博士論文

Efficient algorithms for processing genome-wide chromatin information

(ゲノム全域のクロマチン情報を処理する効率的アルゴリズム)

市川　和樹

Contents

Abstract

Development of sequencing technology such as Chip-seq, MNase-seq, and Single-Molecule Real-Time sequencing has been accelerating genome-wide chromatin information collections. To gain insight into biological systems from large-scale chromatin data, there is a pressing need to have efficient analytical methods. To overcome this problem, I devised two novel algorithms for processing chromatin information and verified the efficiency and effectiveness of my methods using real biological datasets.

One algorithm named "BoostKCP" boosts the calculation of *k*-means clustering in terms of the Pearson correlation distance, which is widely used for processing large-scale datasets in life science. BoostKCP avoids unnecessary computation in *k*-means clustering by utilizing some heuristic properties specific to the Pearson correlation distance, thereby reducing the overall computational time. To demonstrate the usefulness of the heuristics, I compared its computational time with those of the classic Lloyd's algorithm and other two relevant accelerating methods, the Elkan's and Hamerly's algorithms, using nucleosome positioning data and two other biological data. BoostKCP outperformed other methods in various conditions.

For detecting regions with a specific combination of epigenetic modifications, I invented "CSMinfinder" that is capable of handling large epigenetic information in time linear to the size of a given genome. Precisely, CSMinfinder calculates the similarity score between a focal combination and raw epigenetic states at each DNA position, and outputs an optimal set of large non-overlapping regions (longer than a threshold) that maximizes the sum of similarity scores. With this method, I detected large hypomethylated regions with H3K27me3 marks which overlapped with many

developmental genes in the human and medaka genomes.

In my efforts to achieve more precise analysis on chromatin information, I found it essential to use more accurate genomic sequences with a smaller number of gaps. For this purpose, I used chromatin conformation capture data collected by the Hi-C method, and constructed new medaka genomes for three inbred strains so that each genome has only hundreds of gaps and contains pericentromeric regions.

General Introduction

Chromatin is a conformation in eukaryotic cell composed of DNA, proteins and RNA [1]. The basic unit of chromatin is a structure consisting of a segment of DNA wound around histone octamer composed of two copies of each of the four core histone proteins. In each nucleosome, approximately 146bp base pairs of DNA wrap around histone core particle 1.65 times in a left-handed super-helical turn [2], [3]. Fundamental structure is an iteration of nucleosome core particles and bare DNA sequence between nucleosomes called linker DNA [4], [5]. As higher order structures of chromatin DNA form heterochromatin or heterochromatin classified by the level of condensation [6]. In heterochromatin, a highly condensed chromatin forms, gene expressions are suppressed, conversely chromatin condensation are loosened, and gene transcriptions are activated.

Gene regulation by chromatin has been widely studied in epigenetics. In chromatin structure, nucleosome positioning patterns are known to be one of the important factors regulating gene transcription [7], [8]. Especially, nucleosome positioning around transcription starting sites have an important role in gene expression. Nucleosomes downstream of promoters are called +1 nucleosomes and are known to be stably positioned [9]. In the *S. cerevisiae* genome, it has been reported the presence of nucleosome-depleted regions (NDRs) in upstream of promoters [10]. As the factors of regulating nucleosome positioning, DNA sequence [11], DNA methylation [12] and histone modifications [13] are thought to be important; however, the mechanism of nucleosome positioning have not fully understood.

In addition to nucleosome positioning, histone modifications influence activation and

repression of gene expression [14], [15]. Histone tails which are N-terminals of histone proteins are likely to have chemical modifications including methylation, acetylation, phosphorylation and ubiquitination. Histone modifications change the chromatin construction and affect gene transcription in a variety of ways [16]. Histone acetylation is correlated with transcriptional activation [17], [18]. Acetylation reduces positive charge of histone and loosen the binding between nucleosome and DNA, thereby promoting gene expression. On the other hand, lysine methylation causes both transcriptional activation and suppression determined by position of residue in histone tail [19], [20]. Histone H3 lysine 4 (H3K4) methylation is enriched specifically in hypomethylated gene promoter regions and causes transcriptional activation [21], [22]. Conversely, in embryonic stem cells, hypomethylated regions around promoters of developmentally regulated genes are frequently marked with H3K27me3 which repress gene transcription [23], [24]. These regions are often longer than several kilo base pairs, and genes are stayed in “poised” state, which is not simply suppressed. Poised states are thought to be essential for embryonic cells to maintain pluripotency, and previous research in medaka genome suggested that shortening development hypomethylated domains with H3K27me3(K27HMD) weakens repression so that developmental genes are activated [25]. Large K27HMD around developmental gene promoter are often conserved between medaka and human, suggesting a combination of hypomethylation and H3K27me3 in vertebrate could be a common mechanism behind gene regulation.

By the recent advances on the development of sequencing technology such as Chip-seq [26], MNase-seq [27], and single molecule real time [28]–[30] sequencer, genome-wide chromatin information can be captured at a feasible cost. To analyze such massive data and gain new insight, efficient methods for processing are needed.

In my thesis I devised two novel methods focused on accelerating *k*-means clustering using Pearson correlation distance, and detecting regions which modified by specific epigenomic combinations.

Clustering is an unsupervised learning method to classify data into groups based on similarity among data. Typical clustering methods are hierarchical clustering, *k*-means clustering, self-organizing maps, and principal components analysis [31]. To process biological data, such as gene expression data [32]–[35], histone modifications [36]–[44] and nucleosome positioning [12], [45]–[54], various clustering methods are utilized to discover biological findings in clustered groups. Extensive research studies have been done to classify nucleosome positioning around transcription starting sites using clustering. In yeast, for example, nucleosome positioning in the regions within 800bp of TSSs were clustered into 4 patterns by *k*-means clustering, and characteristic of clusters were annotated according to Gene Ontology [46]. A striking example different from the previous research [55] was that two of four cluster were found to lack clear NDR. In the human genome, nucleosome patterns were classified into 17 clusters and roughly divided into categories which have strongly positioned nucleosomes in upstream of the TSS and in downstream of the TSS [54]. It was revealed by the precise clustering general model of nucleosome-depleted regions in upstream of promoters is not necessarily hold.

Detecting chromatin states with distinct combinations of chromatin modification patterns have been also widely researched. ChromHMM [42] is a statistical method for classifying epigenetic modifications and dividing a DNA sequence into sub-regions of similar chromatin states using Hidden Markov model. In the human genome,

ChromHMM listed chromatin states by combinations of histone modifications, and identified region specific chromatin states.

In my doctoral thesis, I devised two novel algorithms for processing genome-wide chromatin information to solve above mentioned problems. In Chapter 1, to classify large high dimensional biological data such as nucleosome positioning signal data, I invented “BoostKCP”, an accelerating method for *k*-means clustering using the Pearson correlation distance. I applied BoostKCP to human nucleosome positioning signal data and two other biological data, and compared the computational time with classic Lloyd’s algorithm, the first, simple *k*-means clustering, and other two accelerating methods, Elkan’s and Hamerly’s algorithms. In a variety of conditions, my method outperformed Lloyd’s, Elkan’s and Hamerly’s algorithm.

In Chapter 2, I proposed a linear time algorithm for detecting regions which modified by specific epigenomic combinations called “CSMinfinder”. My algorithm calculates the similarity score between a focal combination of epigenetic modifications and raw epigenetic states at each DNA position, and is able to detect a set of non-overlapping regions which maximizes the sum of similarity scores under the constraint that the length of each region is greater than or equal to a given minimum threshold. Using CSMinfinder, I detected large hypomethylated and modified by H3K27me3 regions (K27HMD) which contained many developmental genes in the medaka and human genome.

In Chapter 3, I constructed medaka draft genome using SMRT sequencing reads intended to make less gap genome to elucidate chromatin conformation. I used Hi-C

data to anchor contigs to chromosome which contained centromeric repeats that could not be assembled in past medaka genome.

Chapter 1

A Simple but Powerful Heuristic Method for Accelerating *k*-Means Clustering of Large-Scale Data in Life Science

Introduction

This chapter is a modified version of my paper "A Simple but Powerful Heuristic Method for Accelerating *k*-Means Clustering of Large-Scale Data in Life Science" [56].

Nucleosome is a fundamental unit of chromatin structure in eukaryotes, which is composed of a segment of DNA wound around eight histone proteins core [1]. Each nucleosome core is consisting of 2 copies each of core histones H2A, H2B, H3 and H4 and about 146bp of DNA are wrapped around histone octamer [2], [3]. Chromatin structure is composed of nucleosomes and free DNA between nucleosomes called "linker DNA". Generally, existence of nucleosome prevents binding of transcription factor and nucleosome positioning patterns are thought to be associated with gene regulations [7], [8]. Especially it is known that around transcription starting sites there are nucleosome-free regions on the upstream of promoters, and +1 nucleosome downstream of promoters are stably positioned in whole genome [10]. Recent studies have revealed the association between nucleosome and gene regulation by clustering nucleosome patterns. As example, in yeast, nucleosome patterns are classified into four clusters and tendencies of genes in each clusters are researched [46]. In the human genome, nucleosome positioning around transcription starting sites were classified into 17 clusters and asymmetric pattern which have strongly positioned nucleosomes in upstream of TSS [54]. Mechanism of regulating nucleosome positioning has not been fully understood and clustering nucleosome positioning is thought to be a useful method for getting new insight of chromatin structures.

A variety of clustering algorithms, such as hierarchical clustering, *k*-means clustering, self-organizing map (SOM), and principal components analysis (PCA), have been used

for gain insights into biological systems (for review, see [31]).

Of these, *k*-means clustering is the most widely used to process large-scale data sets, in part because the computational complexity of hierarchical clustering is quadratic or higher in the number of data points, while *k*-means clustering algorithms have lower computational complexity [57]. Accelerating *k*-means clustering algorithms is still necessary to process the growing volume of biological data due to the recent progress in data collection by next-generation sequencing.

The basic concept of *k*-means clustering is simple.

1. It first selects *k* cluster centroids in some manner. The behavior of the algorithm is highly sensitive to the initial selection of *k* initial centroids, and many efficient initialization methods have been proposed to calculate better *k* centroids [57]–[64]. In this study, I use the initialization method proposed by Bradley and Fayyad [62], since it consistently performs better than the other methods in terms of several criteria according to the recent report by Celebi et al [57].
2. Subsequently, *k*-means clustering repeats the process of assigning individual points to their nearest centroids and updating each of *k* centroids as the mean of points assigned to the centroid until no further changes occur on the *k* centroids [65].

Quantifying the same data points is essential. Various measures are available, such as Euclidean distance, Manhattan distance, Pearson correlation distance, and Spellman rank correlation. Of these, Euclidean distance and Pearson correlation distance have been widely used for large-scale biological data processing [34], [35], [54], [66], [67]. Euclidean distance is sensitive to scaling, while correlation is unaffected by scaling. Precisely, given two data of high dimension such that their patterns are quite similar but their scales are different, Euclidean distance is not suitable for measuring the similarity. To avoid this problem, standardized Euclidean distance, which is not sensitive to scaling,

is frequently used [34], [67]–[71].

Of note, standardized Euclidean and Pearson correlation distances are equivalent in the sense that both yield the same *k*-means clustering result for identical sets of *k* initial centroids because the standardized Euclidean distance is proportional to the square root of the Pearson correlation distance [34], [71], and the two distances always produce consistent orderings. Thus, optimization methods designed to calculate one distance are applicable to the other.

Despite the importance of the Pearson correlation and standardized Euclidean distances for machine learning, optimization methods customized for these distances are largely unexplored. In general, several efficient *k*-means clustering algorithms have been proposed for processing Euclidean distances by utilizing the triangle inequality [72]–[74] or by analyzing the correlation coefficient between the centroids [75]. Thus, I can use optimization methods for the Euclidean distance to yield a *k*-means clustering result based on the standardized Euclidean distance that is in agreement with that based on the Pearson correlation distance [34].

I instead examined the properties of the Pearson correlation distance and devised a simple and novel method for avoiding unnecessary computation in order to boost *k*-means clustering using the Pearson correlation distance. I demonstrate that my method outperforms pruning method applications using the Euclidean distance [72]–[74] compared with those that use the standardized Euclidean distance. My method has been best optimized for *k*-means clustering using the standardized Euclidean and Pearson correlation distances.

Methods

I first introduce the definition of Pearson's correlation coefficient.

Definition. *To measure the distance between two d dimensional vectors* $\boldsymbol{x} = (\boldsymbol{x}[1], \dots, \boldsymbol{x}[d])$, $\boldsymbol{y} = (\boldsymbol{y}[1], \dots, \boldsymbol{y}[d])$, *I define Pearson's correlation coefficient:*

$$\rho(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{d}\sum_{i=1}^{d}\left(\frac{\boldsymbol{x}[i] - \bar{\boldsymbol{x}}}{\sigma_{\boldsymbol{x}}}\right)\left(\frac{\boldsymbol{y}[i] - \bar{\boldsymbol{y}}}{\sigma_{\boldsymbol{y}}}\right),$$

where $\bar{\boldsymbol{x}}$ *denotes the average of* $\boldsymbol{x}[1], \dots, \boldsymbol{x}[d]$, *and* $\sigma_{\boldsymbol{x}}$ *is the standard deviation, defined as* $\sqrt{\sum_{i=1}^{d}(\boldsymbol{x}[i] - \bar{\boldsymbol{x}})^2/d}$. *Let* $\|\boldsymbol{x}\|$ *denote its length, defined as* $\sqrt{\sum_{i=1}^{d}\boldsymbol{x}[i]^2}$.

Note that Pearson's correlation coefficient ranges from -1 to 1, i.e., $-1 \le \rho(\boldsymbol{x}, \boldsymbol{y}) \le 1$. The Pearson's correlation coefficient $\rho(\boldsymbol{x}, \boldsymbol{y})$ itself does not serve as a distance because when $\boldsymbol{x}$ and $\boldsymbol{y}$ are more similar to each other, $\rho(\boldsymbol{x}, \boldsymbol{y})$ becomes larger and approaches 1 rather than 0.

Definition.*[76] The Pearson correlation distance* $dis(\boldsymbol{x}, \boldsymbol{y})$ *is defined as* $1 - \rho(\boldsymbol{x}, \boldsymbol{y})$.

The Pearson correlation distance approaches 0 when $\boldsymbol{x}$ and $\boldsymbol{y}$ are similar. In contrast, when $\boldsymbol{x}$ and $\boldsymbol{y}$ are more dissimilar, the Pearson's correlation coefficient decreases to -1, and the Pearson correlation distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ increases approaching 2. The range of the distance is $0 \le \mathrm{dis}(\boldsymbol{x}, \boldsymbol{y}) \le 2$. The Pearson correlation distance violates the triangular inequality.

Example. When $\boldsymbol{x}_1 = (9, 3, 1), \boldsymbol{x}_2 = (3,1, 9)$, and $\boldsymbol{x}_3 = (1,3, 9)$, Pearson correlation distances are

$$\text{dis}(\boldsymbol{x}_1, \boldsymbol{x}_2) = 1.5, \ \text{dis}(\boldsymbol{x}_2, \boldsymbol{x}_3) = 0.115, \text{ and } \ \text{dis}(\boldsymbol{x}_1, \boldsymbol{x}_3) = 1.846,$$

which do not meet the triangular inequality:

$$\text{dis}(\boldsymbol{x}_1, \boldsymbol{x}_2) + \text{dis}(\boldsymbol{x}_2, \boldsymbol{x}_3) \geq \text{dis}(\boldsymbol{x}_1, \boldsymbol{x}_3)$$

I illustrate here two examples that clarify how the Pearson correlation distance differs from the Euclidean distance.

Example. When $\boldsymbol{x}_1 = (1, 3, 9)$, $\boldsymbol{x}_2 = (0.9, 0.3, 0.1)$, and $\boldsymbol{x}_3 = (0.1, 0.3, 0.9)$, $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$ have similar patterns, but their scales are different, while $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ have dissimilar patterns, yet their Euclidean distance is smaller than the distance between $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$**.** Indeed, we have:

$$\text{dis}(\boldsymbol{x}_1, \boldsymbol{x}_3) = 0 < 1.84615 = \text{dis}(\boldsymbol{x}_2, \boldsymbol{x}_3),$$

while

$$\|\boldsymbol{x}_1 - \boldsymbol{x}_3\| = 8.58545 > 1.13137 = \|\boldsymbol{x}_2 - \boldsymbol{x}_3\|.$$

The next example illustrates the discrepancy between the Pearson correlation distance and the "normalized" Euclidean distance.

Example. When $\boldsymbol{x}_1 = (0.1, 0.3, 10), \boldsymbol{x}_2 = (0.1, 1, 10)$, and $\boldsymbol{x}_3 = (0.1, 0.1, 1)$, Pearson correlation distances meet

$$\text{dis}(\boldsymbol{x}_1, \boldsymbol{x}_3) = 0.00016 < 0.00338 = \text{dis}(\boldsymbol{x}_2, \boldsymbol{x}_3),$$

implying that $\boldsymbol{x}_3$ is more similar to (correlated with) $\boldsymbol{x}_1$ than is $\boldsymbol{x}_2$. In contrast, the normalized Euclidean distance yields the opposite ordering:

$$\left\| \frac{\boldsymbol{x}_1}{\|\boldsymbol{x}_1\|} - \frac{\boldsymbol{x}_3}{\|\boldsymbol{x}_3\|} \right\| = 0.11304 > 0.08920 = \left\| \frac{\boldsymbol{x}_2}{\|\boldsymbol{x}_2\|} - \frac{\boldsymbol{x}_3}{\|\boldsymbol{x}_3\|} \right\|$$

I next define the standardized Euclidean distance.

Definition. *Let* $\text{dis\_}SE(\boldsymbol{x}, \boldsymbol{y})$ *denote*

$$\sqrt{\sum_{i=1}^{d} \left(\frac{\boldsymbol{x}[i] - \overline{\boldsymbol{x}}}{\sigma_{\boldsymbol{x}}} - \frac{\boldsymbol{y}[i] - \overline{\boldsymbol{y}}}{\sigma_{\boldsymbol{y}}} \right)^2}$$

the standardized Euclidean distance between two $\boldsymbol{d}$ *dimensional vectors* $\boldsymbol{x}$ *and* $\boldsymbol{y}$.

The square root of the Pearson correlation is proportional to the standardized Euclidean distance.

Proposition.*[34], [71]*

$$\sqrt{2d}\sqrt{dis(\boldsymbol{x}, \boldsymbol{y})} = \text{dis\_SE}(\boldsymbol{x}, \boldsymbol{y})$$

The Pearson correlation distance and the standardized Euclidean distance produce consistent orderings; namely, for any $\boldsymbol{x_1}, \boldsymbol{y_1}, \boldsymbol{x_2}, \boldsymbol{y_2}$,

$$\text{dis}(\boldsymbol{x_1}, \boldsymbol{y_1}) \leq \text{dis}(\boldsymbol{x_2}, \boldsymbol{y_2})$$

if and only if

$$\text{dis\_SE}(\boldsymbol{x_1}, \boldsymbol{y_1}) \leq \text{dis\_SE}(\boldsymbol{x_2}, \boldsymbol{y_2}).$$

I note here that the Pearson correlation distance and its square root are largely different. For example, $\sqrt{dis(\boldsymbol{x},\boldsymbol{y})} = 0.4$ when $dis(\boldsymbol{x},\boldsymbol{y}) = 0.16$, and $\sqrt{dis(\boldsymbol{x},\boldsymbol{y})} = 1.3$ when $dis(\boldsymbol{x},\boldsymbol{y}) = 1.69$. In general, two proximal (distal, respectively) points of the Pearson correlation distance < 1 (> 1) become more distant (closer) according to the square root of the Pearson correlation distance.

Next, I outline Lloyd's algorithm, which implements *k*-means clustering. Given n points in d dimensional space, a *k*-means algorithm starts with selecting k initial centroids, $\{\boldsymbol{c}_p \mid p = 1, \ldots, k\}$, in some way. It then repeats the following two steps until no further changes occur in any of the k centroids:

- Assigning step: Assign each of n points to its nearest centroid.
- Updating step: Update each $\boldsymbol{c}$ of $\boldsymbol{k}$ centroids as the mean of points assigned to $\boldsymbol{c}$.

Lloyd proposed the basic concept of the above procedure [65].
Suppose that it takes $\Theta(d)$ time to compute the distance between two d-dimensional points. A naïve implementation of the assigning step is to calculate the distance between each point and each centroid, which takes a $\Theta(dkn)$ time in total, while the updating step needs a $\Theta(dn)$ time. Thus, accelerating the assigning step is crucial. Here, I present a way of avoiding unnecessary computation in the assigning step by finding unchanged nearest centroids.

Selecting the distance between points is crucial in *k*-means clustering. The Euclidean and Pearson correlation distances are not always consistent and may produce different

clustering results for an identical set of k initial centroids because during the assigning step, the centroid nearest to each vector can differ according to the distance selected. In contrast, the standardized Euclidean and Pearson correlation distances produce consistent orderings, and consequently the centroid closest to each vector is the same regardless of the distance selected. Using this property, I show that both distances yield the same clustering result.

Proposition. *For an identical set of k initial centroids, the k-means clustering algorithm produces the same clustering result for each of the standardized Euclidean distance as the Pearson correlation distance.*

Proof.

I prove the inductive hypothesis stating that before each round of iteration, the set of k centroids for the standardized Euclidean distance is identical to that for the Pearson correlation distance. The hypothesis holds true before the first iteration simply because the same set of k initial centroids is the input for each distance. Assuming that the hypothesis is true before the ith iteration, after the assigning step, the centroid nearest to each vector is identical for each of the two distances because for any vector $\boldsymbol{x}$ and any centroids $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, $dis(\boldsymbol{x}, \boldsymbol{c}_1) \leq dis(\boldsymbol{x}, \boldsymbol{c}_2)$ if and only if $dis\_SE(\boldsymbol{x}, \boldsymbol{c}_1) \leq dis\_SE(\boldsymbol{x}, \boldsymbol{c}_2)$. Thus, after the updating step, the set of vectors closest to each centroid c is identical for the two distances, implying that the mean of the set, the revised centroid, is also identical. Consequently, the inductive hypothesis is true before the $(i+1)$-th iteration.

This proposition allows us to perform k-means clustering with the Pearson correlation distance by using optimization algorithms developed for the (standardized)

Euclidean distance [72]–[74]; however, it is unclear whether methods for the Euclidean distance are effective for accelerating the performance when using the standardized Euclidean distance. I show relevant experimental results in the next section.

For the following, I describe my new algorithm customized for the Pearson correlation distance. Centroids are updated frequently and are likely to move long distances in early stages of the repetitive steps. In contrast, in later steps, centroids are unlikely to move, and therefore, the assigning step has a tendency to reassign each point to the previous centroid as the nearest one, which should be avoided. Thus, we can accelerate the assigning step if we can test whether the nearest centroid for a point remains unchanged without recalculating the distances between the point and all centroids. Suppose that after the updating step, the centroid $\boldsymbol{c}_p$ nearest to $\boldsymbol{x}$ moves to $\boldsymbol{c}_p{}'$ for $p = 1, \dots, k$, and any other centroid $\boldsymbol{c}_q$ $(q = 1, \dots, k, q \neq p)$ moves to $\boldsymbol{c}_q{}'$. We ask if $\boldsymbol{x}$ is still closest to cluster $\boldsymbol{c}_p{}'$ after the updating step:

$$\mathrm{dis}(\boldsymbol{c}_p{}', \boldsymbol{x}) \leq \mathrm{dis}(\boldsymbol{c}_q{}', \boldsymbol{x}),$$

for $q = 1, \dots, k (q \neq p)$

To check this test efficiently for any point $\boldsymbol{x}$ without recalculating the new distances on both sides of the inequality, we will develop an efficient method to estimate an upper bound of the new distance $\mathrm{dis}(\boldsymbol{c}_p{}', \boldsymbol{x})$ using the existing distance $\mathrm{dis}(\boldsymbol{c}_p, \boldsymbol{x})$:

$$\mathrm{dis}(\boldsymbol{c}_p{}', \boldsymbol{x}) \leq \mathrm{dis}(\boldsymbol{c}_p, \boldsymbol{x}) + \mathrm{an\_upper\_bound},$$

where we will define "an_upper_bound(≥ 0)" shortly.

Similarly, we will derive a lower bound of $\mathrm{dis}(\boldsymbol{c}_q', \boldsymbol{x})$ using the previous distance $\mathrm{dis}(\boldsymbol{c}_q, \boldsymbol{x})$:

$$\mathrm{dis}(\boldsymbol{c}_q, \boldsymbol{x}) + \mathrm{a\_lower\_bound} \leq \mathrm{dis}(\boldsymbol{c}_q', \boldsymbol{x})$$

for $q = 1, \ldots, k (q \neq p)$, where $\mathrm{a\_lower\_bound} \leq 0$.

Using these methods, we can implement a pruning procedure. If

$$\mathrm{dis}(\boldsymbol{c}_p, \boldsymbol{x}) + \mathrm{an\_upper\_bound} \leq$$

$$\mathrm{dis}(\boldsymbol{c}_q, \boldsymbol{x}) + \mathrm{a\_lower\_bound} \text{ for } q = 1, \ldots, k (q \neq p), \ (*)$$

we can confirm $\mathrm{dis}(\boldsymbol{c}_p', \boldsymbol{x}) \leq \mathrm{dis}(\boldsymbol{c}_q', \boldsymbol{x}) \quad (q \neq p)$ without calculating the new distances, while retaining the final solution. In the next round of the assigning step, it might be necessary to calculate the new distances, but we can omit this step by substituting $\mathrm{dis}(\boldsymbol{c}_p, \boldsymbol{x}) + \mathrm{an\_upper\_bound}$ and $\mathrm{dis}(\boldsymbol{c}_q, \boldsymbol{x}) + \mathrm{a\_lower\_bound}$ for new distances $\mathrm{dis}(\boldsymbol{c}_p', \boldsymbol{x})$ and $\mathrm{dis}(\boldsymbol{c}_q', \boldsymbol{x})$ respectively because this replacement does not violate the validity of the pruning procedure in the next assigning step. In cases in which the inequality (*) does not hold, we calculate $\mathrm{dis}(\boldsymbol{c}_p', \boldsymbol{x})$ and $\mathrm{dis}(\boldsymbol{c}_q', \boldsymbol{x})$ for $q = 1, \ldots, k \ (q \neq p)$, and determine the centroid nearest to $\boldsymbol{x}$.

To facilitate the simple description of formula and derivations, I introduce a method of decomposing the Pearson's correlation coefficient $\rho(\boldsymbol{x}, \boldsymbol{y})$ into two vectors called "correlation coefficient vectors."

Definition. Correlation coefficient vectors are defined as

$$\frac{1}{\sqrt{d}}\left(\frac{\boldsymbol{x}[1]-\overline{\boldsymbol{x}}}{\sigma_{\boldsymbol{x}}},\frac{\boldsymbol{x}[2]-\overline{\boldsymbol{x}}}{\sigma_{\boldsymbol{x}}},\cdots\frac{\boldsymbol{x}[d]-\overline{\boldsymbol{x}}}{\sigma_{\boldsymbol{x}}}\right),$$

$$\frac{1}{\sqrt{d}}\left(\frac{\boldsymbol{y}[1]-\overline{\boldsymbol{y}}}{\sigma_{\boldsymbol{y}}},\frac{\boldsymbol{y}[2]-\overline{\boldsymbol{y}}}{\sigma_{\boldsymbol{y}}},\cdots\frac{\boldsymbol{y}[d]-\overline{\boldsymbol{y}}}{\sigma_{\boldsymbol{y}}}\right),$$

for $\boldsymbol{x}=(\boldsymbol{x}[1],\ldots,\boldsymbol{x}[d])$ and $\boldsymbol{y}=(\boldsymbol{y}[1],\ldots,\boldsymbol{y}[d])$, respectively. Let $CC\boldsymbol{x}$ and $CC\boldsymbol{y}$ denote the respective correlation coefficient vectors.

Note that the Pearson's correlation coefficient $\rho(\boldsymbol{x},\boldsymbol{y})$ is equal to the inner product of $CC\boldsymbol{x}$ and $CC\boldsymbol{y}$**;** i.e., $\rho(\boldsymbol{x},\boldsymbol{y})=(CC\boldsymbol{x},CC\boldsymbol{y})$. Any correlation coefficient vector $CC\boldsymbol{x}$ is of length 1; namely, $\|CC\boldsymbol{x}\|=1$, and similarly, $\|CC\boldsymbol{y}\|=1$.

To facilitate the discussion of calculating better upper and lower bounds, I introduce a new definition.

Definition. *Let* $\boldsymbol{c}$ *and* $\boldsymbol{c}'$ *be respective centroids before and after the updating step, and let* $CC\boldsymbol{c}$ *and* $CC\boldsymbol{c}'$ *be their correlation coefficient vectors. Let* $\Delta dis(\boldsymbol{c},\boldsymbol{c}',\boldsymbol{x})$ *denote* $dis(\boldsymbol{c}',\boldsymbol{x})-dis(\boldsymbol{c},\boldsymbol{x})$*, the distance variation of point* $\boldsymbol{x}$ *to* $\boldsymbol{c}$ *and* $\boldsymbol{c}'$.

For example, $\mathrm{dis}(\boldsymbol{c}_p{}',\boldsymbol{x})\le\mathrm{dis}(\boldsymbol{c}_p,\boldsymbol{x})+\mathrm{an\_upper\_bound}$ can be concisely described by

$$\Delta dis(\boldsymbol{c}_p,\boldsymbol{c}_p{}',\boldsymbol{x})\le\mathrm{an\_upper\_bound}.$$

Another merit of this notation is that we are able to transform the distance variation into an inner product of $(CC\boldsymbol{c} - CC\boldsymbol{c}')$ and $CC\boldsymbol{x}$ **:**

$$\begin{aligned}\Delta dis(\boldsymbol{c}_p, \boldsymbol{c}_p', \boldsymbol{x}) &= dis(\boldsymbol{c}_p', \boldsymbol{x}) - dis(\boldsymbol{c}_p, \boldsymbol{x}) \\ &= \rho(\boldsymbol{c}_p, \boldsymbol{x}) - \rho(\boldsymbol{c}_p', \boldsymbol{x}) \\ &= (CC\boldsymbol{c}_p, CC\boldsymbol{x}) - (CC\boldsymbol{c}_p', CC\boldsymbol{x}) \\ &= (CC\boldsymbol{c}_p - CC\boldsymbol{c}_p', CC\boldsymbol{x})\end{aligned}$$

This inner product allows us to estimate an upper bound and a lower bound of $\Delta dis(\boldsymbol{c}_p, \boldsymbol{c}_p', \boldsymbol{x})$ by analyzing the two vectors independently as well as by considering each dimension separately.

We can derive an upper bound and a lower bound that are effective for any point $\boldsymbol{x}$ for which the nearest centroid is $\boldsymbol{c}_p$**.** A simple approach is to derive two bounds from

$$\begin{aligned}\left\|\Delta dis(\boldsymbol{c}_p, \boldsymbol{c}_p', \boldsymbol{x})\right\| &= \left\|(CC\boldsymbol{c}_p - CC\boldsymbol{c}_p', CC\boldsymbol{x})\right\| \\ &\leq \left\|CC\boldsymbol{c}_p - CC\boldsymbol{c}_p'\right\| \|CC\boldsymbol{x}\|,\end{aligned}$$

where the inequality holds because of the Cauchy-Schwarz inequality. Because $\|CC\boldsymbol{x}\| = 1$, we can use $\left\|CC\boldsymbol{c}_p - CC\boldsymbol{c}_p'\right\|$ and $-\left\|CC\boldsymbol{c}_p - CC\boldsymbol{c}_p'\right\|$ as upper and lower bounds, respectively, and I define them as follows:

Definition.

$$\text{upperA}(\boldsymbol{c}_p, \boldsymbol{c}_p') \stackrel{\text{def}}{=} \left\|CC\boldsymbol{c}_p - CC\boldsymbol{c}_p'\right\|$$

$$\text{lowerA}(\boldsymbol{c}_p, \boldsymbol{c}_p') \stackrel{\text{def}}{=} -\left\|CC\boldsymbol{c}_p - CC\boldsymbol{c}_p'\right\|$$

These upper and lower bounds are simple formulas but effective for eliminating unnecessary computation. It takes $\Theta(dk)$ time to calculate the lower and upper bounds for all *k* centroids, and $\Theta(k)$ space to store these bounds. I also design more complicated bounds by taking the sum of the differences at individual coordinates.

Definition. *Let* $S_{\boldsymbol{c}_p}$ *denote the set of all points for which the nearest centroid is* $\boldsymbol{c}_p$

$$\text{upperB}\left(\boldsymbol{c}_p, \boldsymbol{c}_p', S_{\boldsymbol{c}_p}\right) \stackrel{\text{def}}{=} \sum_{j=1}^{d} \text{maximum}\left(CC\boldsymbol{c}_p[j] - CC\boldsymbol{c}_p'[j], S_{\boldsymbol{c}_p}\right),$$

where

$$\text{maximum}\left(z, S_{\boldsymbol{c}_p}\right) \stackrel{\text{def}}{=} \begin{cases} z \times \max\{CC\boldsymbol{x}[j] \ \ | \boldsymbol{x} \in S_{\boldsymbol{c}_p}\} & z \geq 0 \\ z \times \min\{CC\boldsymbol{x}[j] \ \ | \boldsymbol{x} \in S_{\boldsymbol{c}_p}\} & z < 0 \end{cases}$$

For $q = 1, \ldots, k \ \ (q \neq p)$*, define*

$$\text{lowerB}\left(\boldsymbol{c}_q, \boldsymbol{c}_q', S_{\boldsymbol{c}_p}\right) \stackrel{\text{def}}{=} \sum_{j=1}^{d} \text{minimum}\left(CC\boldsymbol{c}_q[j] - C\boldsymbol{c}_q'[j], S_{\boldsymbol{c}_p}\right),$$

where

$$\text{minimum}\left(z, S_{\boldsymbol{c}_p}\right) \stackrel{\text{def}}{=} \begin{cases} z \times \min\{CC\boldsymbol{x}[j] \ \ | \boldsymbol{x} \in S_{\boldsymbol{c}_p}\} & z \geq 0 \\ z \times \max\{CC\boldsymbol{x}[j] \ \ | \boldsymbol{x} \in S_{\boldsymbol{c}_p}\} & z < 0 \end{cases}$$

Proposition. *For any* $\boldsymbol{x} \in S_{\boldsymbol{c}_p}$,

$$\Delta dis\left(\boldsymbol{c}_p, \boldsymbol{c}_p', \boldsymbol{x}\right) \leq \text{upperB}\left(\boldsymbol{c}_p, \boldsymbol{c}_p', S_{\boldsymbol{c}_p}\right) \text{ and}$$

$$\text{lowerB}\left(\boldsymbol{c}_q, \boldsymbol{c}_q', S_{\boldsymbol{c}_p}\right) \ \leq \Delta dis\left(\boldsymbol{c}_q, \boldsymbol{c}_q', \boldsymbol{x}\right) \quad (q \neq p).$$

It takes $\Theta(dn + dk^2)$ time and $\Theta(dk + k^2)$ space in order to calculate $\text{upperB}\left(\boldsymbol{c}_p, \boldsymbol{c}_p', S_{\boldsymbol{c}_p}\right)$ and $\text{lowerB}\left(\boldsymbol{c}_q, \boldsymbol{c}_q', S_{\boldsymbol{c}_p}\right)$ $(p = 1, \ldots, k,\ q = 1, \ldots, k, q \neq p)$ for every cluster $\boldsymbol{c}_p$.

Proof

$$\Delta dis(\boldsymbol{c}_p, \boldsymbol{c}_p', \boldsymbol{x}) = \left((CC\boldsymbol{c}_p - CC\boldsymbol{c}_p'), CC\boldsymbol{x}\right)$$

$$= \sum_{j=1}^{d} \left(CC\boldsymbol{c}_p[j] - CC\boldsymbol{c}_p'[j]\right) \times CC\boldsymbol{x}[j]$$

$$\leq \sum_{j=1}^{d} \text{maximum}\left(CC\boldsymbol{c}_p[j] - CC\boldsymbol{c}_p'[j], S_{\boldsymbol{c}_p}\right)$$

$$= \text{upperB}\left(\boldsymbol{c}_p, \boldsymbol{c}_p', S_{\boldsymbol{c}_p}\right)$$

$$\Delta dis(\boldsymbol{c}_q, \boldsymbol{c}_q', \boldsymbol{x}) = \left((CC\boldsymbol{c}_q - CC\boldsymbol{c}_q'), CC\boldsymbol{x}\right)$$

$$= \sum_{j=1}^{d} \left(CC\boldsymbol{c}_q[j] - CC\boldsymbol{c}_q'[j]\right) \times CC\boldsymbol{x}[j]$$

$$\geq \sum_{j=1}^{d} \text{minimum}\left(CC\boldsymbol{c}_q[j] - CC\boldsymbol{c}_q'[j], S_{\boldsymbol{c}_q}\right)$$

$$= \text{lowerB}\left(\boldsymbol{c}_q, \boldsymbol{c}_q', S_{\boldsymbol{c}_q}\right)$$

For efficiency, I first compute the maximum and minimum of $\{CC\boldsymbol{x}[j] \mid \boldsymbol{x} \in S_{\boldsymbol{c}_p}\}$ for each dimension $j = 1, \ldots, d$ and for each cluster $\boldsymbol{c}_p$ $(p = 1, \ldots, k)$, and store this information in a table of size $\Theta(dk)$. This tabulation process takes $\Theta(dn)$ time. Looking up the table, it is possible to calculate $\text{upperB}\left(\boldsymbol{c}_p, \boldsymbol{c}_p', S_{\boldsymbol{c}_p}\right)$ for any cluster

$\boldsymbol{c}_p$ in $\Theta(d)$ time, and $\text{lowerB}\left(\boldsymbol{c}_q, \boldsymbol{c}_q', S_{\boldsymbol{c}_p}\right)$ for $(k-1)$ clusters $\boldsymbol{c}_q$ $(q = 1, \ldots, k, q \neq p)$ in $\Theta(d(k-1))$ time. Repeating this calculation for each cluster $\boldsymbol{c}_p = \boldsymbol{c}_1, \ldots, \boldsymbol{c}_k$ requires $\Theta(dk^2)$ time and $\Theta(k^2)$ space for storing upper and lower bounds.

Using the above two calculations for upper and lower bounds, we devise the pruning procedure that checks

$$\text{dis}(\boldsymbol{c}_p, \boldsymbol{x}) + \text{upperA}(\boldsymbol{c}_p, \boldsymbol{c}_p') \leq \text{dis}(\boldsymbol{c}_q, \boldsymbol{x}) + \text{lowerA}(\boldsymbol{c}_q, \boldsymbol{c}_q'),$$

or

$$\text{dis}(\boldsymbol{c}_p, \boldsymbol{x}) + \text{upperB}\left(\boldsymbol{c}_p, \boldsymbol{c}_p', S_{\boldsymbol{c}_p}\right) \leq \text{dis}(\boldsymbol{c}_q, \boldsymbol{x}) + \text{lowerB}\left(\boldsymbol{c}_q, \boldsymbol{c}_q', S_{\boldsymbol{c}_p}\right)$$

for each $\boldsymbol{x}$ of n points ($\boldsymbol{x} \in S_{\boldsymbol{c}_p}$ for each $p = 1, \ldots, k$) and for each $q = 1, \ldots, k$ $(q \neq p)$. If $\boldsymbol{x}$ meets one of the inequalities, we can confirm $\text{dis}(\boldsymbol{c}_p', \boldsymbol{x}) \leq \text{dis}(\boldsymbol{c}_q', \boldsymbol{x})$ $(q \neq p)$ by skipping the calculation of the new distances. The total computation time of checking the above inequality is $\Theta(kn)$. Using upperB and lowerB requires additional computational time $\Theta(dn + dk^2)$ and space $\Theta(dk + k^2)$, which is constantly required to calculate the two bounds in each iteration. In contrast, computing upperA and lowerA needs $\Theta(dk)$ time and $\Theta(k)$ space.

For each $\boldsymbol{x}$ that violates the above inequality, new distances $\text{dis}(\boldsymbol{c}_p', \boldsymbol{x})$ and $\text{dis}(\boldsymbol{c}_q', \boldsymbol{x})$ for $q = 1, \ldots, k$ $(q \neq p)$ are computed to find the centroid nearest to $\boldsymbol{x}$. In the best case, no calculation is needed. In the worst case, however, it is necessary to compute new distances $\text{dis}(\boldsymbol{c}_p', \boldsymbol{x})$ for $p = 1, \ldots, k$ and n points, and the worst time

complexity is $O(dkn)$. Recall for comparison that the assigning step of Lloyd's algorithm requires $\Theta(dkn)$ time.

I have defined two heuristic algorithms: one uses upperA and lowerA, and the other upperB and lowerB to prune unnecessary computations when performing *k*-means clustering using the Pearson correlation distance. I call the former BoostKCP (boundA) and the latter BoostKCP (boundB), where BoostKCP stands for Boosting K-means Clustering for Pearson correlation distance.

I compare the performance of Elkan's and Hamerly's methods, BoostKCP(boundA), and BoostKCP(boundB) with respect to time and space complexity. Although individual method accelerates Lloyd's algorithm using lower and upper bounds to prune unnecessary computation, each iteration requires $O(dkn)$ time in the worst case. Thus, I summarize the overhead of computing lower and upper bounds in terms of time and space complexity (Table 1). The entries of "time/iteration" show the asymptotic overhead computation time required to calculate lower and upper bounds in each iteration by individual algorithms. The entries for BoostKCP have been described, while those for Elkan's and Hamerly's algorithms are detailed in [73]. Table 1 shows that the time and space complexity of BoostKCP(boundA) are smaller than those of the other methods. In the experimental results, I will show that BoostKCP(boundA) also outperforms the others in terms of computational performance using real biological data sets, confirming that BoostKCP(boundA) is a simple and powerful heuristic method for accelerating *k*-means clustering when using Pearson correlation and standardized Euclidean distances.

TABLE 1
Comparison of the Asymptotic Overhead Spent by Calculating Lower and Upper Bounds in Addition to Lloyd's Algorithm in Terms of Time and Space Complexity

	time / iteration	memory
BoostKCP(boundA)	$\Theta(dk)$	$\Theta(k)$
BoostKCP(boundB)	$\Theta(dn + dk^2)$	$\Theta(dk + k^2)$
Elkan	$\Theta(dk^2)$	$\Theta(kn + k^2)$
Hamerly	$\Theta(dk^2)$	$\Theta(n)$

Results

Data sets

I generated a synthetic data set of vectors whose elements were randomly selected from 0 to 1 using the Mersenne twister [77], a widely used pseudorandom number generator with an extraordinarily long cycle of $2^{19,937} - 1$. I generated data sets of 50,000 vectors of dimension $\boldsymbol{d}$ = 10, 20, 50, 101, 201, 501, 1,001, and 2,001. This random data set was an extreme example from which meaningful clusters were difficult to extract. I used these sets to compare the effectiveness of BoostKCP (boundA) and BoostKCP (boundB) for pruning unnecessary computation.

In order to compare BoostKCP with other available state-of-the-art pruning methods, I used three different types of high-dimensional real biological data sets rather than random data sets. The first real data set was a set of vectors with human nucleosome positioning signals at genomic positions surrounding transcription start sites (TSSs). A nucleosome positioning signal at a genomic position is a real value and represents the possibility of the presence of nucleosome centers at that position. From the GENCODE database (version 7) [78], I obtained human nucleosome positioning signals using MNase-sequencing and the TSSs of the human reference genome hg19. I repeated the process of merging neighboring TSSs within 1,000 bp into a group, and I selected representative TSSs whose expression levels were maximal in individual groups. From the representative TSSs, I excluded those having any other TSSs within 1,000 bp on the reverse strand to eliminate their effect. Subsequently, from the nucleosome positioning signal data, I generated a base set of 56,772 vectors of dimension 2,001 (~400M bytes) such that their elements were real-valued nucleosome positioning signals within 1,000

bp around representative TSSs and more than half of the elements within 50, 100, 250, and 500 bp of the TSSs were nonzero. To monitor how the algorithms behave for data of different dimension, from the base set, I generated sets of vectors of dimension d = 101, 201, 501, 1001, and 2001 by selecting the elements within 50, 100, 250, 500, and 1000 bp surrounding the TSSs. The last digit "1" of dimension d indicates the TSS position. Because of the construction of the base set, more than half of the elements in each vector is guaranteed to be nonzero. For smaller dimensions d =10, 20, and 50, I selected every $(2000/d)$-th element for d = 10, 20, and 50 from the base set; *e.g.*, elements at -1000, -800, -600, …, +600, and +800 bp for d=10. The second real data set was a typical example of gene expression data, a set of 54,613 genes from 180 glioma samples [79]. The third real data set was a set of 60,000 gray-level images of handwritten letters in the MNIST database [80]. Each image consisted of 28 x 28 pixels, and I set dimension $d = 28^2 = 784$.

Comparison of computational performance

I compared the following five methods:

- Lloyd's algorithm [65].
- BoostKCP (boundA).
- BoostKCP (boundB).
- Elkan's algorithm [72].
- Hamerly's algorithm [73], [74].

I used the first three methods to compute k-means clustering using the Pearson correlation distance. In contrast, since the latter two algorithms were designed to process the Euclidean distance, I used these to calculate k-means clustering using the

standardized Euclidean distance, the results of which are equal to those using Pearson correlation distance as described in the previous section. For any initial centroid set, the above five methods give the same final clustering result.

Selecting the initial set of k centroids largely affects the final result, and for this purpose, I used Bradley and Fayyad's method [62] because it performed better than the other applicable initialization methods for several criteria [57]. After selecting the initial centroids, I measured the elapsed time during the application of each method towards the same initial centroid set derived from different types of data. I excluded the time required to compute the initial set of centroids because it was typically much less than the time used to compute k-means clustering. I monitored the computational performance using an Intel(R) Xeon(R) CPU E5-2680 v3 processor with a clock rate of 2.50GHz and 529 GB of main memory.

I first compared the performances of BoostKCP (boundA) and BoostKCP (boundB) using 50,000 random vectors of dimension d = 10, 20, 50, 101, 201, 501, 1,001, and 2,001. I calculated the average elapsed time by executing 10 trials for d = 10, 20, 50, 101, 201, 501, but five trials for d =1,001 and 2,001, due to the large amount of computation. I observed that BoostKCP (boundA) outperformed BoostKCP (boundB). Specifically, I calculated the performance improvement by BoostKCP(boundA) as the acceleration rate; i.e., the elapsed time for BoostKCP (boundB) divided by that for BoostKCP (boundA). Fig. 1 displays the elapsed time and acceleration rate for each dimension and for k = 10, 20, and 30. In all cases, BoostKCP (boundA) was faster than BoostKCP (boundB) partly because computing lower and upper bounds for BoostKCP(boundA), $\Theta(dk)$, is less expensive than computing those for

BoostKCP(boundB), $\Theta(dn + dk^2)$, where d is the dimension, n is the number of data, and k is the number of clusters (Table 1). I therefore used BoostKCP (boundA) for my comparisons with the other four algorithms using real data sets.

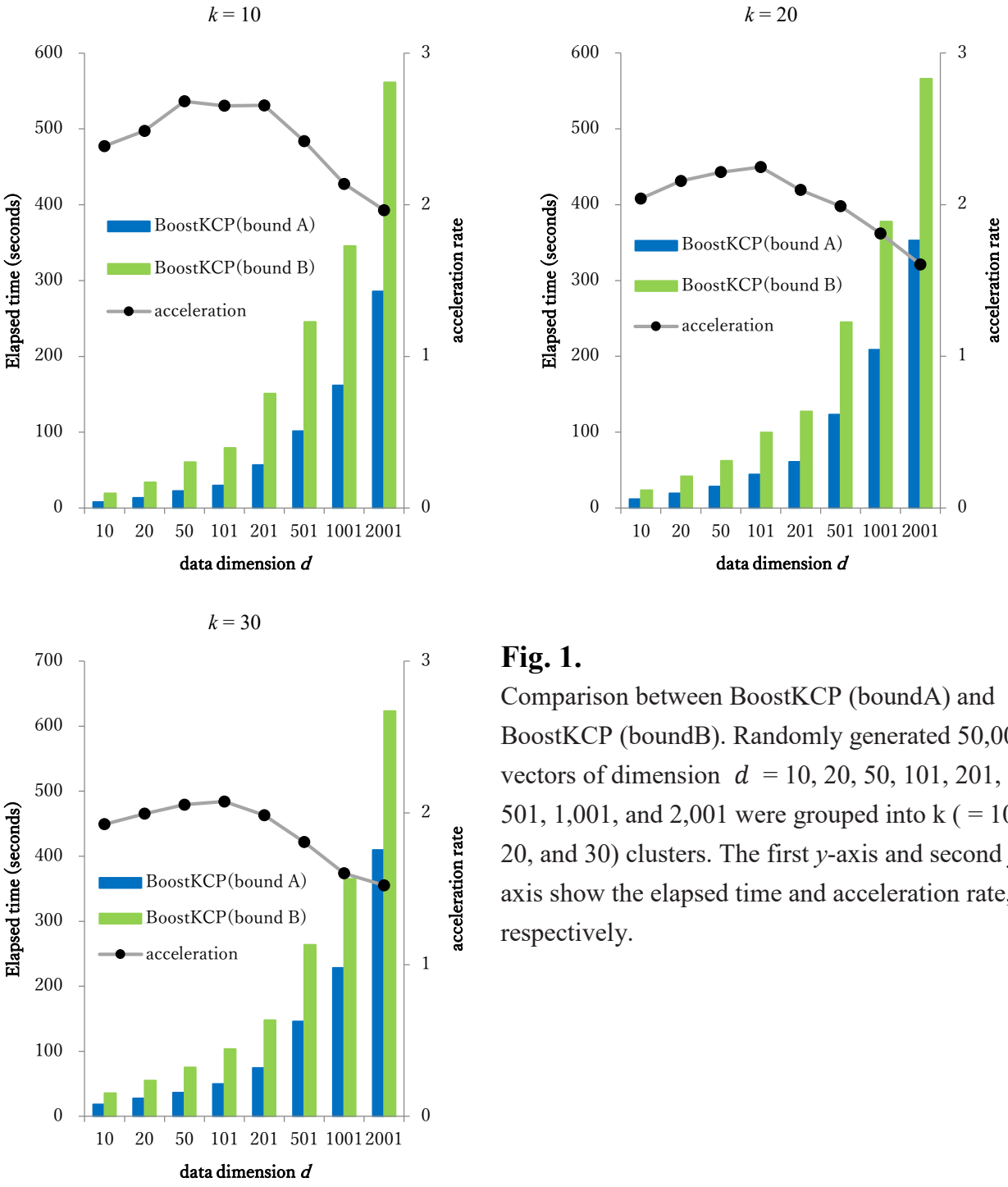


Fig. 1.
Comparison between BoostKCP (boundA) and BoostKCP (boundB). Randomly generated 50,000 vectors of dimension d = 10, 20, 50, 101, 201, 501, 1,001, and 2,001 were grouped into k (= 10, 20, and 30) clusters. The first y-axis and second y-axis show the elapsed time and acceleration rate, respectively.

I next compared BoostKCP (boundA) with Lloyd's, Elkan's, and Hamerly's algorithms using real biological data sets. For measuring the performance improvement by BoostKCP(boundA), I again defined the acceleration rate as the average elapsed time of each algorithm divided by that of BoostKCP (boundA).

Fig. 2 shows the experimental results obtained by applying the four algorithms to the nucleosome positioning data for dimension d =10, 20, 50, 101, 201, 501, 1,001 and 2,001 and for number of clusters k = 10, 20, and 30. I set these values for k because nucleosome positioning signal vectors can be categorized into 10–30 groups with biologically meaningful characteristics [54]. I computed the average elapsed time by performing 10 trials with the exception of five trials where d =1,001 and 2,001. Figs. 2A, 2B, 2C show the BoostKCP (boundA) acceleration rates compared with those of the Lloyd's, Elkan's, and Hamerly's algorithms. BoostKCP (boundA) clearly outperformed Lloyd's and Hamerly's algorithms for all parameter value combinations, and it was also faster than Elkan's algorithm.

It has been reported that Hamerly's algorithm is often faster than Elkan's algorithm for various low-dimensional (d < 50) data using the Euclidean distance [73], [74]; however, Hamerly's algorithm did not work as well for nucleosome positioning data using the standardized Euclidean distance (Figs. 2A, 2B, 2C). I remark here that the standardized Euclidean distance between two points is likely to be much smaller than the Euclidean distance between the two points, implying that the points are densely distributed in standardized Euclidean space. When handling more densely distributed points, greater care has to be taken for pruning unnecessary computation. In each iteration, Elkan's algorithm carefully maintains the lower and upper bounds for the distance between each

point and each centroid, while Hamerly's algorithm considers the closest and second closest centroids only. For pruning unnecessary computation, put another way, Elkan's algorithm requires more time and space to estimate tighter bounds than does Hamerly's algorithm, allowing the former to be more effective in removing unnecessary computation than the latter.

Figs. 2D, 2E, and 2F display the average elapsed time when using each combination of d and k values; however, there is insufficient information as to how these times differed, since the elapsed time in each trial largely depended on the selection of the initial k vectors. To understand this further, I investigated how the elapsed time in each trial changed depending on the number of iterations when I applied BoostKCP (boundA), Elkan's, and Lloyd's algorithms to the nucleosome positioning signal data of dimension $d = 501$ for $k = 10$, 20, and 30. I did not consider Hamerly's algorithm because its performance was similar to that of Lloyd. Fig. 3A shows that how elapsed time of individual algorithm changes for ten different initial sets of centroids. The figure shows that the elapsed time of each algorithm increased in proportion to the number of iterations. A major difference between the three algorithms was that the elapsed time of Elkan's and Lloyd's algorithms increased for larger values of k, but that of my pruning method was almost independent of k, which explains why the acceleration rate increased for larger values of k, as seen in Fig. 2.

To gain a better understanding of this, Fig. 3B presents an in-depth analysis, showing the elapsed time in each iteration of the three algorithms. Each iteration time for Lloyd's algorithm is almost constant because the algorithm does not avoid unnecessary computation, while each iteration time for BoostKCP (boundA) and Elkan's algorithm

for k = 10, 20, and 30 decreased markedly after the first few steps. In later steps, the elapsed time of BoostKCP (boundA) became almost independent of the value of k, giving the account that its overall elapsed time was almost proportional to the number of iterations but independent of k, as shown in Fig. 3A. In contrast, the elapsed time of Elkan's algorithm in each iteration increased for larger values of k. This is because in each iteration, Elkan's algorithm maintains a large array of lower and upper bounds for the distance between each ~56K points and each k centroid at an expense. In contrast, BoostKCP (boundA) needs to calculate only the lower and upper bounds for each k centroid (Table 1).

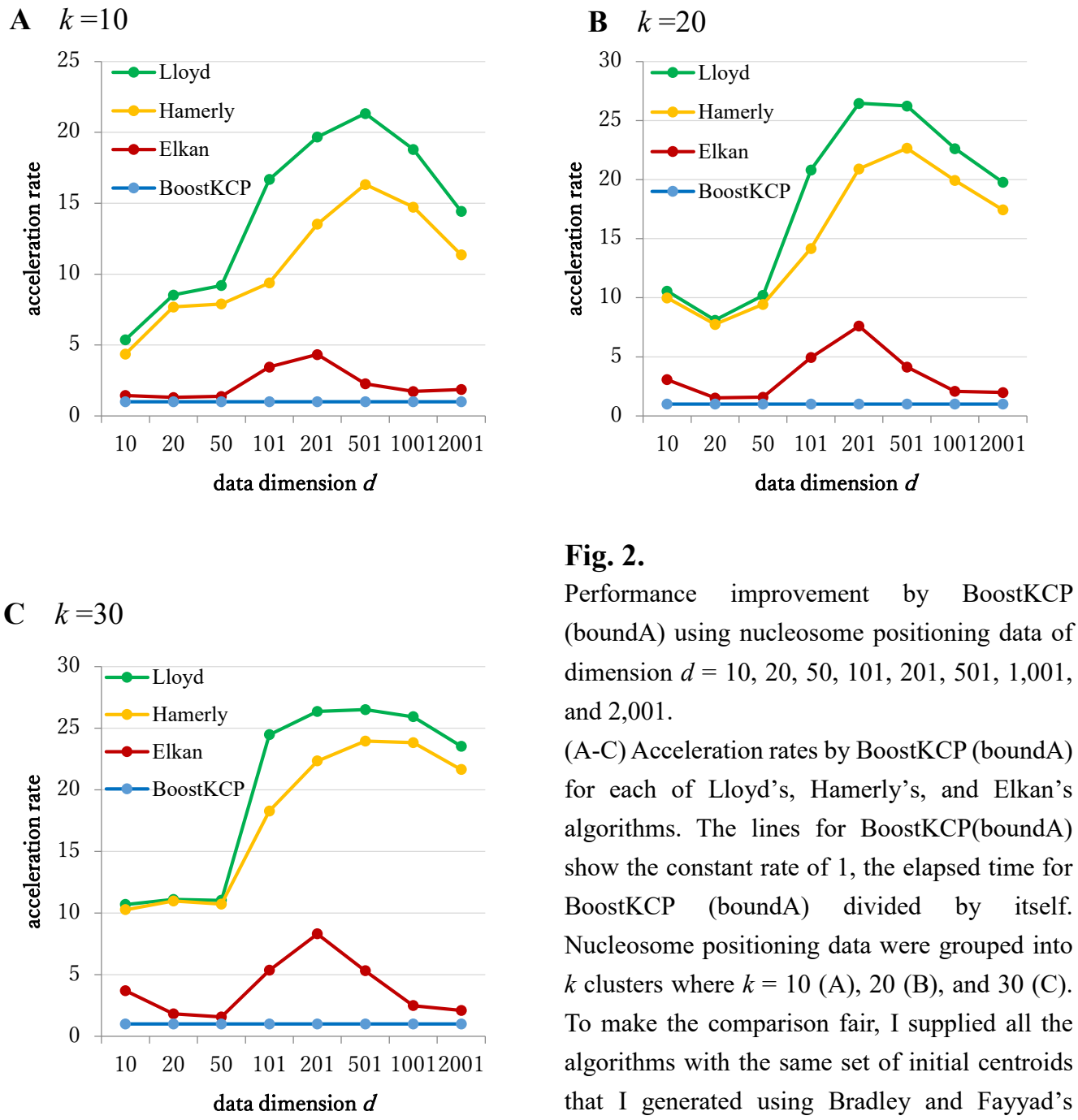


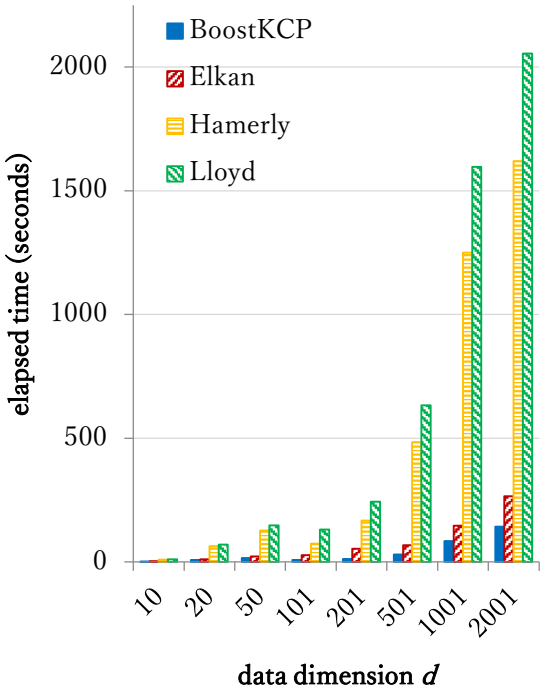
Fig. 2.

Performance improvement by BoostKCP (boundA) using nucleosome positioning data of dimension d = 10, 20, 50, 101, 201, 501, 1,001, and 2,001.

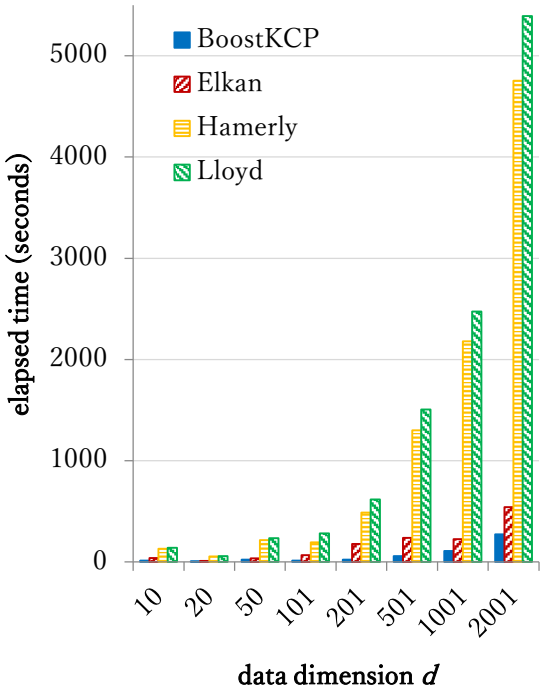
(A-C) Acceleration rates by BoostKCP (boundA) for each of Lloyd's, Hamerly's, and Elkan's algorithms. The lines for BoostKCP(boundA) show the constant rate of 1, the elapsed time for BoostKCP (boundA) divided by itself. Nucleosome positioning data were grouped into k clusters where k = 10 (A), 20 (B), and 30 (C). To make the comparison fair, I supplied all the algorithms with the same set of initial centroids that I generated using Bradley and Fayyad's method.

(D-F) The average elapsed time of BoostKCP (boundA), Lloyd's, Hamerly's, and Elkan's algorithms.

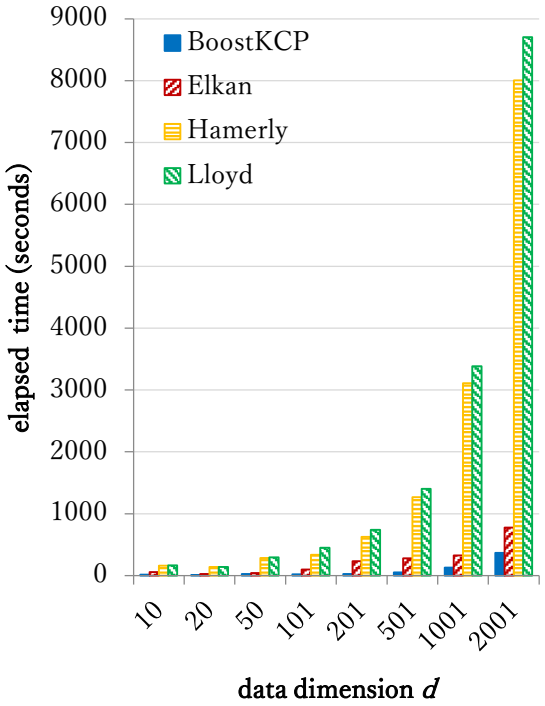
D k =10



E k =20



F k =30



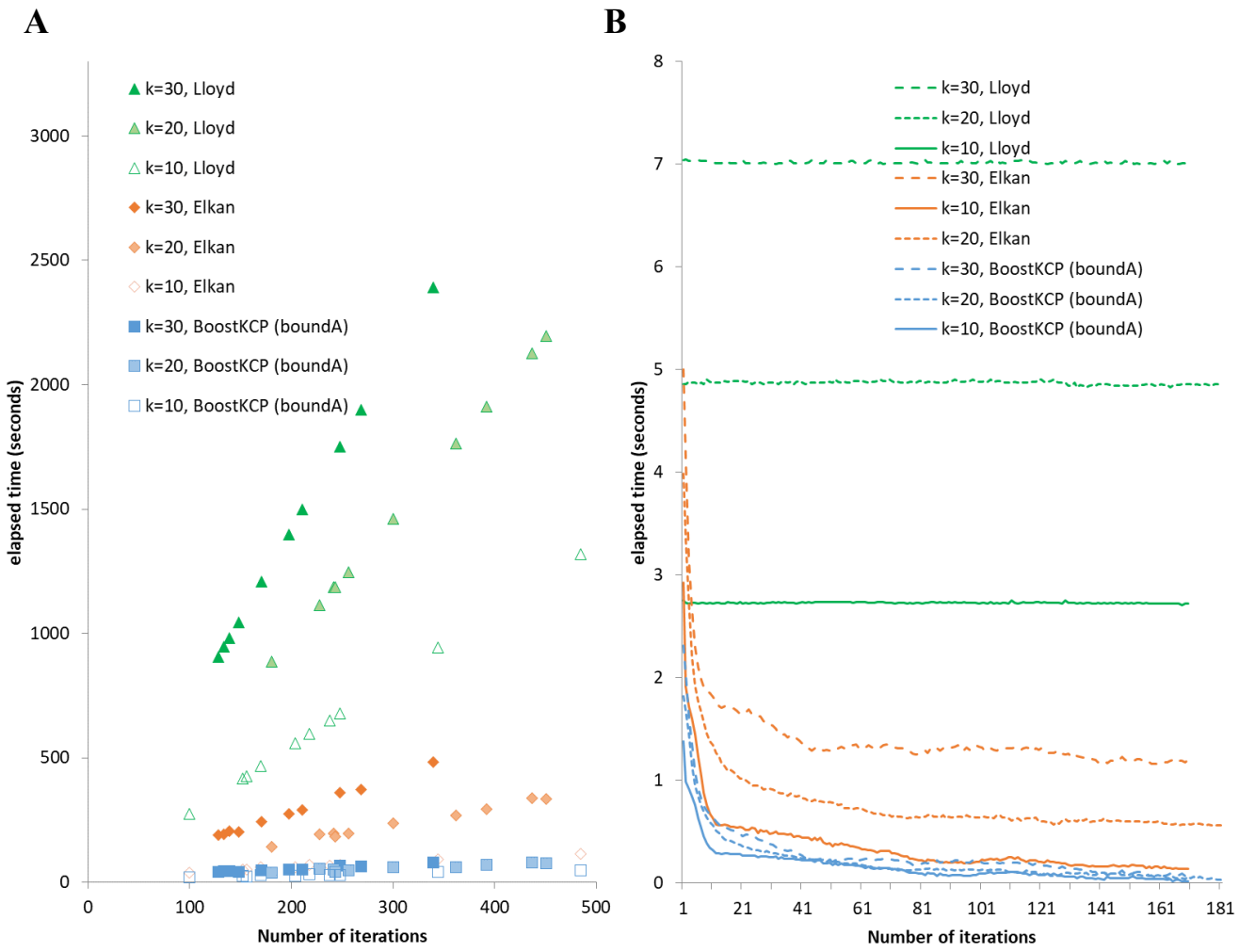


Fig. 3.

In-depth performance analysis on *k*-means clustering of nucleosome positioning data.
(A) Analysis of clustering nucleosome positioning data of dimension $d = 501$ by BoostKCP (boundA), Elkan's and Lloyd's algorithms. Hamerly's algorithm was not considered because Lloyd's and Hamerly's algorithms performed similarly. A dot represents the number of iterations (*x*-axis) and the elapsed time (seconds) of each experiment of 10 trials for $k = 10$, 20 and 30.
(B) Elapsed time of each iteration (including the assigning and updating steps) in typical trials.

I then applied the three algorithms to the gene expression data, a set of 54613 vectors of dimension d = 180. Because the dimension was fixed, I grouped the data into k (= 2, 3, 10, 20, 30, 40, 50, 60, 70) clusters of genes to determine if BoostKCP (boundA) achieved better performance with larger values of k. Fig. 4 shows the average elapsed time for ten trials and the acceleration rate of BoostKCP (boundA). The three algorithms used Bradley and Fayyad's method to generate the same set of initial centroids. BoostKCP (boundA) outperformed Elkan's and Lloyd's algorithms for each k except for the case that the acceleration rate by BoostKCP (boundA) for Elkan's algorithm was 1.54 when k = 2. The acceleration rates were 1.66, 1.79, and 2.22 when k = 3, 10, and 20, respectively. The acceleration rate increased for larger values of k, which was consistent with the performance improvement that I observed for the nucleosome positioning data in Fig. 2.

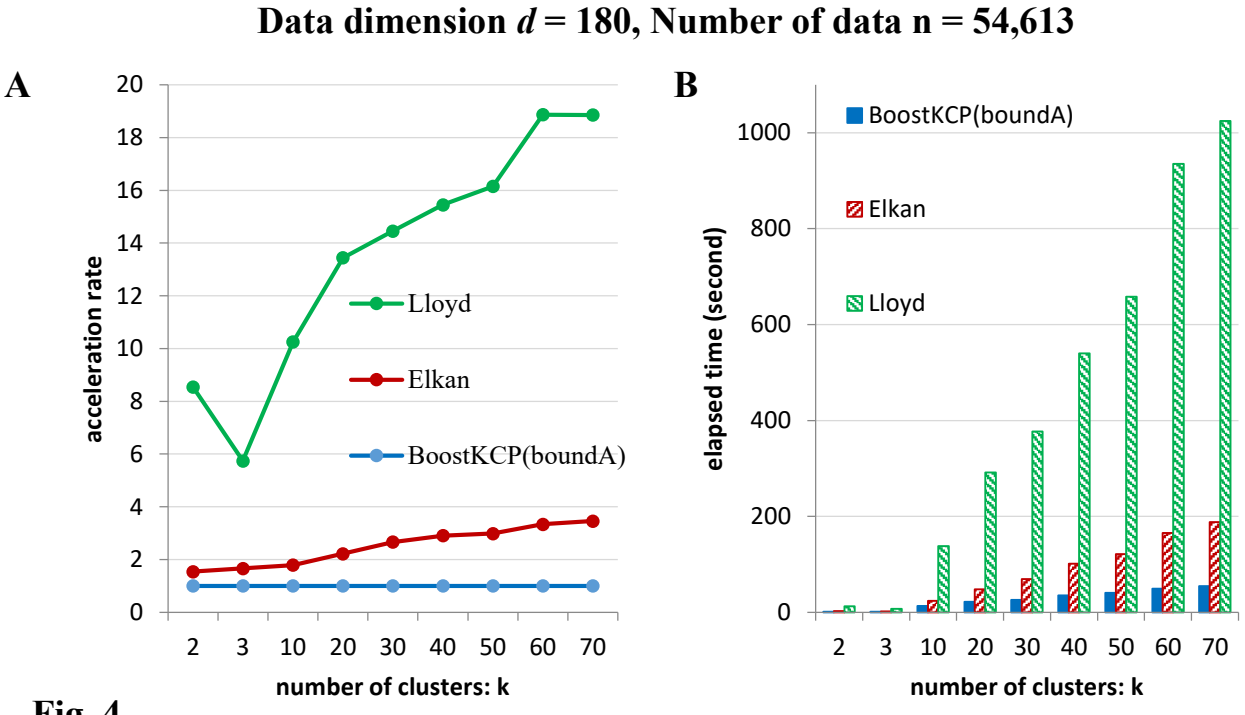


Fig. 4.

Performance improvement by BoostKCP (boundA) using gene expression data of dimension d = 180 to group the data into k (=2, 3, 10, 20, 30, . . ., 70) clusters.

(A) Acceleration rates by BoostKCP (boundA) for each of Elkan's and Lloyd's algorithms.

(B) Average elapsed time of 10 trials.

I also applied BoostKCP (boundA) and Elkan's algorithm to a data set of handwritten letters (d = 784) to obtain 78 (= k) groups. The average acceleration rate of the 10 trials was high (4.76 – 8.16) presumably because the number of clusters was large. Fig. 5 shows the elapsed time, acceleration rate, and number of iterations for each of the ten trials. The iteration numbers are likely to be smaller than those in Fig. 3A because the images of the handwritten letters are grouped inherently. In general, the number of iterations depends on individual data, and it tends to be smaller when the focal data have inherently discriminating groups of similar vectors that are relatively easier to categorize. In contrast, randomly generated data avoid this data skewness; thus, the algorithms spend more time searching for centroids.

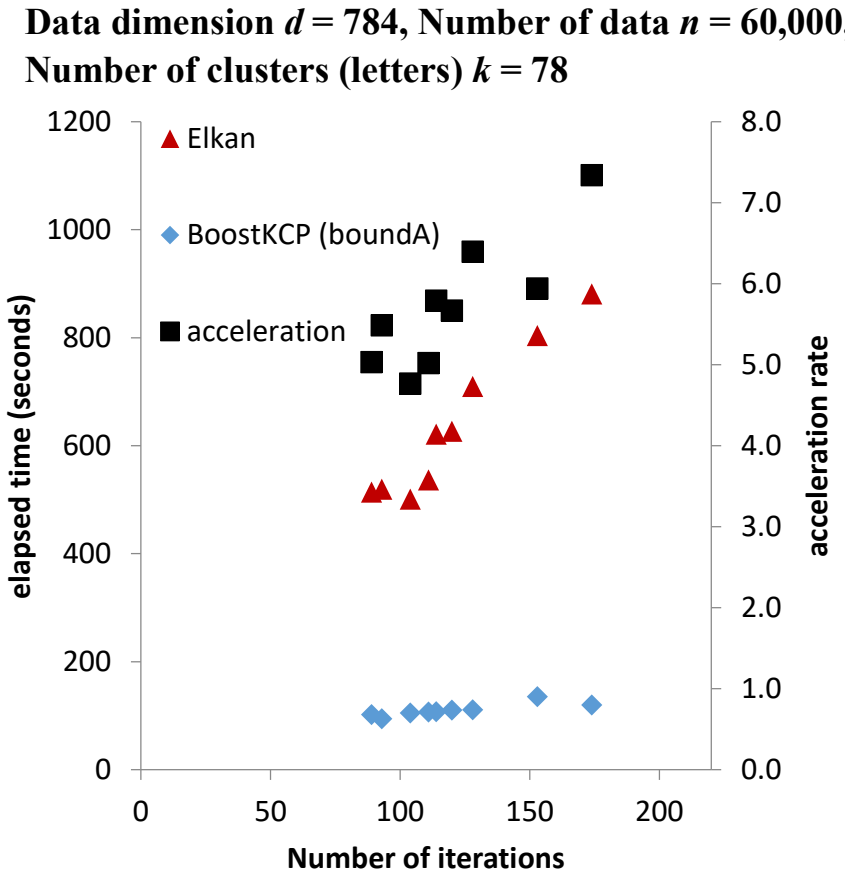


Fig. 5.
The elapsed time, acceleration rate, and number of iterations of each of ten attempts to cluster handwritten letter images of dimension 784 (=d) into 78 (=k) groups using BoostKCP (boundA) and Elkan's algorithm.

I have so far examined situations when the number of clusters (k) ranges from two to 78 simply because these numbers of groups are of interest in real biological applications. I here investigate whether BoostKCP (boundA) outperforms Elkan's and Lloyd's algorithms for larger values of k, such as k = 100 and 500. Indeed, Fig. 6 illustrates that BoostKCP (boundA) was the winner when the three algorithms were used to cluster the nucleosome positioning data of dimension d = 10, 20, 50, 101, and 201 into k = 100 and 500 groups.

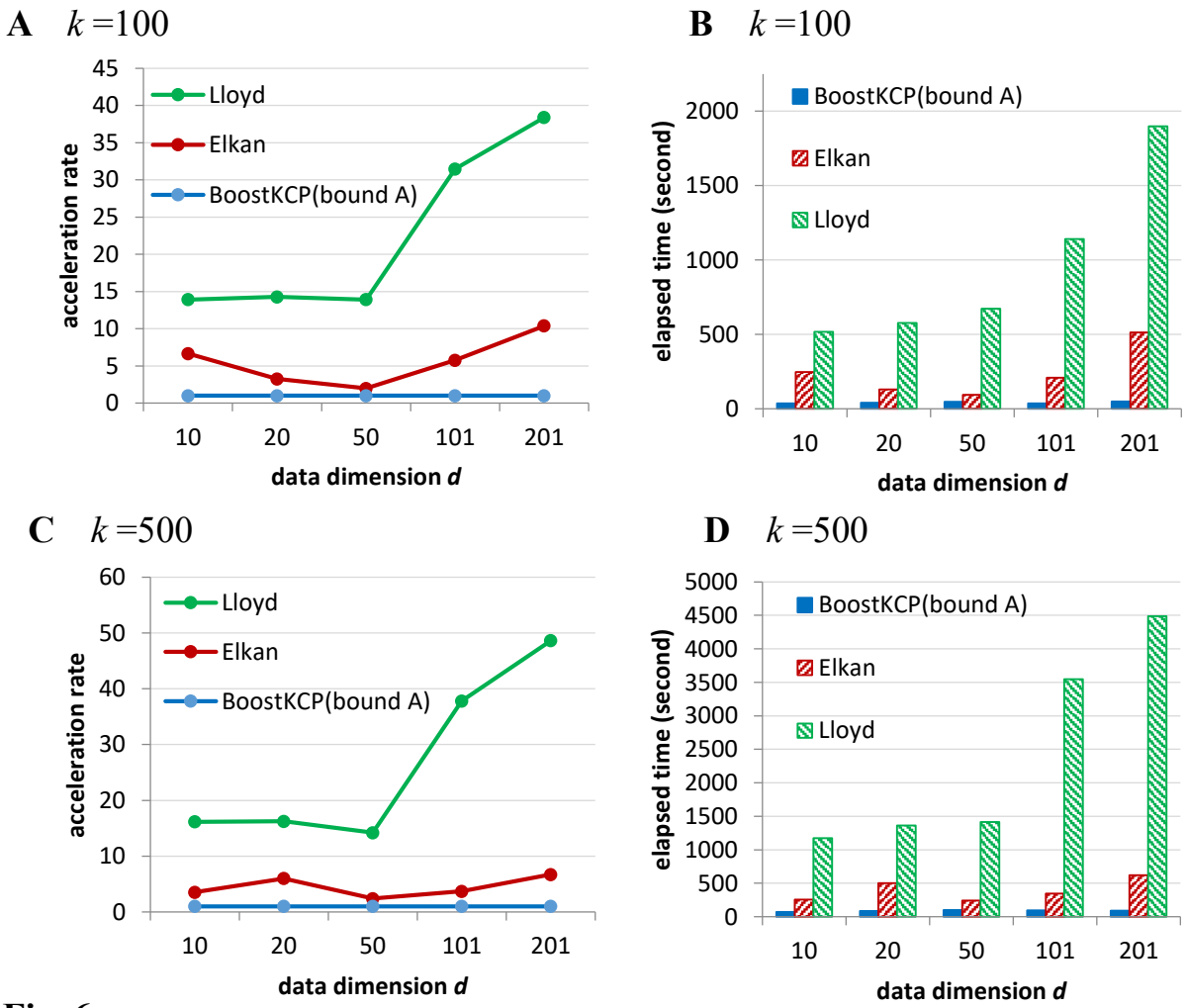


Fig. 6.

Performance improvement by BoostKCP (boundA) using nucleosome positioning data of dimension d = 10, 20, 50, 101, and 201 to group the data into k = 100 and 500 clusters. (A,C) Acceleration rates by BoostKCP (boundA) for each of Elkan's and Lloyd's algorithms when k = 100 (A) and k = 500 (C). (B,D) Average elapsed time of ten trials for BoostKCP (boundA), Elkan's, and Lloyd's algorithms when k = 100 (B) and k = 500 (D).

Conclusions and Discussion

High-dimensional data, such as epigenome data, nucleosome positioning, and gene expression patterns, are quite common in biological research. *K*-means clustering using the Pearson correlation and standardized Euclidean distances has proven useful for obtaining novel insight from such large-scale biological data sets; however, it is likely to be a computationally intense task, thus demanding a method for accelerating computational performance for high-dimensional biological data. I have addressed the problem of eliminating unnecessary calculations associated with the *k*-means clustering algorithm. In this chapter, I introduced BoostKCP, a simple but powerful heuristic method that has proved useful for reducing the computational time. I applied BoostKCP to nucleosome positioning signal data sets and other two types of real biological data sets of dimension d = 10, 20, 50, 101, 180, 201, 501, 784, 1,001 and 2,001 to perform *k*-clustering for k = 2, 3, 10, 20, 30, 40, 50, 60, 70, 78, 100, and 500. BoostKCP outperformed Elkan's, Lloyd's, and Hamerly's algorithms in most cases. My concept is also applicable to *k*-medians clustering, which uses the median of points in a cluster as the cluster representative, and this method is applied frequently to generate tight clusters.

Chapter 2

A linear time algorithm for detecting long genomic regions enriched with a specific combination of epigenetic states

Introduction

This chapter is a modified version of my paper "A linear time algorithm for detecting long genomic regions enriched with a specific combination of epigenetic states" [81].

Epigenetic modifications have been shown to play a vital role in regulating gene expression. Recent genome-wide studies have revealed that in vertebrates, although most CpG sites in DNA sequences are highly methylated, hypomethylated CpG islands proximal to genes are involved in regulating gene expression [82]. Specifically, hypermethylated CpG islands in promoter regions are relevant to gene silencing, while hypomethylated CpG islands are in an active or permissive state for transcription [83]. In addition to cytosine methylation of CpG sites, some histone modifications around promoter regions also are known to affect the regulation of gene expression [84], [85].

It was found recently that long hypomethylated regions enriched with H3K27me3 were likely to overlap with regions encoding key genes essential for cell development and differentiation in human embryonic stem cells [86], mouse hematopoietic stem cells [87], early *Xenopus tropicalis* embryos demonstrates [88], and medaka fish blastula (half-day) embryos [25]. Although many hypomethylated domains (HMD) are subjected to modification of the active histone mark H3K4me2 that promotes gene expression [89]–[92], it is remarkable that ~300 HMDs of length >4 kb rarely have H3K4me2 histone marks but have repressive H3K27me3 histone marks, and are found in association mostly with developmental genes [25]. Promoters in HMD with H3K27me3 marks (called, "K27HMD") are in a 'poised' state, in which the genes are not simply silenced but are ready for activation immediately during cell differentiation, which is important for sustaining the pluripotency of pluripotent cells [23], [24]. Figure

7 shows four examples of long K27HMD regions that include developmental genes such as *cbx4*, *cbx8*, *hoxa* genes, *six2*, *hnf6*, and *zic1/4*.

Thus, there has been considerable interest in long K27HMD regions with biologically important characteristics. However, computational methods for detecting long K27HMD regions are still heuristic and *ad hoc*, emphasizing the need to develop an effective algorithm from a profound background in computation theory. For example, to identify K27HMD, Nakamura et al. proposed a heuristic method that used certain *ad hoc* parameter settings to define hypomethylated regions and H3K27me3 peak detection [25]. The method is not guaranteed to output K27HMD regions longer than a given threshold, and it often generates regions of differing lengths. ChromHMM [42] is a statistical method that classifies epigenetic modifications into classes of combinations and divides a DNA sequence into sub-regions such that each sub-region has a uniform combination of epigenetic states while neighboring sub-regions have distinct characteristics. ChromHMM has been used successfully to partition regions surrounding genes into active/inactive promoters, exons, and introns by analyzing epigenetic codes. Although ChromHMM can be used for K27HMD detection by setting its parameters to find regions that are hypomethylated and marked by H3K27me3, ChromHMM often generates many short regions and thus is not suitable for detecting large K27HMD regions. Overall, these previous methods have simply not been designed to output regions of lengths greater than or equal to a given minimum threshold.

To address this problem, I propose a linear time algorithm for calculating a set of non-overlapping regions such that the set maximizes the score of focal combinations of epigenetic modifications (*e.g.*, K27HMD) and the length of each region is greater than

or equal to a given minimum threshold (*e.g.*, 4 kb). I define the score of a focal combination of epigenetic modifications at each DNA position as the similarity between the vector of focal epigenetic states and the vector of raw epigenetic states at the position. I then define the similarity score of a set of regions as the sum of similarity scores of all positions in the set. This method solves several issues in previous heuristic methods because it allows us to set a minimum region length for detecting 'long' regions of biological importance and guarantees the output of an optimal set of long regions that maximizes an objective function.

I implemented the algorithm. I call the program CSMinfinder (Chromatin State with minimum length finder). With CSMinfinder, I identified large K27HMD regions in the medaka and human genomes [25], [93], [94] that overlapped many developmental genes. CSMinfinder can be applied to epigenetic data from other vertebrates for understanding cell development and differentiation.

CSMinfinder runs in time proportional to the size of the genome, and it can process vertebrate genomes in feasible amounts of time. Although I applied CSMinfinder specifically to K27HMD, it can be used for the detection of regions with other types of epigenetic combinations by defining the vector of focal epigenetic states appropriately.

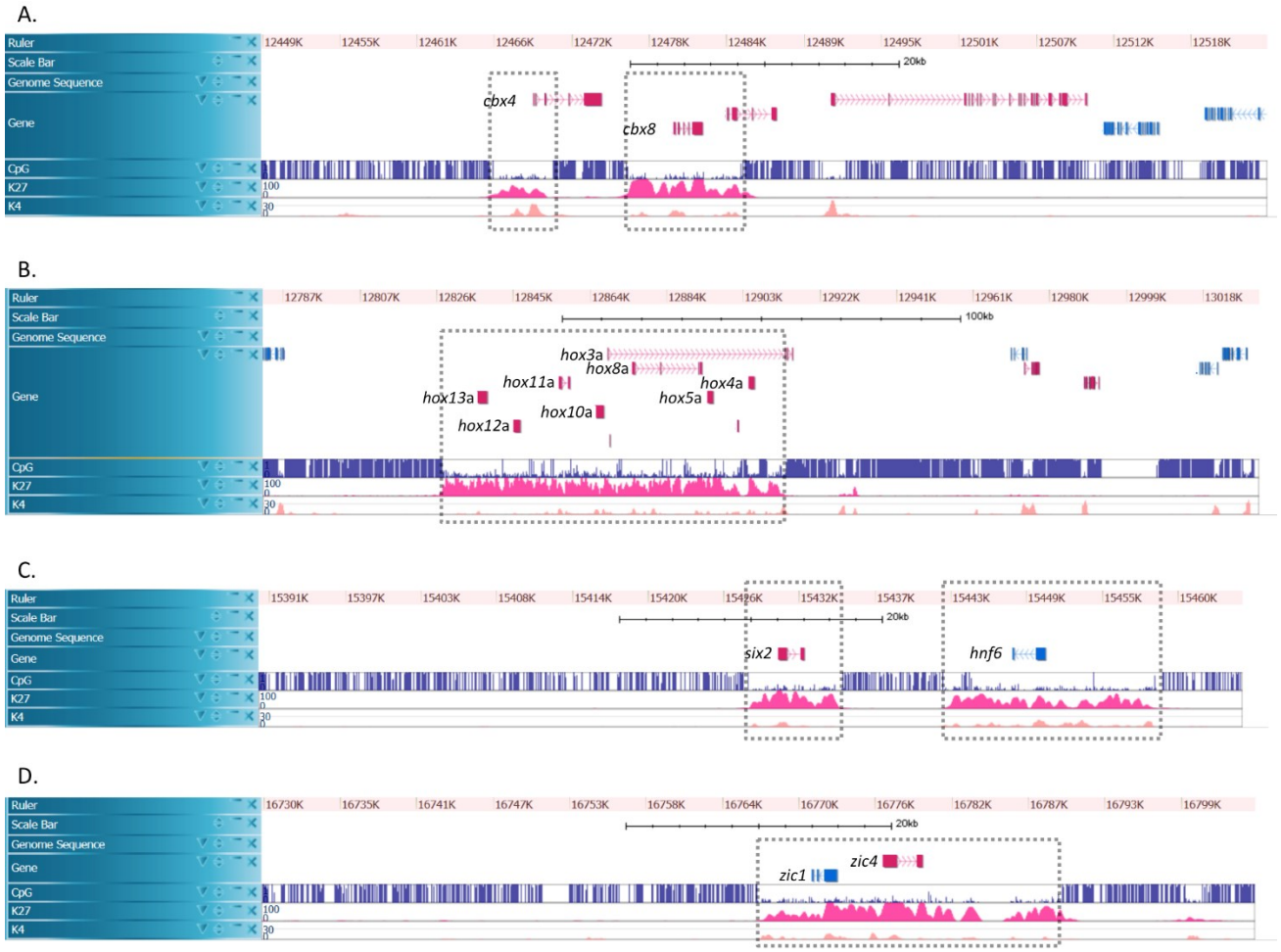


Fig. 7. Examples of long K27HMD regions in the medaka genome

Examples of K27HMD regions enclosed in dashed boxes. Each screen capture shows an image in a medaka genome browser that displays tracks of gene structures, CpG methylation levels observed by bisulfite sequencing, and levels of H3K27me3 and H3K4me2 in blastula cells (half-day embryos). **A.** A K27HMD region of length ~4 kbp with *cbx4*, and a ~8 kbp region with *cbx8*. **B.** A large region of length ~90 kbp with *hoxa* genes. **C.** A ~6 kbp region with *six2*, and a ~14 kbp region with *hnf6*. **D.** A ~20 kbp region with *zic1* and *zic4*.

Methods

To detect long regions of focal epigenetic states, I formulated this as a problem of finding an optimal set of disjoint (non-overlapping) regions in a sequence that maximizes the sum of similarity scores in all regions. My method calculates a similarity score between a vector of epigenetic modifications at each position and the feature vector of a focal epigenetic state, such as K27HMD, and outputs the set of regions with the highest sum of similarity scores.

Calculating a similarity vector

I need to generate a modification vector at each position from epigenomic signal data. For example, to create benchmark datasets in this study, I binarized the modification signal level at each position using BinarizeSignal in ChromHMM [42], which classified the signal at each position into 0 or 1 according to a Poisson background model. Subsequently, I defined a modification vector as the vector with binary scores of modifications at each position.

Definition 1. Let $w_1, w_2 \ldots, w_n$ be non-overlapping windows of the same length (*e.g.*, 200 bp in this study) in a DNA sequence. Let $s_i^1, \ldots, s_i^k$ be binary or real-valued signals of k modifications in window i. The modification vector of w_i is defined as $M_i = (s_i^1, \ldots, s_i^k)$. Let F denote the feature vector of a focal modification pattern with k elements. The similarity score of M_i and F is defined as their inner product minus a given threshold τ.

Example. Suppose that $k = 3$, $\tau = 1.3$, $F = (1,1,0)$, $M_1 = (1,1,0)$, $M_2 = (1,0,1)$ and $M_3 = (0,0,1)$. Similarity scores of F and M_i are 0.7, -0.3, and -1.3 for $i = 1,2,3$.

When the inner product of M_i and F is positive for all $i = 1, \dots, n$, the optimal set of regions that maximizes the sum of similarity scores in the regions becomes the entire region, $[1, n]$, which may not be informative. If we want to select a set of regions whose modification vectors are closer to the feature vector F, we can set the threshold τ to an appropriate positive value to yield a negative similarity score for the inner product that is lower than τ. Positions with negative similarity scores are less likely to be included in the optimal set of regions. A higher threshold is likely to divide the entire genome into smaller regions with a higher precision, while a lower threshold yields an opposite trend. In this manner, for a series of windows $w_1, w_2 \dots, w_n$ in a DNA sequence, we generate a series of similarity scores.

Detecting an optimal set of disjoint regions

To detect regions of focal epigenetic states such as K27HMD, I present an algorithm for calculating an optimal set of disjoint regions in a sequence that maximizes the sum of similarity scores for all regions. In addition, to identify sufficiently long regions, I define a minimum length threshold of regions such that each region is longer than or equal to the minimum length. The problem can be defined as follows.

Definition 2. Let L =$\{L_i \mid i = 1,2,\dots,n\}$ be a series of real valued weights L_i (*e.g.*, similarity scores). Let C be a series of non-overlapping regions I_j $(j = 1,\dots k)$ of L such that the length of each I_j is greater than or equal to a given minimum threshold m_1, and the length of the interval between I_{j-1} and I_j is greater than or equal to another given minimum threshold m_0. That is, C is a series of regions of the form $\{[a_1, b_1], \cdots [a_k, b_k]\}$ $(1 \leq a_1 < b_1 < a_2 < b_2 \cdots < a_k < b_k \leq \text{n})$ such that

1. $a_t + m_1 - 1 \leq b_t$ for $t = 1,\dots,k$ (the minimum length constraint on regions),
2. $b_{t-1} + m_0 < a_t$ for $t = 2,\dots,k$ (the minimum length constraint on intervals between regions), and
3. $a_1 = 1$ or $a_1 > m_0$ (the first region start at position 1 or at position larger than m_0).

Readers may find the last condition strange because it appears to disallow the situation that the first region starts at position $a_1 \leq m_0$. I used the condition to simplify the presentation of my linear-time algorithm, which is described later. To obtain such an optimal series of regions that the first region starts at $a_1 \leq m_0$, for example, you can temporarily add m_0 negative weights in front of L, calculate the optimal series, and restore the coordinate.

To calculate a C that maximizes the sum of weights in C, $\sum_{i\in I\in C} L_i$, I used a dynamic programming algorithm developed by Csurös [95]. Here, I outline the algorithm.

Definition 3. I assume that all series meet the conditions given in Definition 2. Let $w(C)$ denote the sum of weights in C, $\sum_{i\in I\in C} L_i$. I consider two cases: that in which the last region of C ends at i and that in which it does not. When the last region does not end at i, let $C^0_{i,m}$ denote a series of regions that maximizes $w(C)$ among all series, such that the last region ends at position $b_k \leq i-m$, where $m \geq 1$. When the last region ends at i, let $C^1_{i,m}$ denote a series of regions that maximizes $w(C)$ among all series, such that the last region is of length $\geq m$ (≥ 1); specifically, $a_k + m - 1 \leq i$ $(= b_k)$.

Example. When $i = 12$, and $L = (1,1,-3,1,1,-3,1,1,1,1,-2,1)$, we have

$$C^0_{12,1} = \{[1,2],[4,5],[7,10]\},\ C^0_{12,4} = \{[1,2],[4,5],[7,8]\},$$

$$C^1_{12,7} = \{[1,2],[4,12]\},\ C^1_{12,12} = \{[1,12]\}\ .$$

According to this definition, C maximizing $w(C)$ is either $C^0_{n,1}$ or C^1_{n,m_1}. For calculating these two series, I define here $w(C^0_{i,m})$ and $w(C^1_{i,m})$ recursively for $i = 1,\dots,n$ and $m \geq 1$.

Definition 4. I define the following four types of weight sums, $W^0_{short}(i)$, $W^0_{long}(i)$, $W^1_{short}(i)$, and $W^1_{long}(i)$, depending on whether the last region ends at i or not (denoted as 1 or 0, respectively) and whether the minimum length constraint is satisfied or not (denoted as *long* or *short*, respectively):

$W^0_{short}(1) = 0,\ W^1_{short}(1) = L_1,$

$W^0_{short}(i) = \ w(C^0_{i,1}),\ W^0_{long}(i) = \ w(C^0_{i,m_0}),$

$W^1_{short}(i) = \ w(C^1_{i,1}),\ W^1_{long}(i) = \ w(C^1_{i,m_1})$

Csurös showed that these four types of weight sums can be calculated recursively as follows [95]:

$W^0_{short}(i) = max\{W^0_{short}(i-1), W^1_{long}(i-1)\}$ for $i \in [2, n]$

$W^1_{short}(i) = L_i + max\{W^0_{long}(i-1), W^1_{short}(i-1)\}$ for $i \in [2, n]$

$W^0_{long}(i) = W^0_{short}(i - m_0 + 1)$ for $i \in [m_0, n]$

$W^1_{long}(i) = W^1_{short}(i - m_1 + 1) + \sum_{j=i-m_1+2}^{i} L_j$ for $i \in [m_1, n]$

Recall that C maximizing $w(C)$ is either $C^0_{n,1}$ or C^1_{n,m_1}. From $W^1_{long}(n)$, I can build the series of regions, C^1_{n,m_1}, by tracing back the process of calculating $W^1_{long}(n)$. Similarly, from $W^0_{short}(n)$, I can obtain $C^0_{n,1}$.

I implemented the above idea. I call the program CSMinfinder.

Results

Data sets

To compare CSMinfinder with other available methods for detecting large K27HMD, I used real biological datasets from the medaka-fish and human genomes, each of which was a set of vectors of DNA methylation levels at CpG sites, determined by bisulfite sequencing, and H3K4me2 and H3K27me3 histone modification Chip-seq data [25]. I set the window size to 200 bp, normalized the data using a Poisson distribution model, and set the binarized score of a window to 1 if its probability was < 0.0001 and to 0 otherwise.

Detecting large K27HMD in medaka epigenomic data

I compared CSMinfinder with ChromHMM [42] and Nakamura's method [25].

- Using ChromHMM, I estimated six chromatin states and divided the given DNA sequence into these six states. Specifically, ChromHMM asks users to input the number of epigenetic states beforehand. Thus, I started with inputting a small number into ChromHMM, increased the number gradually one by one until I found a state similar to K27HMD, hypo-methylated DNA modification and H3K27me3 histone modification, and called the number *sufficient.* Inputting a value larger than the sufficient number into ChromHMM did not make much sense because it just output a state similar to K27HMD. The sufficient number was six. Among the six states, one represented hypomethylated DNA modifications and the H3K27me3 histone modification. I therefore treated the state as K27HMD.
- Nakamura's method detects a hypomethylated domain on a DNA sequence that has more than nine contiguous CpG sites with low methylation (methylation level <0.4)

and no more than four contiguous highly methylated CpG sites. Parameters are selected heuristically. A hypomethylated domain is treated as a K27HMD if it contains H3K27me3 peaks detected by QuEST [96], such that each peak is more than three times larger than the average.

- In CSMinfinder, I used two types of minimum length thresholds, 4 kbp and 8 kbp, to evaluate the effect of this constraint. I set the minimum length of any interval between regions to 600 bp.

Comparing the performance in detection of large K27HMD around genes in the medaka genome

Large K27HMD regions of length >4 kbp suppress the expression of many developmental genes [25]. Thus, I verified the effectiveness of CSMinfinder for detecting large K27HMD regions surrounding genes in the medaka genome. Nakamura's method could detect 246 large K27HMD regions containing the promoter regions of developmental genes (e.g., *hox* clusters) that were relevant to transcriptional regulation and the developmental process. CSMinfinder detected 911 K27HMD regions, and of these, 386 regions contained promoter regions of >4 kbp in size and contained 242 of the 246 regions identified using Nakamura's method. Indeed, CSMinfinder's regions covered 91% of bases in the entire regions detected by Nakamura's method. Specifically, although the exact boundaries of individual regions estimated by the two methods were unlikely to be consistent, these regions largely overlapped each other. These results demonstrate the high concordance between CSMinfinder and Nakamura's methods as well as the ability of CSMinfinder to identify more K27HMD regions than did Nakamura's method.

I assessed the quality of each K27HMD region in terms of their low average DNA methylation level because this property is considered to be essential in maintaining the suppression of developmental gene expression in embryonic cells [25]. Indeed, Figure 8 shows the tendency of the average methylation level in the vertical axis to become lower for a longer K27HMD region, the length of which is displayed in the horizontal axis. This trend was also observed with all three methods.

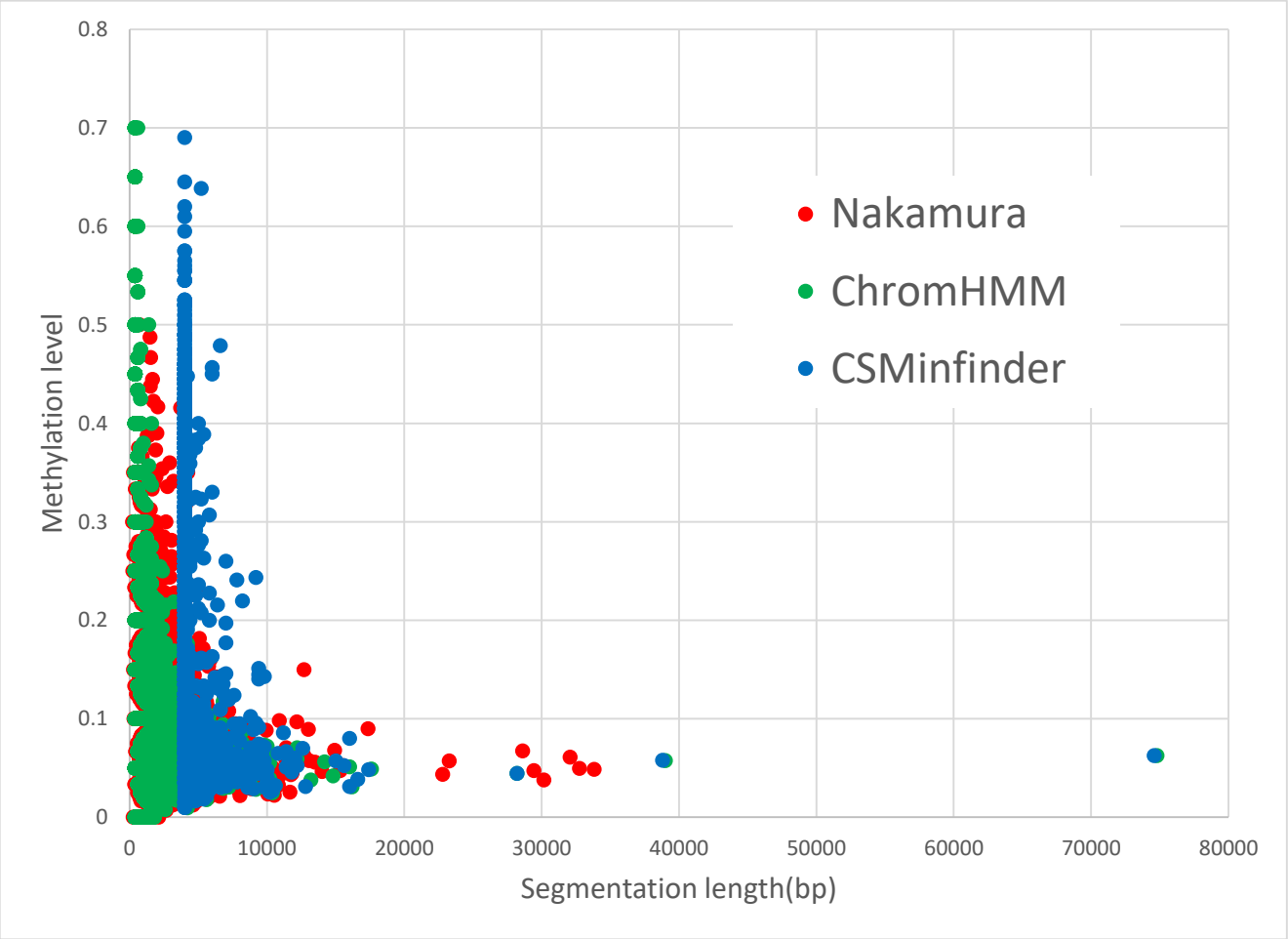


Fig. 8. - Lengths and average methylation levels of K27HMD regions in the medaka genome

Each dot represents a region that is identified by CSMinfinder, ChromHMM, and Nakamura's method in the medaka genome. The *x*-axis shows the length of a K27HMD region and the *y*-axis presents the average methylation level of the region.

I then compared the performance of the three methods by examining the length distributions of K27HMD regions in the medaka genome. Figure 9A shows the length distributions of large K27HMD regions (>4 kb in size) estimated by each of the three methods. Setting the minimum length threshold to 4 kbp in CSMinfinder detected more regions of length > 6 kbp but fewer regions of length > 7 kbp compared with Nakamura's method. CSMinfinder can output longer regions by setting the minimum length threshold to a higher value. For example, setting the minimum length to 8 kbp, CSMinfinder found more regions than did Nakamura's method (Figure 9C).

Analysis of large K27HMD regions in human epigenomic data

I also compared CSMinfinder with the other two for processing human epigenomic data. For ChromHMM, I calculated the sufficient number for the human data according to the procedure described before, and I classified epigenetic modification data into seven states rather than six so as to identify a state similar to K27HMD. The sufficient numbers of epigenetic states in the human and medaka data differed due to the difference in data quality. The sufficient number in the medaka data was smaller than that in the human data presumably because epigenetic state signals in the medaka data were clearer.

In CSMinfinder, I set the minimum length threshold to 8 kbp and the interval between regions to 600 bp. I also searched an ideal value of threshold τ by repeated trials to detect large continual regions, and I set τ to 1.4 and 1.6 in the respective medaka and human data.

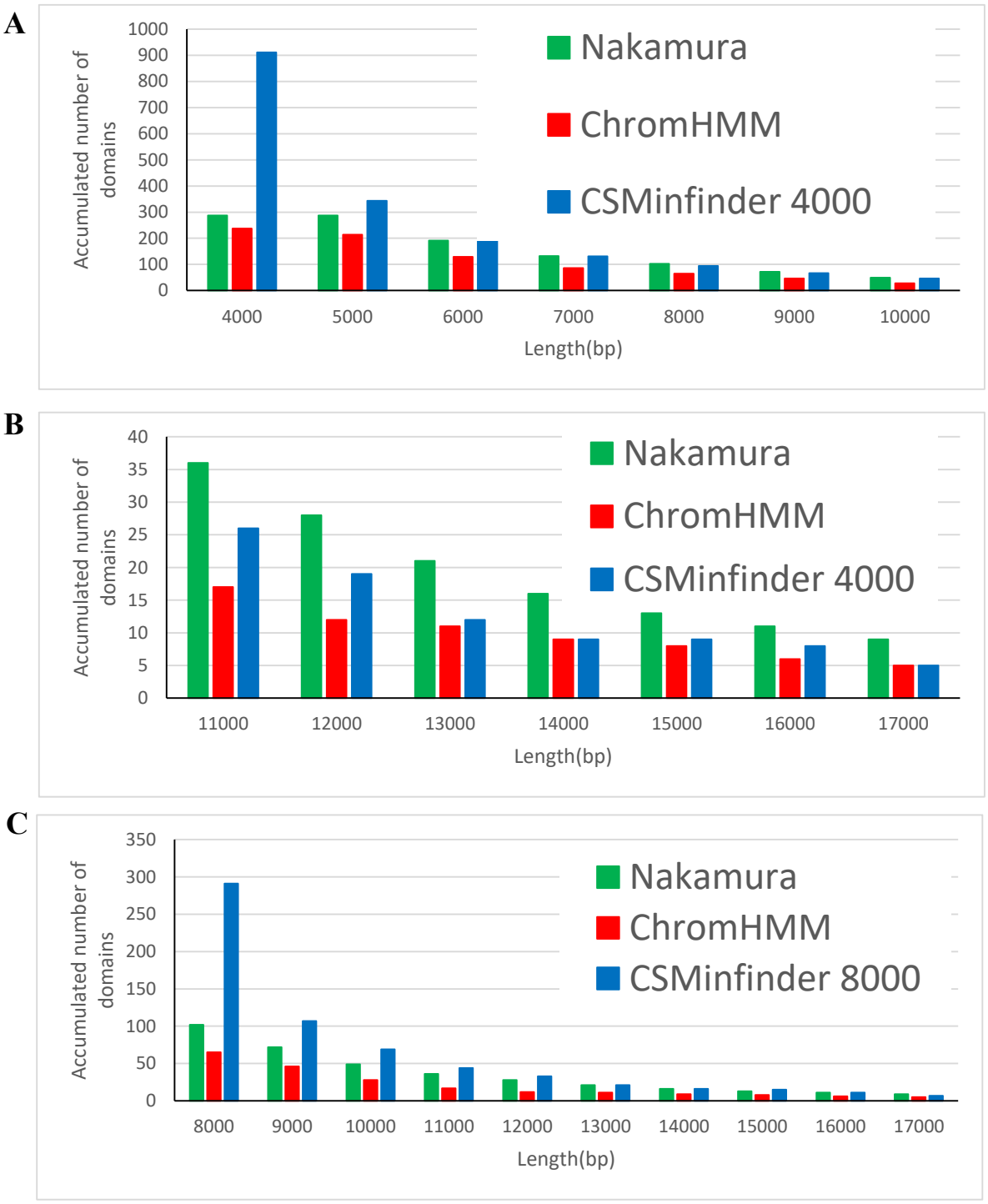


Fig. 9. Length distribution of large K27HMD regions in the medaka genome

Comparison between CSMinfinder (minimum length threshold of 4 kbp), ChromHMM, and Nakamura's method. The *x*-axis shows the minimum length of K27HMD regions, and the *y*-axis shows the accumulated number of K27HMD regions longer than or equal to the threshold in the *x*-axis. Because of the space limitations, the histogram is divided into two sub-histograms **A** (threshold is $\leq$ 10 kbp) and **B** (threshold $\geq$ 11 kbp). **C.** In this case, I set the minimum threshold to 8 kbp using CSMinfinder.

Because the human genome is longer than the medaka genome, I focused on large K27HMD regions of length > 8 kbp. Nakamura's method detected 314 regions, and CSMinfinder identified 542 regions, including 291 of those found using Nakamura's method. Again, there was high concordance between the results obtained by the two methods. Figure 10 shows examples of large K27HMD regions detected around developmental genes. Although CSMinfinder and Nakamura's method yielded slightly different regions with distinct boundaries in the output, each created regions of similar sizes. In contrast, ChromHMM yielded shorter regions than the other two did. Specifically, I compared the length distribution of large K27HMDregions estimated by each of the three methods (Figure 11). I found that CSMinfinder and Nakamura's method were comparable. Precisely, although the number of extremely large regions longer than 12 kbp is slightly smaller than the number found by Nakamura's method, CSMinfinder could detect similar numbers of large regions between 8 kbp to 12 kbp. Later I will discuss the reason why ChromHMM were inferior to the other two methods.

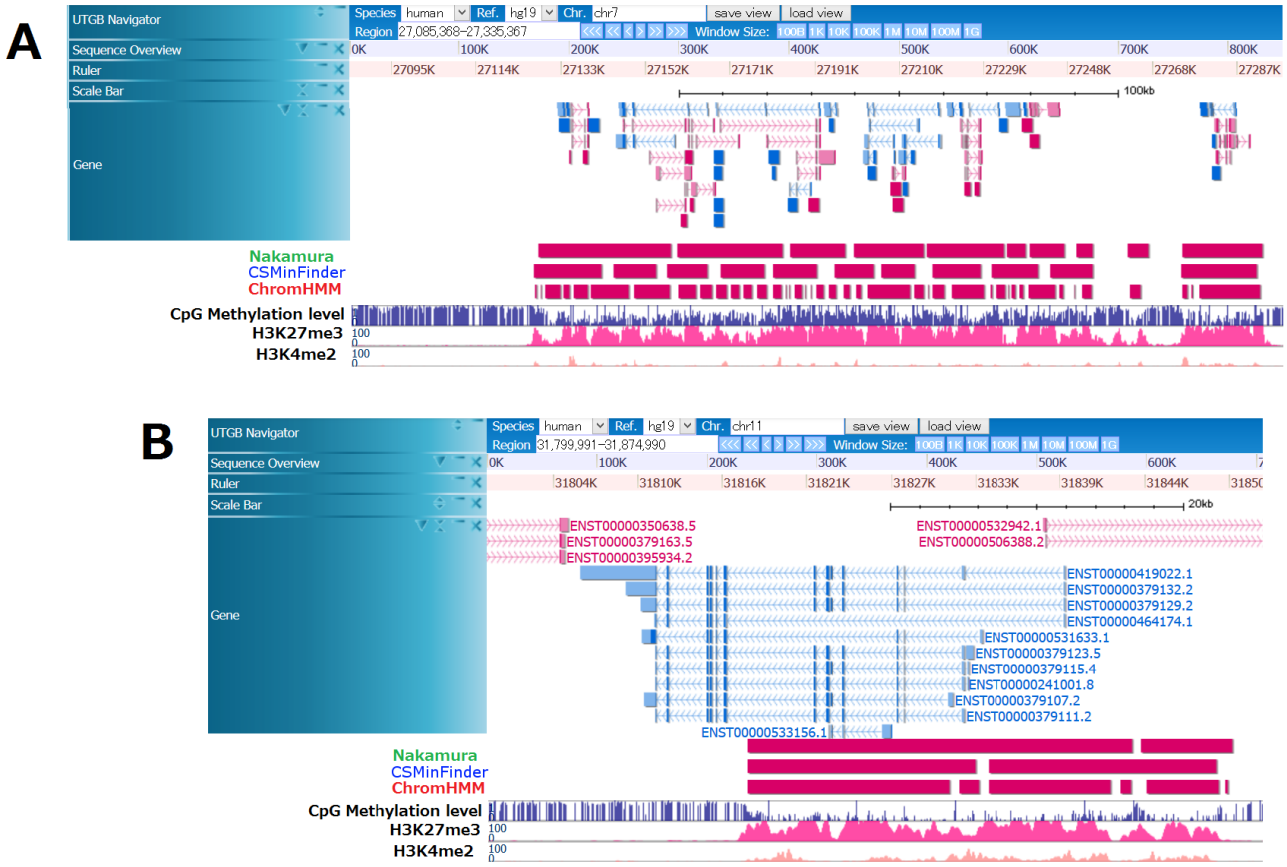


Fig. 10. Examples of large K27HMD regions around developmental genes in the human genome.

A. The figure displays large K27HMD in the human chromosome 7 around a cluster of *hox* genes that regulate the body plan of the head-tail axis. ChromHMM yielded much smaller K27HMD regions as output than did the other two methods.

B. These several K27HMD on human chromosome 11 were located around *pax6*, a gene that regulates eye and brain development. CSMinfinder and Nakamura's method detected large K27HMD regions of >4 kbp in size and output large regions that largely overlapped; however, ChromHMM divided these regions into smaller ones.

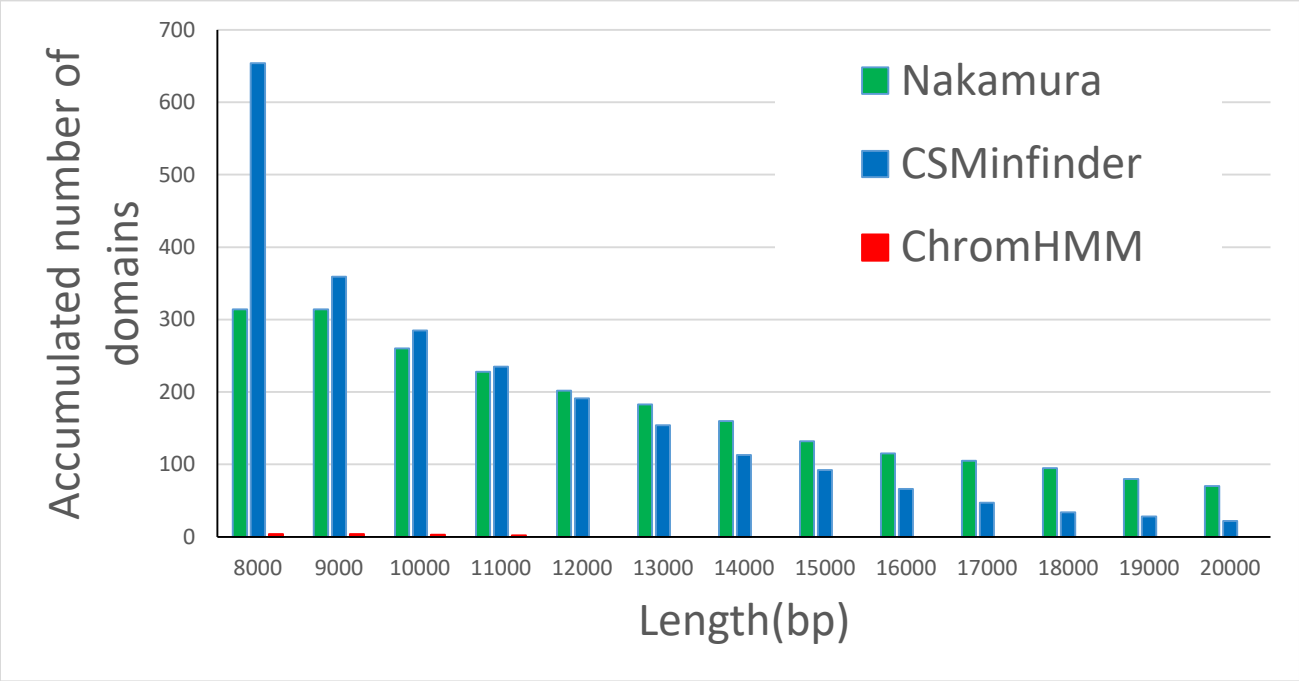


Fig. 11. - Length distribution of large K27HMD regions in the human genome. Comparison between CSMinfinder (minimum length threshold of 8 kbp), ChromHMM, and Nakamura's method. The *x*-axis shows the minimum K27HMD region length threshold, and the *y*-axis shows the accumulated number of K27HMD regions longer than or equal to the threshold on the x-axis.

Computational performance and software availability

I observed the computational performance of CSMinfinder using Intel Xeon CPU E5-2670 processor with a clock rate of 2.60 GHz and 66GB of main memory. The computation time to calculate the optimal series of regions was negligible. Figure 12 shows that the average elapsed time was less than 2 seconds when I processed the epigenetic data of any of human and medaka chromosomes. Furthermore, Figure 12 also illustrates that the elapsed time is almost proportional to the size of each chromosome, thereby confirming experimentally that the worst-case time complexity of the algorithm is linear in the input size. CSMinfinder does not consume a large amount of main memory. CSMinfinder is made available at the following site:

URL: http://mlab.cb.k.u-tokyo.ac.jp/~ichikawa/Segmentation/

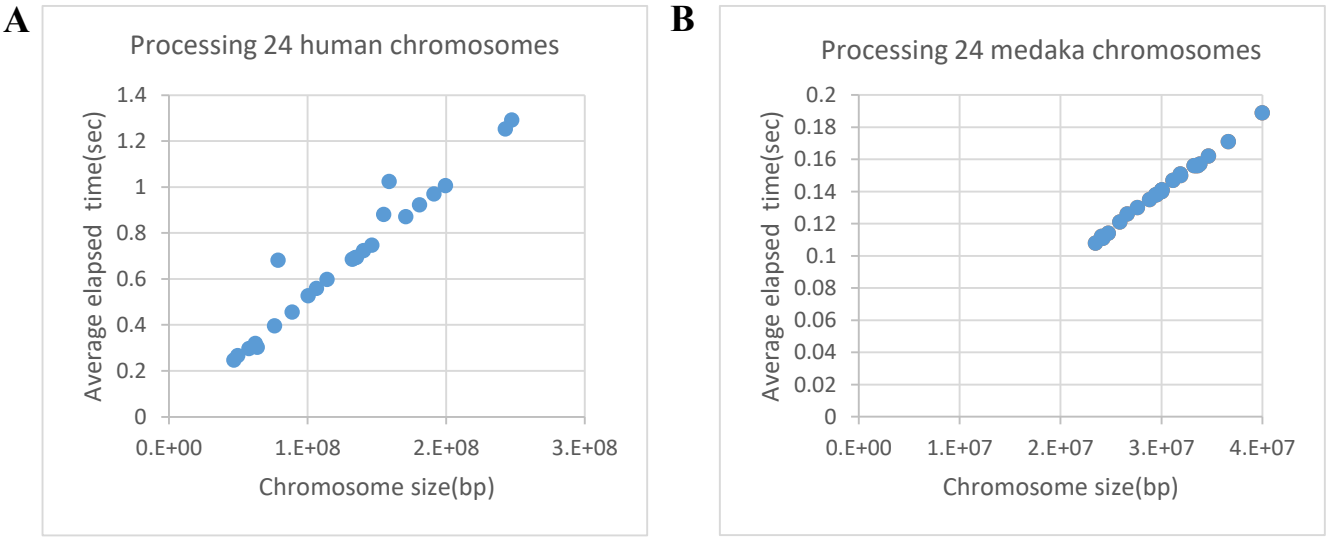


Fig. 12. - Average elapsed time of processing human (A) and medaka (B) chromosomes ten times by using CSMinfinder

The minimum threshold is set to 8 kbp for handing the human genome, and 4 kbp for the medaka genome. Each dot represents a chromosome, the x-axis value shows the size of the chromosome, and the y-axis value is the average elapsed time.

Conclusions and Discussion

In this chapter, I proposed a method that estimates large K27HMD region [25], [86]–[88], [92] by calculating the similarity between the vector of focal epigenetic states and that of raw epigenetic states at each DNA position. The advantage of this algorithm (CSMinfinder) is the output of an optimal series of regions while allowing us to set the minimum length threshold on individual regions. I estimated large K27HMD in the medaka and human genomes and verified that CSMinfinder was comparable to Nakamura's heuristic method [25] designed to detect K27HMD and was advantageous over ChromHMM in terms of the lengths of K27HMD regions.

For the medaka genomic data, ChromHMM performed well and could detect as many long regions as CSMinfinder did; however, for the human genomic data, ChromHMM found a smaller number of large K27HMD regions of length > 8 kbp than the other two methods did. This was likely due to the differences in characteristics between the medaka and human genomic data. In the medaka genome, the data were collected from an inbred stain in which the genomic differences between the two alleles were quite small. Thus, methylation levels were bimodal and were clearly divided into two states, hypomethylated and hypermethylated, making it relatively easy to identify blocks of hypomethylated domains. In the human genome, however, the majority of methylation levels were poised because the human genome is diploid intrinsically and allele-specific methylation is prevalent, making it more difficult to detect clear boundaries between hypermethylated and hypomethylated domains. Although many DNA methylation levels are ambiguous in the human genome, ChromHMM attempts to assign one state to each position. Positions with vague DNA methylation levels are assigned only a

single state by ChromHMM. Thus, ChromHMM is likely to output too many short regions.

One advantage of CSMinfinder is that we can set the minimum region length for specific purposes. For example, in the medaka genome, using an 8-kbp minimum length threshold merged some of the shorter regions that were generated using a 4-kbp minimum threshold into a longer continuous region. Thus, we could obtain longer regions using a higher minimum length threshold. Similarly, we can also adjust the minimum threshold for defining similarity scores between modification vectors and the feature vector for a variety of purposes. Setting the minimum threshold to a lower value generates more regions that are less similar to the feature vector of interest. Having more than one series of regions that may overlap can be informative. We can therefore tune CSMinfinder easily to meet various demands.

In this chapter, I demonstrated the advantages of my algorithm by detecting large K27HMD regions that have attracted much interest because of their importance in characterizing the behavior of developmental genes and confirmed the performance of my algorithm. CSMinfinder is not limited to the identification of large K27HMD regions but can be used for the detection of other large DNA regions that have different types of epigenetic state combinations associated with regulating gene functions.

Chapter 3

De novo assembly of medaka fish genome using SMRT sequencing and construction of chromosome map using Hi-C data

Introduction

This chapter is based on the paper “Centromere evolution and CpG methylation during vertebrate speciation,” in which I am the first author [97].

The medaka, Japanese killifish (Oryzias latipes) is freshwater fish distributed in East Asia including Japan. Medaka has many useful characters for model organism such as small size of whole genome sequence (~800Mb), short generation time and easiness to breeding, and thought to valuable for elucidating fish genome as zebrafish [98], [99]. Especially, some medaka inbred strains which can mate and produce healthy offspring under laboratory conditions established in medaka. Two medaka inbred strains HNI which is a medaka inbred strain from local subpopulations in north Japan, and Hd-rR from south japan are estimated to be diverged in ~18 million years ago (MYA) [100]. About 16 million SNP are discovered between Hd-rR and HNI, and it account for 3.4% of whole DNA sequence [100]–[102]. In spite of the higher mutation rate, Hd-rR and HIN can produce healthy offspring . These inbred strains are thought to be in the middle of speciation and research of structure variants between inbred strains are valuable for resolving mechanism of evolution and differentiation [103]. Sequencing whole medaka genome have been attempted more than ten years ago, and version 1 of the medaka reference genome from Hd-rR inbred strain using Sanger sequencer was reported in 2007 [104].

Past researches using chromatin information in medaka genome have revealed new findings in epigenetics. In human it was known that SNP rate around methylated CpG site is significantly higher than other regions. However, genetic variation between human reference genomes is not sufficiently high to analyze the relation between

methylation and genetic variation. Using high mutation rate between Hd-rR and HNI, W. Qu *et al.* showed that SNP rate around methylated CpG site is also high and "CGCG" motif possibly related to the regulation of hypomethylation [93]. Nucleosome positioning around promoter regions was researched and typical patterns in hypomethylated domains and short DNA motif which regulate nucleosome positioning pattern was discovered [105]. Estimation of nucleosome positioning using DNA sequence was worked well in yeast genome [106], [107], however in vertebrate sequence preference of nucleosome could not be determined. Around methylated transcription starting site DNase I signal have periodical pattern in 180bp interval, in contrast in hypomethylated domains nucleosome positioning around TSS have 200bp interval in medaka genome[105]. It was also revealed that in hypomethylated linker DNA specific 6-mer sequence exist in significantly high probability. The research of long hypomethylated domains with H3K27me3 marks at developmental promoter showed that poised state by epigenetic modifications have an important role in cell development and differentiation [25].

As stated above analysis of chromatin conformation in medaka genome is thought to be led to novel finding in epigenetics. However, in version 1 of the medaka genome contained low-quality regions and 97,933 sequence gaps [104], particularly assembly around centromere regions which contain abundant tandem repeat sequence was difficult by short reads in sanger sequencing. Centromere is the region of a chromosome where combined with spindle fiber in cell division, and have an important role in chromosome separation. In centromere region-specific proteins such as CENP-A are accumulated and construct heterochromatin [108]. Mechanism of regulating composition in centromere was not perfectly elucidated. In yeast specific base sequence

locate in centromere and guide nucleosome positioning [106], [107]. However, in vertebrate it was reported that neocentromeres which are regions which haven't peculiar sequence work as centromere [109], and centromere are thought to be controlled by epigenetic structures.

In my research I used single-molecule real-time sequencing and Hi-C data to construct new medaka genome containing centromere regions which could not be assembled by ver.1 medaka draft genome.

Hi-C is a technology to capture chromatin conformations in genome [110]. In Hi-C method genomes are cross-linked by formaldehyde and fragmented by restriction enzyme. Fragments are ligated and digested. The resulting DNA fragments are sequenced. Hi-C data can detect the chromatin interaction in genome-wide sequence. Recent studies in contact genomics show that the information of chromatin contacts can be used to determine genomic positions and some applications for genome scaffolding by Hi-C data was invented.

In this chapter I constructed medaka genomes of three inbred medaka strains using single-molecule real-time sequence technology. I utilized single nucleotide polymorphism genetic markers, BAC/fosmid-end pairs to anchor contigs to the 24 medaka chromosomes. Additionally, I used Hi-C data to locate contigs which contain centromeric repeats but hardly to be anchored by other methods. To show the comprehensive ness of new draft genome I illustrate some examples, Tol2 elements [111], Y-specific regions [112], [113] and large structure variant [114] which could not be found in version 1 of the medaka genome [104].

Results

Generating long contigs using SMRT sequencing

DNA was collected from adult medaka testes of the Hd-rR, HNI, and HSOK strains. A SMRT sequencer (PacBio RS II) was used to collect ~13.4, ~14.8, and ~5.5 million subreads, with average lengths of 6,519 bp, 3,575 bp, and 10,972 bp, from the Hd-rR, HNI, and HSOK strains, respectively. The three datasets are equivalent to coverages of ~109-, ~66.0-, and ~75.8-fold, assuming a medaka genome size of 800 Mbp. The FALCON assembler [115] was used to generate contigs; the respective N50 contig lengths were ~2.5, ~1.3, and ~3.5 Mbp. The assembled contigs was polished by Quiver [116] and Pilon [117] using Illumina-derived short reads. Then, the new Hd-rR assembly was compared with the medaka genome version 1 that was generated by using Sanger sequencing technology [104], and the high-level sequence identity (99.8%) was confirmed. To assess the large-scale orderings of regions in the contigs, the 19,448 pairs of BAC-end Sanger reads was mapped approximately to the identical Hd-rR contigs in order. Only 0.3% of BAC-end pairs were inconsistent, confirming that the assembled contigs were of high quality.

Chromosome map construction

I used 2,347 single nucleotide polymorphism (SNP) genetic markers to construct a chromosomal map of the Hd-rR strain [104]. Assuming that genetic markers are distributed uniformly, a marker would be available every ~341kbp. Some 90% of contigs were sufficiently long to bear genetic markers; the respective N90 contig lengths of Hd-rR, HNI and HSOK were ~653, ~450, and ~1,102kbp. Thus, I skipped the

traditional step of connecting contigs into longer scaffolds, instead attempting to directly anchor contigs to the 24 medaka chromosomes using genetic markers (Methods).

Certain contigs failed to be anchored to any chromosomes because they did not contain genetic markers. For Hd-rR contig anchoring, I used 48,955 BAC-end pairs and 199,657 fosmid-end pairs that had earlier been collected [104]. By scaffolding Hd-rR contigs connected by multiple BAC/fosmid-end pairs, I was able to anchor additional 23 Hd-rR contigs to chromosomes (Methods). A total of 768 BAC-end pairs and 376 fosmid-end pairs linked the Hd-rR contigs. This suggests that the gaps between contigs are likely to be longer than fosmid clones of median length 37.5kbp, and longer reads would be needed to fill such gaps. I used Hi-C data to locate 11 orphan contigs which could not be anchored onto chromosomes (Methods). I finalized the draft genomes by inserting a 1kbp gaps between neighboring contigs; I term these drafts version 2.2.4. In this version, the total numbers of bases in the contigs anchored to the Hd-rR, HNI, and HSOK chromosomes were ~733.5, ~677, and ~744 Mbp respectively with 491, 717 and 318 gaps. Thus, the quantity of gaps was dramatically lower than the ~100,000 gaps in the previous Sanger-sequence Hd-rR genome assembly.

To demonstrate the comprehensive nature of the current sequences, I examined the distributions of *Tol*2 element insertions. *Tol*2 is 4682bp in length, and represents an example of an early innate autonomous transposon in a vertebrate genome [111]. While the previous Sanger-sequence genome assembly had no full *Tol*2 matches, the new Hd-rR, HNI and HSOK genomes bore 15, 5, and 16 full matches, respectively, in different positions. These occurrences were >99.4% identical to the reference *Tol*2 sequence, implying their horizontal transfer after the divergence of Hd-rR and HNI. Another

example is the Y-specific region carrying *DMY*, the male-determining gene, the first non-mammalian equivalent of *SRY* [112]. *DMY* had mapped to three scaffolds with gaps in the earlier Hd-rR genome (version 1) because of its proximal repetitive elements [113], but I obtained a single contig bearing *DMY* in the version 2.2.4.

Large structural variants between strains

Comparisons among the contigs of the three inbred strains revealed substantial numbers of large SVs including insertions, deletions, duplications, and inversions. The biggest SV is a >15-Mbp inversion in chromosome 11 (Fig. 13), which was suggested [114] but unclear based on the prior Sanger-sequence genome assembly [104]. In the present study, when I anchored contigs onto HNI and HSOK chromosome 11, I identified two pairs of contigs that had two sets of distal genetic markers that were separated by ~16Mb while I found no such pairs in Hd-rR, indicating that the inversion had occurred in the Hd-rR lineage. I determined the inversion breakpoints in focal HNI and HSOK contigs by aligning these contigs with the corresponding region of Hd-rR. Contigs surrounding the breakpoints of the inversion are associated with their contig identifiers (e.g., 83F and 481F). In the HNI genome, the two breakpoints are located at 7F and 143F, whereas in the Hd-rR genome, one breakpoint lies between 83F and 481F, and the other is between 240F and 138F. This is partly because the breakpoints lie in the long repetitive regions shown in Figure 13B.

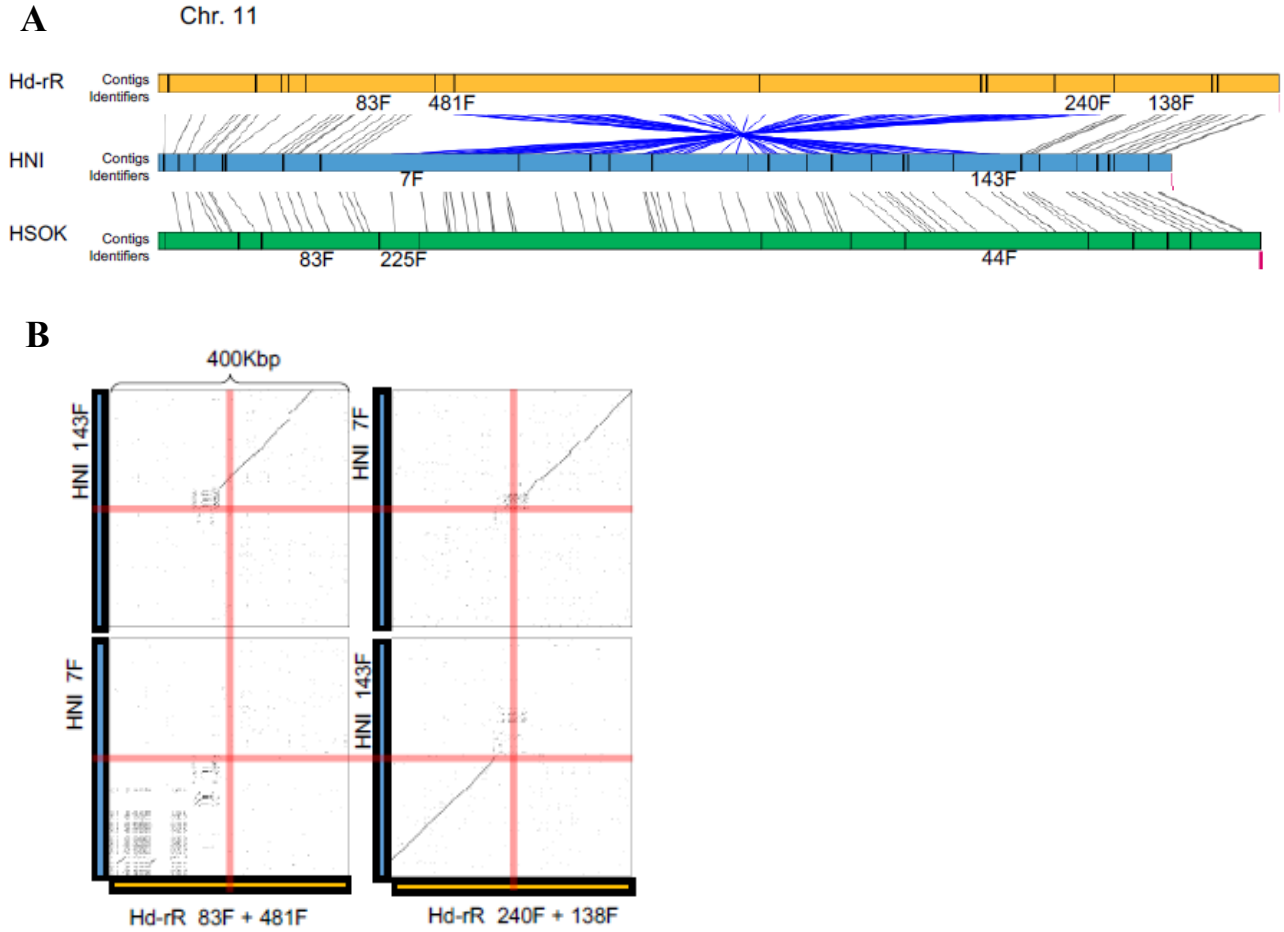


Fig. 13. Large inversion in chr 11

A. An extremely large inversion (>15 Mbp) in chromosome 11 was evident when Hd-rR and HNI were compared. The presence of the inversion was suggested by the Sanger-sequence genome assembly; however, the contigs assigned to chromosomes were not of sufficient length to reveal the boundaries of the inversion.

B. Dot plots comparing the four pairs of Hd-rR and HNI regions that contain the two breakpoints of the inversion. The inversion was surrounded by highly repetitive regions of ~200 kb and ~10 kb in size, which were difficult to detect using short read sequencing technology.

Methods

Data Availability

I deposited the sequence data of SMRT reads and assembled genomes from Hd-rR, HNI, and HSOK at the NCBI SRA (BioProject Accession: PRJNA325079 for Hd-rR, PRJNA325193 for HNI, PRJNA325194 for HSOK), and the in-situ Hi-C reads from Hd-rR and d-rR at NCBI SRA (PRJNA378460 for Hd-rR, PRJNA378464 for d-rR). The accession number of the RNA-seq data for gene prediction is DRA005309, and the accession number of two RNA-seq biological replicates from blastulae of Hd-rR and HNI is SRP116580. The assembled genomes of the three strains, a comparative genomic analysis of the three strains, a medaka gene model, DNA methylation estimation from SMRT sequencing kinetic data, and a web browser for visualizing these datasets are available at http://utgenome.org/medaka_v2/ .

Generating a chromosome map for each strain

I used 2,347 SNP genetic markers to anchor contigs of the three strains to the 24 medaka chromosomes using the alignment software program ispcr (in-silico PCR), which is available at https://github.com/mkasa/klab/blob/master/script/ispcr. I ordered the contigs along each chromosome according to the genetic distances between markers. Some contigs were subsumed by other (longer) contigs; I eliminated the former redundant contigs. I detected 17 misassembled contigs in the Hd-rR strain, 16 in the HNI strain, and 8 in the HSOK strain; all contained genetic markers originating from two different chromosomes. I corrected these misassembled contigs by dividing them into two subcontigs by reference to the genetic markers, and anchored the partitioned (sub)contigs to their respective chromosomes. I also anchored remaining Hd-rR contigs

that were connected by multiple BAC/fosmid-end pairs. Specifically, after considering the estimated median sizes of BAC and fosmid clones (135kbp and 37.5kbp), I used BAC-end (fosmid-end) reads mapping to a position within 150 and 50kbp from one end of a contig. In contrast, for HNI and HSOK, sufficient BAC-end and fosmid-end pairs were unavailable and no Hi-C data were collected. I instead located 44 HNI contigs with no genetic markers to chromosomes by reference to their best matches to Hd-rR contigs. Some Hd-rR, HNI, and HSOK contigs remain unoriented because they were associated with only a single genetic marker, or multiple genetic markers at the same genetic distance apart. I attempted to determine the orientation of each unoriented contig by reference to the orientations of the best-matched contigs in the other strains.

Collecting Hi-C reads

In situ Hi-C was performed as previously described [118] with slight modifications. Samples (~2x10^6 cells of fibroblast or liver/brain from single individuals) were fixed with 1% (v/v) formaldehyde solution. MboI restriction enzyme (NEB) was used for digestion of cross-linked chromatin. After DNA shearing using the S220 Focused-ultrasonicator (Covaris), 300-500bp fragments were selected using AMPure XP beads (Beckman Coulter). End-repair, adapter ligation and library amplification were performed using KAPA Hyper Prep Kit (KAPA BIOSYSTEMS). Libraries were sequenced for 101 cycles from both ends on Illumina HiSeq 1500.

Assembly by Hi-C data

I used Hi-C data to locate 11 orphan contigs which contained centromeric repeats but failed to be anchored onto chromosomes because of the absence of genetic markers on them. First I trained a naïve Bayes classifier to predict the chromosome of each orphan contig considering its contact frequency information with individual chromosomes. For each orphan contig, contact frequency a_i with chromosome i was calculated by the number of Hi-C reads mapped between the contig and chromosome i. The contact frequency variables $a_1, \ldots, a_{24}$ are conditionally independent of each other given the chromosome i.The posterior probability of the orphan contig anchored to chromosome c is

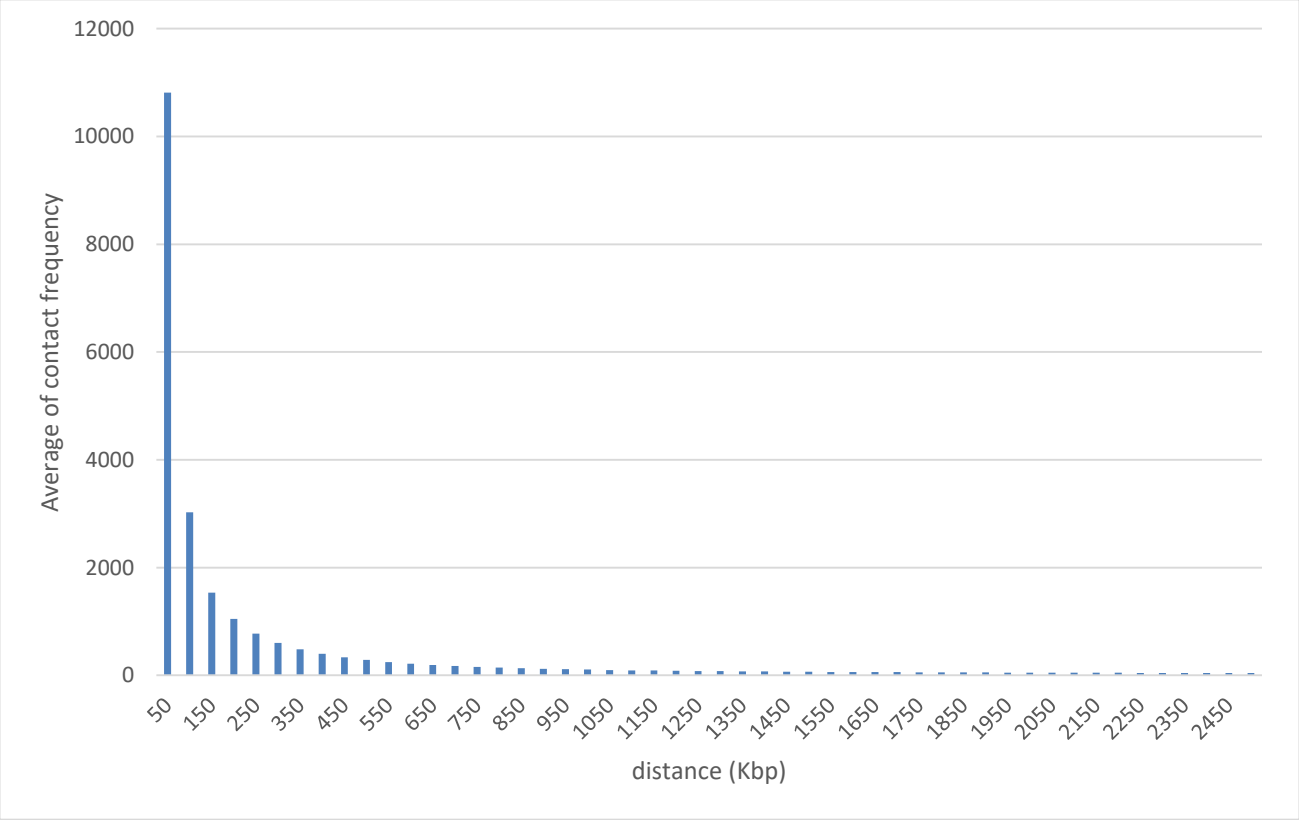
$$p(c|a_1, \ldots a_{24}) = \frac{p(c)\prod_{i=1}^{24} p(a_i|c)}{Z}$$

where $p(c)$ is a prior probability proportional to the number of contigs in chromosome c, $p(a_i|c)$ is a conditional probability of contact frequency a_i under the condition that the orphan contig was anchored to chromosome c and Z is a normalization factor. I verified the correctness of the above naïve Bayes classifier by checking whether 500 contigs that were already anchored by genetic markers were also accurately classified to chromosomes which had the highest posterior probability. Indeed, I confirmed that all contigs could be correctly classified. Thus I assigned the chromosomes to eleven orphan contigs with centromeric repeats by using the naïve Bayes classifier.

Next I predicted the precise positions and orderings of the eleven orphan contigs in their assigned chromosomes. To this end, I utilized the property that, along each chromosome, the contact frequency increased almost exponentially towards one position (Fig. 14). Certainly, the average contact frequency of the 1Mbp region surrounding the position was clearly higher than that outside. According to this property, for each orphan contig

that was anchored by the naïve Bayes classifier, I calculated the contact frequency between the orphan contig and anchored contigs in the chromosome assigned to the orphan contig, and located the orphan contig next to the position which had the highest contact frequency.

Fig14. - Contact frequency distribution between paired-end Hi-C reads.



Conclusions and Discussion

In this chapter, I constructed new draft genome of three inbred medaka strains using single-molecule real-time sequencing. The number of gaps in version 2.2.4 medaka genome are 491, 717 and 318 in Hd-rR, HNI and HSOK and these are dramatically lower than the ~100,000 gaps in previous Hd-rR genome assembly.

Long reads make it possible to assemble regions which have abundant tandem repeats that are hardly resolved by Sanger-sequence genome assembly and determined breakpoints of large inversion in chr11. Additionally, distributions of *Tol*2 elements in each strain can be identified. Copy numbers and positions in chromosomes of *Tol*2 elements were highly diverged and it implies their horizontal transfer after the divergence of Hd-rR and HNI.

I used Hi-C data to locate 11 contigs which could not be anchored on to chromosome by genetic markers. Assembly around centromere regions is still arduous problem even using SMRT sequencing, therefore to clear the centromere sequence using chromatin conformation data by Hi-C seq was verified as useful method.

In version 2.2.4 of the medaka draft genome new insight of centromere evolutions become clear by the precise examination of centromere methylation in each strain. To achieve more precise analysis on chromatin information, our medaka draft genome is thought to be useful resources for future studies.

Concluding Remarks

In my doctoral thesis, I invented two novel algorithms for processing large-scale chromatin information helpful for gain biological insights.

In Chapter 1, I devised “BoostKCP”, an accelerating method for *k*-means clustering using the Pearson correlation distance. I applied BoostKCP and other two accelerating methods to human nucleosome positioning data of various dimension d =10 – 2001 to perform *k*-means clustering for k= 2-500 and compared computational time. In all conditions my algorithm outperformed other methods and 5-26 times faster than ordinary *k*-means clustering without boosting. My accelerating method for pruning unnecessary calculation is specialized to Pearson correlation distance, therefore my algorithm calculates faster than other boosting methods originally designed for Euclidean distance. My algorithm is effective in various situations, especially in high dimension data which take long time without acceleration. Reducing computational time by BoostKCP make it easy to find better clustering conditions and useful for grasping new knowledge from massive biological data.

In Chapter 2, I presented “CSMinfinder”, a method for detecting regions which modified by specific epigenomic combinations. CSMinfinder calculates the similarity between the vector of focal epigenetic states and that of raw epigenetic states at each DNA position and detects an optimal set of regions that maximizes the sum of similarity. The minimum length threshold of each region in CSMinfinder makes it possible to detect continuous regions. I estimated large K27HMD regions using CSMinfinder in the medaka and human genome and showed that my method could detect equivalent regions

as Nakamura's methods and longer regions compared with ChromHMM. My method could detect 242 regions containing the promoter regions of developmental genes. CSMinfinder is also applied to detect other combination of epigenetic modifications.

In Chapter 3, I performed *de novo* assembly of three inbred medaka strains using Hi-C data. Centromeric regions could be anchored using contiguity of chromatin information and more precise investigation into alternation of epigenetic modifications during speciation appear to be possible by new medaka draft genome.

Acknowledgements

I would like to take this opportunity to express my appreciation to cooperators of my research.

Firstly, I would like to express the deepest appreciation and respect to my supervisor, Dr. Shinichi Morishita, for the kindly guidance and encouragement. Without his accurate advice and support this thesis would not have been accomplished.

I would also like to express my gratitude to Dr. Hiroyuki Takeda and Dr. Ryohei Nakamura for the collaboration and support in biological experiments. Thanks for providing precise biological data and insightful comments.

My appreciation also goes to all co-authors of our paper. I would like to thank Mr. Shingo Tomioka for his meticulous comment of assembly around centromere, Mr. Yuta Suzuki for his valuable suggestions of clustering, Dr. Koichiro Doi and Dr. Jun Yoshimura for the gracious help about computational analyses, Dr. Masahiko Kumagai, Dr. Naoki Irie, Dr. Yusuke Inoue and Ms. Yui Uchida for their technical assistance.

I am also grateful to the member of Morishita Laboratory. I would like to offer my thanks to Mr. Yuichi Motai for his Hi-C pipeline, and Dr. Wei Qu for providing medaka gene data.

I owe my gratitude to the developers of the tools used in my research. I am indebted to Dr. Masahiro Kasahara who invented ispcr and Dr. Taro L. Saito who built up UTGB genome browser.

Finally, I would like to show my greatest appreciation to my parents for their sincere support. Without their encouragement this thesis would not have materialized.

References

[1] J. D. Griffith, “Chromatin Structure: Deduced from a Minichromosome,” *Science (80-.).*, vol. 187, no. 4182, pp. 1202–1203, Mar. 1975.

[2] K. Luger, A. W. Mäder, R. K. Richmond, D. F. Sargent, and T. J. Richmond, “Crystal structure of the nucleosome core particle at 2.8 Å resolution,” *Nature*, vol. 389, no. 6648, pp. 251–260, 1997.

[3] R. D. Kornberg and Y. Lorch, “Twenty-five years of the nucleosome, fundamental particle of the eukaryote chromosome,” *Cell*, vol. 98, no. 3, pp. 285–294, 1999.

[4] R. Kornberg, “Chromatin Structure : A Repeating Unit of Histones and DNA Chromatin structure is based on a repeating unit of eight,” *Science (80-.).*, vol. 184, pp. 868–871, 1974.

[5] A. L. Olins and D. E. Olins, “Spheroid Chromatin Units (ngr Bodies),” *Science (80-.).*, vol. 183, no. 4122, pp. 330–332, 1974.

[6] S. C. Elgin, “Heterochromatin and gene regulation in Drosophila,” *Curr. Opin. Genet. Dev.*, vol. 6, no. 2, pp. 193–202, 1996.

[7] D. E. Schones *et al.*, “Dynamic Regulation of Nucleosome Positioning in the Human Genome,” *Cell*, vol. 132, no. 5, pp. 887–898, 2008.

[8] C. Jiang and B. F. Pugh, “Nucleosome positioning and gene regulation: Advances through genomics,” *Nat. Rev. Genet.*, vol. 10, no. 3, pp. 161–172, 2009.

[9] L. Bai, G. Charvin, E. D. Siggia, and F. R. Cross, “Nucleosome-Depleted Regions in Cell-Cycle-Regulated Promoters Ensure Reliable Gene Expression in Every Cell Cycle,” *Dev. Cell*, vol. 18, no. 4, pp. 544–555, Apr. 2010.

[10] T. N. Mavrich *et al.*, “A barrier nucleosome model for statistical positioning of

nucleosome throughout the yeast genome," *Genome Res.*, vol. 18, pp. 1073–1083, 2008.

[11] T. E. Shrader and D. M. Crothers, "Artificial nucleosome positioning sequences (chromatin/histone-DNA binding/DNA bending)," *Biophysics (Oxf).*, vol. 86, no. October, pp. 7418–7422, 1989.

[12] R. K. Chodavarapu *et al.*, "Relationship between nucleosome positioning and DNA methylation," *Nature*, vol. 466, no. 7304, pp. 388–392, Jul. 2010.

[13] Y. Zhang, H. Shin, J. S. Song, Y. Lei, and X. S. Shirley, "Identifying positioned nucleosomes with epigenetic marks in human from ChIP-Seq," *BMC Genomics*, vol. 9, pp. 1–11, 2008.

[14] B. D. Strahl and C. D. Allis, "The language of covalent histone modifications," *Nature*, vol. 403, no. 6765, pp. 41–45, 2000.

[15] T. Jenuwein, "Translating the Histone Code," *Science (80-.).*, vol. 293, no. 5532, pp. 1074–1080, Aug. 2001.

[16] N. D. Heintzman *et al.*, "Histone modification at human enhancers reflect global cell-type specific gene expression," *Nature*, vol. 459, no. 7243, pp. 108–112, 2009.

[17] M. Shogren-Knaak, "Histone H4-K16 Acetylation Controls Chromatin Structure and Protein Interactions," *Science (80-.).*, vol. 311, no. 5762, pp. 844–847, Feb. 2006.

[18] M. Grunstein, "EBSCOhost: Histone acetylation in chromatin structure and transcription.," *Nature*, vol. 389, pp. 349–352, 1997.

[19] T. Kouzarides, "Histone methylation in transcriptional control," *Curr. Opin. Genet. Dev.*, vol. 12, no. 2, pp. 198–209, 2002.

[20] Y. Zhang and D. Reinberg, "Transcription regulation by histone methylation:

Interplay between different covalent modifications of the core histone tails," *Genes Dev.*, vol. 15, no. 18, pp. 2343–2360, 2001.

[21] A. Shilatifard, "Molecular implementation and physiological roles for histone H3 lysine 4 (H3K4) methylation," *Curr. Opin. Cell Biol.*, vol. 20, no. 3, pp. 341–348, 2008.

[22] J. Cheng *et al.*, "A Role for H3K4 Monomethylation in Gene Repression and Partitioning of Chromatin Readers," *Mol. Cell*, vol. 53, no. 6, pp. 979–992, Mar. 2014.

[23] B. E. Bernstein *et al.*, "A Bivalent Chromatin Structure Marks Key Developmental Genes in Embryonic Stem Cells," *Cell*, vol. 125, no. 2, pp. 315–326, 2006.

[24] X. D. Zhao *et al.*, "Whole-Genome Mapping of Histone H3 Lys4 and 27 Trimethylations Reveals Distinct Genomic Compartments in Human Embryonic Stem Cells," *Cell Stem Cell*, vol. 1, no. 3, pp. 286–298, 2007.

[25] R. Nakamura *et al.*, "Large hypomethylated domains serve as strong repressive machinery for key developmental genes in vertebrates," *Development*, vol. 141, no. 13, pp. 2568–2580, 2014.

[26] P. J. Park, "ChIP-seq: Advantages and challenges of a maturing technology," *Nat. Rev. Genet.*, vol. 10, no. 10, pp. 669–680, 2009.

[27] J. J. Infante, G. L. Law, and E. T. Young, *Analysis of nucleosome positioning using a nucleosome-scanning assay*, vol. 833. 2012.

[28] M. J. Levene, J. Korlach, S. W. Turner, M. Foquet, H. G. Craighead, and W. W. Webb, "Zero-mode waveguides for single-molecule analysis at high concentrations," *Science (80-.).*, vol. 299, no. 5607, pp. 682–686, 2003.

[29] J. Korlach *et al.*, "Selective aluminum passivation for targeted immobilization of

single DNA polymerase molecules in zero-mode waveguide nanostructures," *Proc. Natl. Acad. Sci.*, vol. 105, no. 4, pp. 1176–1181, 2008.

[30] J. Eid *et al.*, "Real-time DNA sequencing from single polymerase molecules," *Science (80-.).*, vol. 323, no. 5910, pp. 133–138, 2009.

[31] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[32] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc Natl Acad Sci USA*, vol. 95, no. 25, pp. 14863–14868, 1998.

[33] P. Tamayo *et al.*, "Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation.," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 96, no. 6, pp. 2907–12, 1999.

[34] D. Jiang, C. Tang, and A. Zhang, "Cluster analysis for gene expression data: a survey," *IEEE Trans. Knowl. ...*, vol. 16, no. 11, pp. 1370–1386, 2004.

[35] P. D'Haeseleer, "How does gene expression clustering work?," *Nat. Biotechnol.*, vol. 23, no. 12, pp. 1499–1501, 2005.

[36] T. S. Mikkelsen *et al.*, "Genome-wide maps of chromatin state in pluripotent and lineage-committed cells," *Nature*, vol. 448, no. 7153, pp. 553–560, Aug. 2007.

[37] N. D. Heintzman *et al.*, "Histone modifications at human enhancers reflect global cell-type-specific gene expression," *Nature*, vol. 459, no. 7243, pp. 108–112, 2009.

[38] P. V. Kharchenko *et al.*, "Comprehensive analysis of the chromatin landscape in Drosophila melanogaster," *Nature*, vol. 471, no. 7339, pp. 480–486, 2011.

[39] T. Consortium *et al.*, "Identification of Functional Elements and Regulatory Circuits by Drosophila modENCODE," *October*, vol. 330, no. 6012, pp. 1787–

1797, 2011.

[40] L. Handoko *et al.*, “CTCF-mediated functional chromatin interactome in pluripotent cells,” *Nat. Genet.*, vol. 43, no. 7, pp. 630–638, Jul. 2011.

[41] T. Liu *et al.*, “Broad chromosomal domains of histone modification patterns in C. elegans.pdf,” pp. 227–236, 2011.

[42] J. Ernst and M. Kellis, “ChromHMM: Automating chromatin-state discovery and characterization,” *Nat. Methods*, vol. 9, no. 3, pp. 215–216, 2012.

[43] M. M. Hoffman, O. J. Buske, J. Wang, Z. Weng, J. A. Bilmes, and W. S. Noble, “Unsupervised pattern discovery in human chromatin structure through genomic segmentation,” *Nat. Methods*, vol. 9, no. 5, pp. 473–476, Mar. 2012.

[44] J. R. Dixon *et al.*, “Topological domains in mammalian genomes identified by analysis of chromatin interactions,” *Nature*, vol. 485, no. 7398, pp. 376–380, 2012.

[45] S. M. Johnson, F. J. Tan, H. L. McCullough, D. P. Riordan, and A. Z. Fire, “Flexibility and constraint in the nucleosome core landscape of Caenorhabditis elegans chromatin,” *Genome Res.*, vol. 16, no. 12, pp. 1505–1516, 2006.

[46] W. Lee *et al.*, “A high-resolution atlas of nucleosome occupancy in yeast,” *Nat. Genet.*, vol. 39, no. 10, pp. 1235–1244, 2007.

[47] I. Whitehouse, O. J. Rando, J. Delrow, and T. Tsukiyama, “Chromatin remodelling at promoters suppresses antisense transcription,” *Nature*, vol. 450, no. 7172, pp. 1031–1035, 2007.

[48] A. Valouev *et al.*, “A high-resolution, nucleosome position map of C. elegans reveals a lack of universal sequence-dictated positioning,” *Genome Res.*, vol. 18, no. 7, pp. 1051–1063, 2008.

[49] T. N. Mavrich *et al.*, “Nucleosome organization in the Drosophila genome,”

Nature, vol. 453, no. 7193, pp. 358–362, May 2008.

[50] M. Liu *et al.*, "Determinants of nucleosome positioning and their influence on plant gene expression," *Genome Res*, pp. 1–14, 2015.

[51] X. Wang, G. O. Bryant, M. Floer, D. Spagna, and M. Ptashne, "An effect of DNA sequence on nucleosome occupancy and removal," *Nat. Struct. Mol. Biol.*, vol. 18, no. 4, pp. 507–509, 2011.

[52] a Whereas and A. Tlrs, "A Packing Mechanism for Nucleosome," *Science (80-.).*, vol. 332, no. May, pp. 977–980, 2011.

[53] J. Wang *et al.*, "Sequence features and chromatin structure around the genomic regions bound by 119 human transcription factors Repository Citation Sequence features and chromatin structure around the genomic regions bound by 119 human transcription factors," *Genome Res.*, vol. 9, pp. 1798–1812, 2012.

[54] A. Kundaje *et al.*, "Ubiquitous heterogeneity and asymmetry of the chromatin environment at regulatory elements," *Genome Res.*, vol. 22, no. 9, pp. 1735–1747, 2012.

[55] I. P. Ioshikhes, I. Albert, S. J. Zanton, and B. F. Pugh, "Nucleosome positions predicted through comparative genomics," *Nat. Genet.*, vol. 38, no. 10, pp. 1210–1215, 2006.

[56] K. Ichikawa and S. Morishita, "A simple but powerful heuristic method for accelerating k-means clustering of large-scale data in life science," *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, vol. 5963, no. c, pp. 1–1, 2014.

[57] M. E. Celebi, H. A. Kingravi, and P. A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert Syst. Appl.*, vol. 40, no. 1, pp. 200–210, 2013.

[58] E. W. Forgy, "Cluster analysis of multivariate data: Efficiency versus

interpretability of classifications," Biometrics, vol. 21, pp. 768–769, 1965.

[59] J. Macqueen, "Some methods for classification and analysis of multivariate observations," *Proc. Fifth Berkeley Symp. Math. Stat. Probab.*, vol. 1, no. 233, pp. 281–297, 1967.

[60] T. F. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theor. Comput. Sci.*, vol. 38, no. C, pp. 293–306, 1985.

[61] I. Katsavounidis, C. C. J. Kuo, and Z. Zhang, "A New Initialization Technique for Generalized Lloyd Iteration," *IEEE Signal Process. Lett.*, vol. 1, no. 10, pp. 144–146, 1994.

[62] P. S. Bradley and P. S. Bradley, "Refining Initial Points for K-Means Clustering," *Microsoft Res.*, pp. 91–99, 1998.

[63] D. Arthur and S. Vassilvitskii, "K-Means++: the Advantages of Careful Seeding," *Proc. eighteenth Annu. ACM-SIAM Symp. Discret. algorithms*, vol. 8, pp. 1027–1025, 2007.

[64] T. Su and J. Dy, "In search of deterministic methods for initializing K-means and Gaussian mixture clustering," *Intell. Data Anal.*, vol. 11, pp. 1–42, 2007.

[65] S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[66] F. D. Gibbons and F. P. Roth, "Judging the quality of gene expression-based clustering methods using gene annotation," *Genome Res.*, vol. 12, no. 10, pp. 1574–1581, 2002.

[67] F. Geraci, M. Leoncini, M. Montangero, M. Pellegrini, and M. E. Renda, "K-Boost: A Scalable Algorithm for High-Quality Clustering of Microarray Gene Expression Data," *J. Comput. Biol.*, vol. 16, no. 6, pp. 859–873, Jun. 2009.

[68] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church,

"Systematic determination of genetic network architecture [see comments]," *Nat Genet*, vol. 22, no. 3, pp. 281–285, 1999.

[69] R. Sharan and R. Shamir, "CLICK: a clustering algorithm with applications to gene expression analysis.," *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, vol. 8, pp. 307–16, 2000.

[70] F. De Smet, G. Thijs, K. Marchal, B. De Moor, and Y. Moreau, "Adaptive quality-based clustering of gene expression profiles," *Bioinformatics*, vol. 18, pp. 735–746, 2002.

[71] K. L. Clarkson, "Nearest-neighbor searching and metric space dimensions," *Nearest-Neighbor Methods Learn. Vis. Theory Pract.*, no. April, pp. 15–59, 2006.

[72] C. Elkan, "Using the Triangle Inequality to Accelerate k-Means," *Proc. Twent. Int. Conf. Mach. Learn.*, pp. 147–153, 2003.

[73] G. Hamerly, "Making k -means even faster," *2010 SIAM Int. Conf. data Min. (SDM 2010)*, pp. 130–140, 2010.

[74] J. Drake and G. Hamerly, "Accelerated k-means with adaptive distance bounds," *5th NIPS Work. Optim. Mach. Learn. OPT2012*, pp. 2–5, 2012.

[75] V. C. Osamor, E. F. Adebiyi, J. O. Oyelade, and S. Doumbia, "Reducing the Time Requirement of k-Means Algorithm," *PLoS One*, vol. 7, no. 12, 2012.

[76] M. H. Fulekar, *Bioinformatics: Applications in Life and Environmental Science.* New York, NY, USA: Springer, 2009.

[77] M. Matsumoto and T. Nishimura, "Mersenne Twister : A 623-dimensionally equidistributed uniform pseudorandom number generator," *Discrete Math.*, 1998.

[78] J. Harrow, A. Frankish, J. M. Gonzalez, E. Tapanari, M. Diekhans, and F.

Kokocinski, “GENCODE: The Reference Human Genome Annotation for The ENCODE Project,” *Genome Res*, vol. 22, pp. 1760–1774, 2012.

[79] L. Sun *et al.*, “Neuronal and glioma-derived stem cell factor induces angiogenesis within the brain,” *Cancer Cell*, vol. 9, no. 4, pp. 287–300, 2006.

[80] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[81] K. Ichikawa and S. Morishita, “A linear time algorithm for detecting long genomic regions enriched with a specific combination of epigenetic states,” *BMC Genomics*, vol. 16, no. Suppl 2, p. S8, 2015.

[82] B. Hendrich and S. Tweedie, “The methyl-CpG binding domain and the evolving role of DNA methylation in animals,” *Trends Genet.*, vol. 19, no. 5, pp. 269–277, 2003.

[83] A. Bird, “DNA methylation patterns and epigenetic memory,” *Genes Dev*, vol. 16, pp. 6–21, 2002.

[84] N. L. Vastenhouw and A. F. Schier, “Bivalent histone modifications in early embryogenesis,” *Curr. Opin. Cell Biol.*, vol. 24, no. 3, pp. 374–386, Jun. 2012.

[85] V. W. Zhou, A. Goren, and B. E. Bernstein, “Charting histone modifications and the functional organization of mammalian genomes,” *Nat. Rev. Genet.*, vol. 12, no. 1, pp. 7–18, 2011.

[86] W. Xie *et al.*, “Epigenomic Analysis of Multilineage Differentiation of Human Embryonic Stem Cells,” *Cell*, vol. 153, no. 5, pp. 1134–1148, May 2013.

[87] M. Jeong *et al.*, “Large conserved domains of low DNA methylation maintained by Dnmt3a,” *Nat. Genet.*, vol. 46, no. 1, pp. 17–23, Jan. 2014.

[88] O. Bogdanović *et al.*, “Temporal uncoupling of the DNA methylome and

transcriptional repression during embryogenesis," *Genome Res.*, vol. 21, no. 8, pp. 1313–1327, 2011.

[89] J.-L. Hu, B. O. Zhou, R.-R. Zhang, K.-L. Zhang, J.-Q. Zhou, and G.-L. Xu, "The N-terminus of histone H3 is required for de novo DNA methylation in chromatin," *Proc. Natl. Acad. Sci.*, vol. 106, no. 52, pp. 22187–22192, 2009.

[90] H. Cedar and Y. Bergman, "Linking DNA methylation and histone modification: Patterns and paradigms," *Nat. Rev. Genet.*, vol. 10, no. 5, pp. 295–304, 2009.

[91] S. K. T. Ooi *et al.*, "DNMT3L connects unmethylated lysine 4 of histone H3 to denovo methylation of DNA," *Nature*, vol. 448, no. 7154, pp. 714–717, 2009.

[92] H. K. Long *et al.*, "Epigenetic conservation at gene regulatory elements revealed by non-methylated DNA profiling in seven vertebrates," *Elife*, vol. 2013, no. 2, pp. 1–19, 2013.

[93] W. Qu *et al.*, "Genome-wide genetic variations are highly correlated with proximal DNA methylation patterns Genome-wide genetic variations are highly correlated with proximal DNA methylation patterns," *Genome Res.*, vol. 22, no. 8, pp. 1419–1425, 2012.

[94] R. Lister *et al.*, "HumanDNAmethylomesatbaseresolutionshowwidespreadepigenomicdifferences," *Nature*, vol. 462, no. 7271, pp. 315–322, 2009.

[95] M. Csurös, "Maximum-scoring segment sets," *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, vol. 1, no. 4, pp. 139–150, 2004.

[96] A. Valouev *et al.*, "Genome-wide analysis of transcription factor binding sites based on ChIP-Seq data," *Nat. Methods*, vol. 5, no. 9, pp. 829–834, 2008.

[97] K. Ichikawa *et al.*, "Centromere evolution and CpG methylation during vertebrate speciation," *Nat. Commun.*, vol. 8, no. 1, 2017.

[98] H. Takeda and A. Shimada, “The Art of Medaka Genetics and Genomics: What Makes Them So Unique?,” *Annu. Rev. Genet.*, vol. 44, no. 1, pp. 217–241, 2010.

[99] Naruse, K., Tanaka, M. & Takeda, H. in *Medaka: A Model for Organogenesis, Human Disease, and Evolution*, Springer, 2011.

[100] D. H. E. Setiamarga *et al.*, “Divergence time of the two regional medaka populations in Japan as a new time scale for comparative genomics of vertebrates,” *Biol. Lett.*, vol. 5, no. 6, pp. 812–816, 2009.

[101] Y. Takehana, N. Nagai, M. Matsuda, K. Tsuchiya, and M. Sakaizumi, “Geographic variation and diversity of the cytochrome b gene in Japanese wild populations of medaka, Oryzias latipes.,” *Zool. Sci*, vol. 20, no. 10, pp. 1279–1291, 2003.

[102] M. Spivakov *et al.*, “Genomic and Phenotypic Characterization of a Wild Medaka Population: Towards the Establishment of an Isogenic Population Genetic Resource in Fish,” *G3: Genes|Genomes|Genetics*, vol. 4, no. 3, pp. 433–445, 2014.

[103] T. Asai, H. Senou, and K. Hosoya, “Oryzias sakaizumii, a new ricefish from northern Japan (Teleostei: Adrianichthyidae),” *Ichthyol. Explor. Freshwaters*, vol. 22, no. 4, pp. 289–299, 2011.

[104] M. Kasahara *et al.*, “The medaka draft genome and insights into vertebrate genome evolution,” *Nature*, vol. 447, no. 7145, pp. 714–719, 2007.

[105] R. Nakamura, A. Uno, M. Kumagai, S. Morishita, and H. Takeda, “Hypomethylated domain-enriched DNA motifs prepattern the accessible nucleosome organization in teleosts,” *Epigenetics and Chromatin*, vol. 10, no. 1, pp. 1–13, 2017.

[106] K. S. Bloom and J. Carbon, “Yeast centromere DNA is in a unique and highly

ordered structure in chromosomes and small circular minichromosomes," *Cell*, vol. 29, no. 2, pp. 305–317, 1982.

[107] M. Fitzgerald-Hayes, L. Clarke, and J. Carbon, "Nucleotide sequence comparisons and functional analysis of yeast centromere DNAs," *Cell*, vol. 29, no. 1, pp. 235–244, 1982.

[108] D. R. Foltz, L. E. T. Jansen, B. E. Black, A. O. Bailey, J. R. Yates, and D. W. Cleveland, "The human CENP-A centromeric nucleosome-associated complex," *Nat. Cell Biol.*, vol. 8, no. 5, pp. 458–469, 2006.

[109] P. E. Warburton, "Chromosomal dynamics of human neocentromere formation," *Chromosom. Res.*, vol. 12, no. 6, pp. 617–626, 2004.

[110] E. Lieberman-Aiden and N. van Berkum, "Comprehensive mapping of long range interactions reveals folding principles of the human genome," *Science (80-.).*, vol. 326, no. 5950, pp. 289–293, 2009.

[111] A. Koga, M. Suzuki, H. Inagaki, Y. Bessho, and H. Hori, "Transposable element in fish," *Nature*, vol. 383, no. 6595. p. 30, 1996.

[112] M. Matsuda *et al.*, "DMY is a Y-specific DM-domain gene required for male development in the medaka fish," *Nature*, vol. 417, no. 6888, pp. 559–563, 2002.

[113] M. Kondo *et al.*, "Genomic organization of the sex-determining and adjacent regions of the sex chromosomes of medaka," *Genome Res.*, vol. 16, no. 7, pp. 815–826, 2006.

[114] T. Kimura *et al.*, "Genetic linkage map of medaka with polymerase chain reaction length polymorphisms," *Gene*, vol. 363, no. 1–2, pp. 24–31, 2005.

[115] M. Pendleton *et al.*, "Assembly and diploid architecture of an individual human genome via single-molecule technologies," *Nat. Methods*, vol. 12, no. 8, pp.

780–786, Aug. 2015.

[116] C. S. Chin *et al.*, "Nonhybrid, finished microbial genome assemblies from long-read SMRT sequencing data," *Nat. Methods*, vol. 10, no. 6, pp. 563–569, 2013.

[117] B. J. Walker *et al.*, "Pilon: An integrated tool for comprehensive microbial variant detection and genome assembly improvement," *PLoS One*, vol. 9, no. 11, 2014.

[118] S. S. P. Rao *et al.*, "A 3D Map of the Human Genome at Kilobase Resolution Reveals Principles of Chromatin Looping," *Cell*, vol. 159, no. 7, pp. 1665–1680, Dec. 2014.