L. Beaudet, Joseph R. Ecker, Peggy J. Farnham, Martin Hirst, Eric S. Lander, Tarjei S. Mikkelsen, and James A. Thomson. The NIH roadmap epigenomics mapping consortium. *Nat Biotech*, 28(10):1045–1048, 10 2010.

3 Daniel Blankenberg, Gregory Von Kuster, Nathaniel Coraor, Guruprasad Ananda, Ross Lazarus, Mary Mangan, Anton Nekrutenko, and James Taylor. Galaxy: A web-based genome analysis tool for experimentalists. *Current protocols in molecular biology*, pages 19–10, 2010.

4 Yuk Hei Chan and Lap Chi Lau. On linear and semidefinite programming relaxations for hypergraph matching. *Mathematical programming*, 135(1-2):123–148, 2012.

5 Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011.

6 Robert C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, 32(5):1792–7, January 2004.

7 Isaac Elias. Settling the intractability of multiple alignment. *Journal of Computational Biology*, 13(7):1323–1339, 2006.

8 ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. *Nature*, 2012.

9 Jason Ernst and Manolis Kellis. ChromHMM: automating chromatin-state discovery and characterization. *Nature Methods*, 2012.

10 Mark B. Gerstein, Zhi John Lu, et al. Integrative analysis of the *Caenorhabditis elegans* genome by the modENCODE project. *Science*, 2010.

11 Belinda Giardine, Cathy Riemer, Ross C. Hardison, Richard Burhans, Laura Elnitski, Prachi Shah, Yi Zhang, Daniel Blankenberg, Istvan Albert, James Taylor, Webb C. Miller, W. James Kent, and Anton Nekrutenko. Galaxy: a platform for interactive large-scale genome analysis. *Genome research*, 15(10):1451–1455, 2005.

12 Jeremy Goecks, Anton Nekrutenko, James Taylor, and The Galaxy Team. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol*, 11(8):R86, 2010.

13 Desmond Higgins and Paul Sharp. CLUSTAL: a package for performing multiple sequence alignment on a microcomputer. *Gene*, 73(1):237–244, 1988.

14 Michael M. Hoffman, Orion J. Buske, Jie Wang, Zhiping Weng, Jeff A. Bilmes, and William Stafford Noble. Unsupervised pattern discovery in human chromatin structure through genomic segmentation. *Nature Methods*, 9:473–476, 2012.

15 Michael M. Hoffman, Jason Ernst, Steven P. Wilder, Anshul Kundaje, Robert S. Harris, Max Libbrecht, Belinda Giardine, Paul M. Ellenbogen, Jeffrey A. Bilmes, Ewan Birney, Ross C. Hardison, Ian Dunham, Manolis Kellis, and William Stafford Noble. Integrative annotation of chromatin elements from ENCODE data. *Nucleic Acids Research*, 41(2):827–841, 2013.

16 Masato Ishikawa, Tomoyuki Toya, Masaki Hoshida, Katsumi Nitta, Atushi Ogiwara, and Minoru Kanehisa. Multiple sequence alignment by parallel simulated annealing. *Comput. Appl. Biosci.*, 9(3):267–273, 1993.

17 Roy Jonker and Ton Volgenant. Improving the Hungarian assignment algorithm. *Operations Research Letters*, 5(4):171–175, 1986.

18 W. Just. Computational complexity of multiple sequence alignment with SP-score. *Journal of Computational Biology*, 8(6):615–623, 2001.

19 Philip Reiner Kensche, Wieteke Anna Maria Hoeijmakers, Christa Geeke Toenhake, Maaike Bras, Lia Chappell, Matthew Berriman, and Richárd Bártfai. The nucleosome landscape of plasmodium falciparum reveals chromatin architecture and dynamics of regulatory sequences. *Nucleic Acids Research*, 44(5):2110–2124, 2016.

20 W. James Kent, Robert Baertsch, Angie Hinrichs, Webb Miller, and David Haussler. Evolution's cauldron: duplication, deletion, and rearrangement in the mouse and human genomes. *Proc. Natl. Acad. Sci. U. S. A.*, 100(20):11484–11489, 30 September 2003.

21 Jin Kim, Sakti Pramanik, and Moon Chung. Multiple sequence alignment using simulated annealing. *Comput. Appl. Biosci.*, 10(4):419–426, 1994.

22 R. D. Kornberg and L. Stryer. Statistical distributions of nucleosomes: nonrandom locations by a stochastic mechanism. *Nucleic Acids Research*, 16(14A):6677–6690, 07 1988.

23 Ekaterina Kotelnikova, Vsevolod Makeev, and Mikhail Gelfand. Evolution of transcription factor DNA binding sites. *Gene*, 347(2):255–263, 2005.

24 Ben Langmead and Steven L. Salzberg. Fast gapped-read alignment with bowtie 2. *Nat Meth*, 9(4):357–359, 04 2012.

25 M. A. Larkin, G. Blackshields, N. P. Brown, R. Chenna, P. A. McGettigan, H. McWilliam, F. Valentin, I. M. Wallace, A. Wilm, R. Lopez, J. D. Thompson, T. J. Gibson, and D. G. Higgins. Clustal W and Clustal X version 2.0. *Bioinformatics*, 23(21):2947–2948, 2007.

26 Elisa Leimgruber, Queralt Seguin-Estevez, Isabelle Dunand-Sauthier, Natalia Rybtsova, Christoph D. Schmid, Giovanna Ambrosini, Philipp Bucher, and Walter Reith. Nucleosome eviction from MHC class II promoters controls positioning of the transcription start site. *Nucleic Acids Res*, 37(8):2514–2528, May 2009.

27 Hongde Liu, Xueye Duan, Shuangxin Yu, and Xiao Sun. Analysis of nucleosome positioning determined by DNA helix curvature in the human genome. *BMC Genomics*, 12:72, Jan 2011.

28 Saulius Lukauskas, Roberto Visintainer, Guido Sanguinetti, and Gabriele B. Schweikert. DGW: an exploratory data analysis tool for clustering and visualisation of epigenomic marks. *BMC Bioinformatics*, 17(16):447, 2016.

29 Andrew Makhorin. GLPK (GNU linear programming kit), 2008.

30 Alessandro Mammana and Ho-Ryun Chung. Chromatin segmentation based on a probabilistic model for read counts explains a large portion of the epigenome. *Genome Biology*, 16(1):151, 2015.

31 Alan Moses, Derek Chiang, Daniel Pollard, Venky Iyer, and Michael Eisen. MONKEY: identifying conserved transcription-factor binding sites in multiple alignments using a binding site-specific evolutionary model. *Genome biology*, 5(12):R98, 2004.

32 S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3):443–453, Mar 1970.

33 C. Notredame and D. G. Higgins. SAGA: Sequence alignment by genetic algorithm. *Nucleic Acids Res.*, 24(8):1515–1524, 1996.

34 C. Notredame, E. A. O'Brien, and D. G. Higgins. RAGA: RNA sequence alignment by genetic algorithm. *Nucleic acids research*, 25(22):4570–4580, 1997.

35 Heather E. Peckham, Robert E. Thurman, Yutao Fu, John A. Stamatoyannopoulos, William Stafford Noble, Kevin Struhl, and Zhiping Weng. Nucleosome positioning signals in genomic DNA. *Genome Res*, 17(8):1170–1177, Aug 2007.

36 A. Polishko, E. M. Bunnik, K. G. Le Roch, and S. Lonardi. PuFFIN: A parameter-free method to build nucleosome maps from paired-end reads. *BMC Bioinformatics*, 15(Suppl 9):S11, 2014.

37 Anton Polishko, Nadia Ponts, Karine G Le Roch, and Stefano Lonardi. NORMAL: accurate nucleosome positioning using a modified gaussian mixture model. *Bioinformatics (Oxford, England)*, 28(12):i242–9, June 2012.

38 Ekaterina Protozanova, Peter Yakovchuk, and Maxim D. Frank-Kamenetskii. Stacked-unstacked equilibrium at the nick site of DNA. *J Mol Biol*, 342(3):775–785, Sep 2004.

39 Rainer Pudimat, Ernst-Günter Schukat-Talamazzini, and Rolf Backofen. A multiple-feature framework for modelling and predicting transcription factor binding sites. *Bioinformatics*, 21(14):3082–3088, 2005.

40 Aaron R. Quinlan and Ira M. Hall. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26(6):841–842, 2010.

41 S. Roy, J. Ernst, P. V. Kharchenko, and P. Kheradpour. Identification of functional elements and regulatory circuits by Drosophila modENCODE. *Science*, 2010.

42 Rafik A. Salama and Dov J. Stekel. A non-independent energy-based multiple sequence alignment improves prediction of transcription factor binding sites. *Bioinformatics*, 29(21):2699–2704, 2013.

43 Eran Segal, Yvonne Fondufe-Mittendorf, Lingyi Chen, AnnChristine Thastrom, Yair Field, Irene K. Moore, Ji-Ping Z. Wang, and Jonathan Widom. A genomic code for nucleosome positioning. *Nature*, 442(7104):772–778, 08 2006.

44 T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J Mol Biol*, 147(1):195–197, Mar 1981.

45 Julie Thompson, Toby Gibson, and Des Higgins. Multiple sequence alignment using ClustalW and ClustalX. *Current protocols in bioinformatics*, Chapter 2, 2002.

46 Julie Thompson, Desmond Higgins, and Toby Gibson. CLUSTALW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673–4680, 1994.

47 Mari-Liis Visnapuu and Eric C Greene. Single-molecule imaging of DNA curtains reveals intrinsic energy landscapes for nucleosome deposition. *Nat Struct Mol Biol*, 16(10):1056–1062, 10 2009.

48 Jia Wang, Shuai Liu, and Weina Fu. Nucleosome positioning with set of key positions and nucleosome affinity. *Open Biomed Eng J*, 8:166–170, 2014.

49 L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4):337–348, 1994.

50 Yong Zhang, Tao Liu, Clifford A. Meyer, Jérôme Eeckhoute, David S. Johnson, Bradley E. Bernstein, Chad Nusbaum, Richard M. Myers, Myles Brown, Wei Li, and X. Shirley Liu. Model-based analysis of ChIP-Seq (MACS). *Genome Biology*, 9(9):R137, 2008.

51 Yong Zhang, Zarmik Moqtaderi, Barbara P. Rattner, Ghia Euskirchen, Michael Snyder, James T. Kadonaga, X. Shirley Liu, and Kevin Struhl. Intrinsic histone-DNA interactions are not the major determinant of nucleosome positions in vivo. *Nat Struct Mol Biol*, 16(8):847–852, Aug 2009.

52 Xiujuan Zhao, Zhiyong Pei, Jia Liu, Sheng Qin, and Lu Cai. Prediction of nucleosome DNA formation potential and nucleosome positioning using increment of diversity combined with quadratic discriminant analysis. *Chromosome Res*, 18(7):777–785, Nov 2010.