Identification of Regulatory Binding Sites Using Minimum Spanning Trees

V. Olman, D. Xu, Y. Xu

Pacific Symposium on Biocomputing 8:327-338(2003)

IDENTIFICATION OF REGULATORY BINDING SITES USING MINIMUM SPANNING TREES

VICTOR OLMAN, DONG XU, YING XU*
Protein Informatics Group, Life Sciences Division
Oak Ridge National Laboratory, Oak Ridge, TN 37831-6480, USA.
* *To whom correspondence should be addressed. Email: xyn@ornl.gov.*

Recognition of protein-binding sites from the upstream regions of genes is a highly important and unsolved problem. In this paper, we present a new approach for studying this challenging issue. We formulate the binding-site recognition problem as a *cluster identification problem*, i.e., to identify clusters in a data set that exhibit significantly different features (e.g., density) than the overall background of the data set. We have developed a general framework for solving such a cluster identification problem. The foundation of the framework is a rigorous relationship between data clusters and subtrees of a minimum spanning tree (MST) representation of a data set. We have proposed a formal and general definition of clusters, and have demonstrated that a cluster is always represented as a connected component of the MST, and further it corresponds to a substring of a linear representation of the MST. Hence a cluster identification problem is reduced to a problem of finding substrings with certain features, for which algorithms have been developed. We have applied this MST-based cluster identification algorithm to a number of binding site identification problems. The results are highly encouraging.

1 Introduction

One of major challenging problems in systems biology is to understand the mechanisms governing the regulation of gene transcriptions. Microarray gene expression chips allow researchers to directly observe the dynamics of transcriptions of many genes simultaneously. A gene's transcriptional level is regulated by proteins (transcription factors), which bind to specific sites in the gene's promoter region, called *binding sites* [1]. Identification of genes that share common protein-binding sites can provide highly useful constraints in modeling of gene-transcriptional machineries [2].

Typically, a protein-binding site is a short (contiguous) fragment located in the upstream region of a gene. The binding sites by the same protein for different genes may not be exactly the same, rather they are similar on the sequence level. Computationally, the binding-site identification problem is often defined as to find short "conserved" fragments, from a set of genomic sequences, which cover many (or all) of the provided genomic sequences.

Because of the significance of this problem, many computer algorithms have been proposed to solve the problem [1,3]. Among the popular computer software for this problem are CONSENSUS [4] and MEME [5]. The basic idea

among many of these algorithms/systems is to find a subset of short fragments from the provided genomic sequences, which show "high" information content [1] in their gapless multiple sequence alignments[a]. While many good algorithms have been proposed, this highly challenging problem remains not fully solved.

We have recently developed a new approach for the binding-site identification problem. Different than the previous methods, we have treated this problem as a clustering problem. Conceptually, we map all the fragments, collected from the provided genomic sequences, into a space so that similar fragments (on the sequence level) are mapped to nearby positions and dissimilar fragments to far away positions. Because of the relatively high frequency of the conserved binding sites appearing in the targeted genomic sequence regions, a group of such sites should form a "dense" cluster in a sparsely-distributed background. So the computational problem becomes to identify and extract such clusters from a "noisy" background. It is worth mentioning that this problem is different from classical clustering, which partitions all the elements of a data set into clusters.

Here we present a new framework for solving such a cluster identification problem. The foundation of this framework is a representation of a data set as a minimum spanning tree. We have demonstrated that no essential information is lost for the purpose of clustering with this representation. The simplicity of the minimum spanning tree structure allows us to deal with the clustering problem in a rigorous and efficient way. The main contribution of this work can be summarized as follows. We proposed a rigorous definition of a cluster, which we believe captures the essence of the concept of *clusters* that people frequently use but without a formal definition. This definition allows us to establish rigorous relationships between data clusters and subtrees of a minimum spanning tree, which further allow us to deal with the binding-site identification problem in a rigorous manner. Preliminary application results to three binding sites are highly encouraging. Though minimum spanning trees (MST) have long been used for data classification and clustering [6,7], we have not seen the kind of in-depth study of minimum spanning trees *versus* data clustering as what we present in this paper.

2 Minimum Spanning Trees *versus* Clusters

Let $D = \{d_i\}$ be a data set. We define a weighted (undirected) graph $G(D) = (V, E)$ as follows. The vertex set $V = \{d_i | d_i \in D\}$ and the edge set $E =$

[a] Very few closely related binding sites differ by an insertion/deletion due to the specific binding configuration of the transcrition factor on the DNA sequence.

$\{(d_i, d_j)|$ for $d_i, d_j \in D$ and $i \neq j\}$. Each edge $(u, v) \in E$ has a distance[b], $\rho(u, v)$, between u and v of V. Here *distance* is a binary relationship, which does not have to be a *metric*. A *spanning tree* T of a (connected) weighted graph $G(D)$ is a connected subgraph of $G(D)$ such that (i) T contains every vertex of $G(D)$, and (ii) T does not contain any cycle. A *minimum spanning tree* (MST) is a spanning tree with the minimum total distance. In this paper, any connected component of a MST is called a *subtree* of the MST.

Prim's algorithm represents one of the classical methods for solving the minimum spanning tree problem [8]. The basic idea of the algorithm can be outlined as follows: *the initial solution is a singleton set containing an arbitrary vertex; the current partial solution is repeatedly expanded by adding the vertex (not in the current solution) that has the shortest edge to a vertex in the current solution, along with the edge, until all vertices are in the current solution.* A simple implementation of Prim's algorithm runs in $O(\|E\| \log(\|V\|))$ time [9], where $\|\cdot\|$ represents the number of elements in a set.

Our first goal is to establish a rigorous relationship between a minimum spanning tree representation of a data set and clusters in the data set. To do this, we need to a formal definition of a cluster.

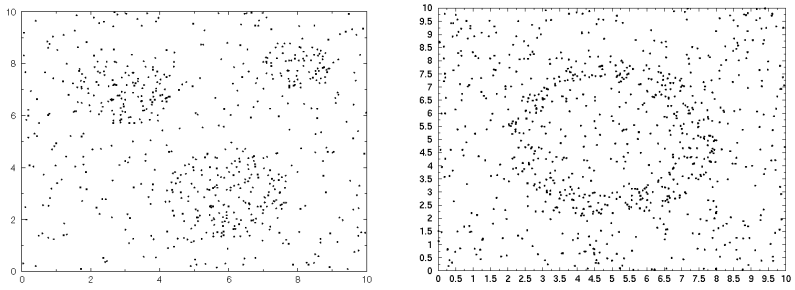
Definition 1: Let D be a data set and $\rho(u, v)$ denote the distance between any pair of data u, v in D. The **necessary condition** for any $C \subseteq D$ to be a cluster is that for any non-empty partition $C = C_1 \cup C_2$, the closest data point $d \in D - C_1$ to C_1 (measured by ρ) must be from C_2. $\square$

Note that the distance between a point to a data set means the distance between the point and its closest data point in the set. Here we provide only the necessary condition of a cluster since we believe that the sufficient condition of a cluster ought to be problem-dependent. We believe that this definition captures the essence of our intuition about a cluster: that is *distances among neighbors within a cluster should be smaller than any inter-cluster distances.*

Theorem 1: For any data set D and a distance measure ρ defined on D, let T be a MST representing D with its edge distances defined by ρ. If $C \subseteq D$ is a cluster by **Definition 1**, then C's data points form a subtree of T.

The proof of Theorem 1 can be found in our previous paper [10]. This theorem implies that a clustering problem can be rigorously reduced to a tree partitioning problem since each cluster is represented as a subtree of the MST representation of the data set. Note that a MST may not be unique for a given graph, and the non-uniqueness of MSTs does not affect the correctness of this theorem.

[b] For the simplicity of proofs and discussions, we assume that no two edges have the same distance. The proofs stay correct when edges are allowed to have the same distance.



(a) (b)

Figure 1: Examples of clusters in the an approximately uniformly-distributed background. (a) Three clusters with higher density than the background. (b) A ring-shaped cluster with higher density than the background.

3 Minimum Spanning Trees *versus* Cluster Identification

A *clustering problem* is typically defined as to partition a given data set into K clusters to optimize some objective function, that generally requires that the data points of the same cluster are "similar" and data points of different clusters are "dissimilar", for some $K > 1$. We have recently developed a set of MST-based clustering algorithms for solving this type of clustering problem [10,11]. A basic assumption for a clustering problem is that data points can be divided into clusters. However in real applications, it is often the case that data points of well-defined clusters may not appear in a vacuum but rather appear in a more general background. Figure 1 shows two such examples. Now we define a more general clustering problem. A *cluster identification problem* is defined as to partition a data set D as $D = B \cup D_1 \cup ... \cup D_p$ such that B is a data set that is approximately uniformly distributed, and each D_i forms a cluster in D, $i \in [1, p]$. A data set D is *uniformly distributed* in a bounded space G if for any region $A \subseteq G$, the number of data points of D in A is approximately proportional to the volume of A. We found that many biological data analysis problems can be modeled as cluster identification problems.

Before we present our algorithm for solving the cluster identification problem, we give an equivalent definition of a cluster. We continue to use D and ρ to denote a data set and the distance measure defined on D. For any $C \subseteq D$, we define an augmentation operation $\mathcal{A}(C) = C \cup \{c^*\}$, where $c^* \in D - C$ is the closest such element to C. We define $\mathcal{A}(D) = D$, and $\mathcal{A}^k = \mathcal{A}(\mathcal{A}^{k-1})$.
Definition 2: The **necessary condition** for a $C \subseteq D$ forms a cluster if for any $c \in C$, $\mathcal{A}^{\|C\|-1}(c) = C$. □

This cluster definition is more closely related to the Prim's algorithm, which allows us to prove some of our results more easily.

Theorem 2: Definitions 1 and **2** are equivalent. That is, if $C \subseteq D$ is a cluster under **Definition 1**, then it is a cluster under **Definition 2**; and vise versa.

Proof: It is straightforward to show that **Definition 1** implies **Definition 2**. Hence we omit the proof. We now prove that **Definition 2** also implies **Definition 1**. It is not difficult to see that all we need to prove is that if C is a cluster under **Definition 2**, then for any $B \subseteq C$, $\mathcal{A}^{\|C\|-\|B\|}(B) = C$.

Let's assume that this is not true, and let $B^* \subset C$ be the subset that does not satisfy this equation. So there exists a $d_0 \in \mathcal{A}^{\|C\|-\|B^*\|}(B^*) \cap (D-C)$. We assume, without loss of generality, $d_0 \in \mathcal{A}(B^*)$ (if not, we can always expand B^*). Hence there exists a $b_0 \in B^*$ such that

$$\rho(b_0, d_0) < \min_{b \in B^*, c \in C - B^*} \rho(b, c). \tag{1}$$

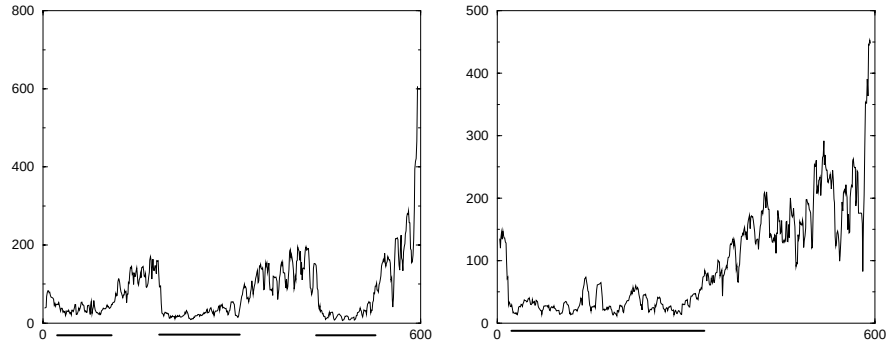
Since C is a cluster under **Definition 2** and $b_0 \in C$, we know $\mathcal{A}^{\|C\|-1}(b_0) = C$. However this contradicts the inequality (1) since inequality (1) implies that d_0 will be added to $\mathcal{A}^{\|C\|-1}(b_0)$ before any other elements of $C - B^*$ can be added. However we know $d_0 \notin \mathcal{A}^{\|C\|-1}(b_0)$. This contradiction implies the correctness of the theorem. $\square$.

Clearly, $\mathcal{A}$ represents the basic selection operation in the Prim's algorithm. Let $(d_1, ..., d_{\|D\|})$ be the list of elements selected (in this order) by the Prim's algorithm when constructing a MST of the data set D and starting from element $d_1 \in D$. This list maps the data set D (of any dimensions) to a string $d_1....d_{\|D\|}$, which we call a *linear representation* of D.

Theorem 3: For any linear representation $L(D)$ of a data set D, any cluster C of D is represented as one substring of $L(D)$.

Proof: We need to show that after the Prim's algorithm selects the first element c_0 of C, it will not select any element outside of C until it has selected every element of C. We assume that this is not the case. Let d_0 be the first element of $D-C$ that gets selected after c_0's selection. We use C' to denote the subset of C that have been selected when d_0 is selected. By our assumption, $C' \subset C$. Let (c_0, d) be the MST edge that gets c_0 selected; similarly (d_0, d') be the corresponding edge of d_0 (note that $d' \in D-C$ by **Definition 2**; otherwise it would contradict the definition). Also let (c_1, c_2) be the first edge selected after d_0's selection, with $c_1 \in C'$ and $c_2 \in C - C'$ (note that such (c_1, c_2) must exist since elements of C form a subtree of the MST). By the order of their selections, we have

$$\rho(c_1, c_2) > \rho(d_0, d') > \rho(c_0, d).$$



(a) (b)

Figure 2: Plots of edge-distances in the order of their selection by the Prim's algorithm. Each valley is underlined. (a) Edge-distance plot of Figure 1 (a). (b) Edge-distance plot of Figure 1 (b).

This apparently contradicts the fact that $\rho(c_1, c_2) < \rho(c_0, d)$, implied by the proof of Theorem 2. So we have proved the theorem. □

A direct application of Theorem 3 is that we can plot the edge distances in the selection order by the Prim's algorithm. In this plot, the x-axis is a linear representation $L(D)$ of data set D, and the y-axis represents the distance of the corresponding MST edge. By Theorem 3 and the definition of a cluster, each cluster should form a "valley" in this plot. Figure 2 shows two examples of such plot. In Figure 2(a), the three (visibly apparent) clusters of Figure 1 (a) are nicely represented by three valleys. Similarly, the one ring-shaped cluster of Figure 1 (b) is well represented as a valley in Figure 2(b). This suggests that we can find clusters from a noisy background through searching a linear string and finding the substrings with certain properties. The following result forms the foundation of our search algorithm. For a substring S of a linear representation $L(D)$ of a data set D, the *left-boundary edge* of S is defined to be the corresponding MST edge when the leftmost element of S was added into the MST. The *right-boundary edge* of S is defined to be the first MST edge, linked with the rightmost element of S, that gets selected after the rightmost element is selected. If the leftmost element of S is a first element of $L(D)$, the distance of the left boundary edge of S is infinity; similarly if the rightmost element of S is a last element of $L(D)$, the distance of the right boundary edge of S is infinity.

Theorem 4: A subtring S of $L(D)$ represents a cluster if and only if (a) S's elements form a subtree, T_S, of D's MST, and (b) S's both boundary edges

have larger distances than any edge-distance of T_S.

Proof: Apparently, Theorem 3 implies the *only-if* condition. The *if* condition can be proved easily as follows. Since the right-boundary edge of S is larger than all tree edges of T_S, we know all other edges connecting S with $D - S$ are larger than the tree edges of T_S, possibly except for the left-boundary edge (because the right-boundary is the first selected such edge after the selection of the left-boundary edge). Since we also know that the left-boundary edge is also larger than T_S's edges, we conclude that S forms a cluster by the proof of Theorem 2. □

We now present an algorithm for finding clusters in a noisy data set D. The algorithm goes through all substrings of $L(D)$ and checks if the (a) and (b) conditions of Theorem 4 are satisfied. We use K to represent the smallest cluster size we care to identify.

Procedure ClusterIdentification(D, K)

Construct a MST, T_D, of data set D, using Prim's algorithm;
Generate a linear representation $L(D)$ of D;

FOR $i = 1$; **UNTIL** $i = \|D\| - K$ **DO**
  FOR $j = i + K$; **UNTIL** $j = \|D\|$ **DO**
    IF (elements of $L(D)[i, j]$ forms a subtree of T_D **AND**
      left- and right-boudanry edges of $L(D)[i, j]$ are larger than edges of $T_{L(D)[i,j]}$)
    THEN output $L(D)[i, j]$ as a cluster.

Checking if a subset of vertices of a tree forms a subtree of the tree can be done in linear time of the number of vertices [12]. So this algorithm takes $O(\|D\|^3)$ time to locate all clusters. Improved computing time may be possible using advanced data structures, but the current running time is adequate for our applications. **Theorem 4 implies that the ClusterIdentification procedure finds all the clusters (defined by our definition) in the data set and finds clusters only.**

To make the discussion on clusters complete, we have the following result regarding the structure among all clusters in a data set. The proof is straightforward.

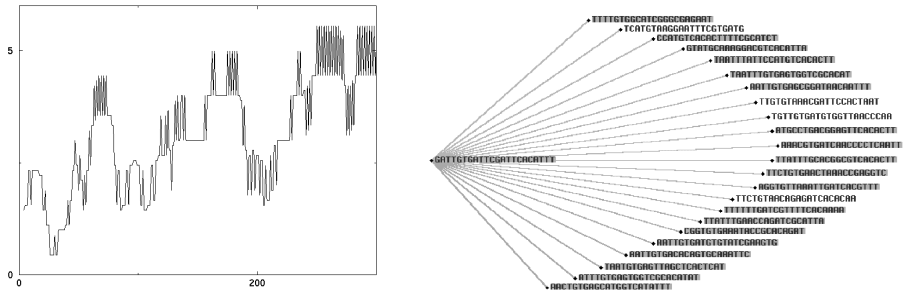
Corollary: For any data set D and its two clusters A and B, if $A \cap B \neq \emptyset$, then $A \subseteq B$ or $B \subseteq A$. □

4 Applications

For a given set of upstream regions of genes, possibly collected through finding genes having correlated expression profiles, our procedure finds the conserved short fragments, say with k bases, as follows. First, it breaks the genomic sequences into k-mers (if k is not known we will go through all k's within some range provided by the user), denoted as a set S. For two k-mers $A = a_1...a_k, B = b_1...b_k \in \mathcal{S}$, we define their distance $\rho(A, B) = \sum_1^k \sigma_i M(a_i, b_i)$, where $M(x, y) = 0$ if $x = y$ otherwise 1. Initially, all σ_i is set to $1/K$, where K is the number of sequences containing at least one of the k-mers A or B. Then we apply the **ClusterIdentification** Procedure to identify all clusters, using ρ as the distance measure. As we discussed earlier, this procedure identifies subsets of S that satisfy the necessary condition of a cluster (for a particular distance measure) while the sufficient condition is problem specific. For the binding site identification problem, the problem-specific conditions should include the following: (1) the position-specific information content [13] of the gapless multiple-sequence alignment, among all the sequence fragments represented by a cluster, should be relatively high; (2) elements of an identified cluster should not be among long, simple repeats; and (3) the data density within a cluster should be relatively higher than the one of the overall background.

To incorporate the information-content constraint into our binding-site identification procedure, we will do the following. After a cluster is identified using the procedure of **ClusterIdentification**, we will measure the position-specific information content. If the overall information content is lower than some threshold, we will discard this cluster for further consideration. Otherwise, we will modify our original distance measure $\rho()$ by the calculated information content as follows. For each position i, we use its information content as σ_i in the next iteration of applying **ClusterIdentification**; and set $M(a_i, b_i) = 2 - (p_i(a_i) + p_i(b_i)) + |p_i(a_i) - p_i(b_i)|$, where $p_i(x)$ represents the frequency of letter x among all letters in position i. We will iterate this procedure for a fixed number of iterations. Our observation has been that for a real binding-site cluster, the procedure converges quickly. Otherwise it diverges.

To "guarantee" some level of uniformity within an identified cluster, we examine the edge-distance distribution of the subtree (of the MST) representing the cluster. If data points are approximately "uniformly" distributed, the edge-distance distribution should be generally unimodular; a multi-modular distribution typically indicates that there are regions within the cluster, which have different density levels. Our procedure discards clusters with non-unimodular



(a) (b)

Figure 3: Identification of CRP binding sites. (a) Edge-distance plot. The deepest valley in the plot corresponds precisely to the subtree of (b). (b) The (sub)tree structure of identified CRP binding sites. The shaded fragments represent known CRP sites. The edge-distance is color-coded, ranging from green to yellow, with green being the shortest.

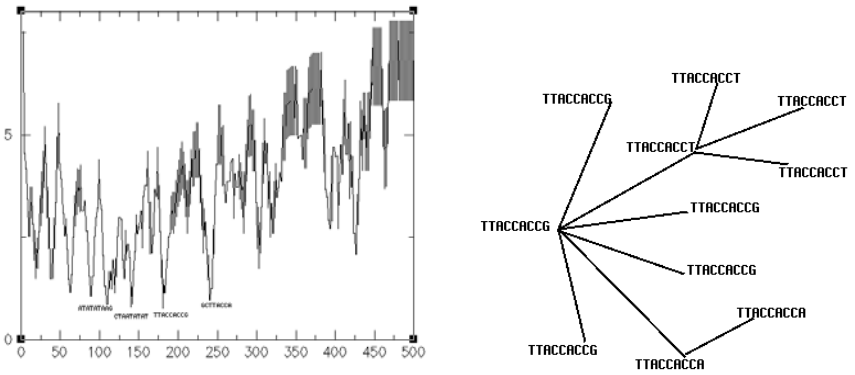
distributions from further consideration.

Our implemented program has a number of parameters, e.g., the minimum cluster size. The values of these parameters are selected, using a simple search program to find an "optimal" performance on a set of genomic sequences with known regulatory binding sites. We have applied this program to a number of binding-site identification problems. We present three case studies here. The computing time for each case took a few minutes on a Unix workstation.

CRP binding sites: This is a widely used testing set from *E. Coli* for validating a method of binding-site identification [1]. The test set consists of 18 sequences (each of length 105 bps) with 23 experimentally verified CRP binding sites (22-mers). Our iterative procedure stopped after two iterations. The only cluster identified is the one shown in Figure 3. The cluster consists of 24 fragments, of which 20 are known CRP sites (out of 23) and the remaining four may or may not be true CRP sites. This result is at least as good as results reported previously by other approaches [1,14].

Yeast binding site: The second application example is for the nucleosome complex proteins promoter binding sites in yeast [15]. There are 8 regulatory sequences, each containing 1000 bp. By using 9-mers, our method identified several clusters (see Fig. 4 (a)). The most populated cluster is TTACCACCG, which also connects several other clusters with similar motifs on the MST (see Fig. 4 (b)). The cluster TTACCACCG and its connected clusters have high information content and they appear in all 8 regulatory sequences. It turned out that TTACCACCG is an experimentally verified motif [15].

Human binding sites: We have applied our method to a set of human



(a) (b)

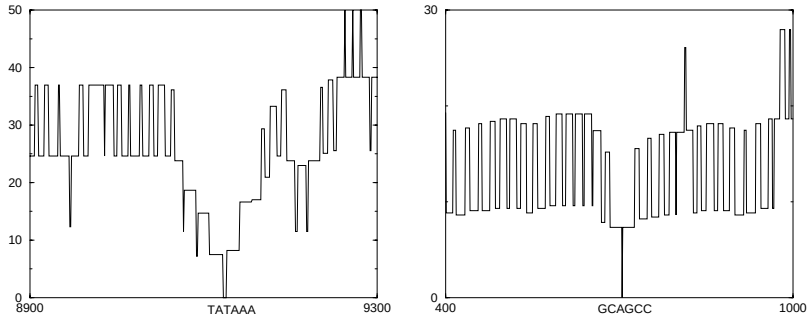
Figure 4: Binding site identification for the yeast data set. (a) Edge-distance plot. (b) The subtree structure of an identified binding site TTACCACCG.

genomic sequences containing binding sites, collected in [16]. The data set has 113 regulatory sequences containing regulatory regions. Each sequence is 300 bp long, with 250 bp upstream and 50 bp downstream of the transcriptional start site. It is known that the TATA box is one of the binding sites. It is also believed that other protein-binding sites may exist in the data set, although no experimental data is available to support this yet. One of the best identification methods for binding sites [17] identified the 6-mer TATAAA as the only identifiable binding motif consensus. Our method has easily identified this motif (see Fig. 5 (a)). In addition, it has identified a number of other possible binding sites, including GCAGCC as shown in Fig. 5 (b). The GCAGCC motif with at most one mis-match appears in 96 regulatory sequences, even more frequently than the TATAAA motif, where appears in 66 regulatory sequences with at most one mis-match. The information content of the GCAGCC motif is also higher:

```
TATAAA: 0.819569 0.793079 0.861977 0.869478 0.802716 0.821651

GCAGCC: 0.755810 0.798584 0.692777 0.906340 0.823345 0.849254
```

where the information content at each position is normalized to the range of 0 to 1. Based on this, we believe that our method is more sensitive in detecting conserved sites.



(a) (b)

Figure 5: Edge-distance plot for binding site identification for the human data set. (a) Around the cluster containing the TATAAA binding site. (b) Around the cluster containing the GCAGCC binding site.

5 Discussion

Based on a formal definition of *clusters*, we have developed a rigorous framework for identifying and extracting clusters from a noisy background. As we believe that many data analysis problems can be formulated as a cluster identification problem or its variant, we expect that this framework will find many interesting applications. The linear representation of a data set allows us to "directly" visualize the cluster structures even for high dimensional data sets. Having such a visualization capability should clearly improve our confidence in our clustering results, since we can "see" the clusters.

Based on the examples we have tested so far, our method clearly shows some advantages over the existing methods. The first advantage is that our method can take advantage of the fact that a regulatory sequence contains the same motif at the different locations, and it will increase the population of the cluster containing the binding site. Another advantage of our method is its sensitivity. Our method is based on a combinatorial approach, which can identify all clusters of possible binding sites. Existing methods generally use sampling techniques, which are likely to miss some binding sites that do not have very strong patterns, as shown in the example of human binding site.

A method is currently being developed for assessing the statistical significance of each identified cluster based on the "depth" and the "width" of the valley representing the cluster in the edge-distance plot (unpublished results).

Acknowledgements

The authors thank Dr. Gary Stormo for providing us the CRP binding site data. The work was supported by ORNL LDRD funding and by the Office of Biological and Environmental Research, U.S. Department of Energy, under Contract DE-AC05-00OR22725, managed by UT-Battelle, LLC.

1. G. D. Stormo and G. W. Hartzell 3rd. *Proc. Natl. Acad. Sci. USA*, 86:1183–1187, 1989.
2. Y. V. Kondrakhin, et al. *Comput. Appl. Biosci.*, 11:477–488, 1995.
3. R. Staden. *Comput. Appl. Biosci.*, 89:293–298, 1989.
4. G. Z. Hertz and G. D. Stormo. *Bioinformatics*, 15:563–577, 1999.
5. T. L. Bailey and M. Gribskov. *J. Comput. Biol.*, 5:211–221, 1998.
6. J. C. Gower and G. J. S. Ross. *Applied Statistics*, 18:54–64, 1969.
7. R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, New York, 1973.
8. R. C. Prim. *Bell System Technical Journal*, 36:1389–1401, 1957.
9. T. H. Cormen, C. E. Leiserson, and R. L. Rivet. *Introduction to Algorithms*. MIT Press, Cambridge, 1989.
10. Y. Xu, V. Olman, and D. Xu. *Bioinformatics*, 18:536-545, 2002.
11. Y. Xu, V. Olman, and D. Xu. *Proceedings of the 12th GIW*, pages 24–33. Universal Academy Press, Tokyo, 2001.
12. A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA, 1974.
13. T. D. Schneider, G. D. Stormo, L. Gold, and A. Ehrenfeucht. *J. Mol. Biol.*, 188:415–431, 1986.
14. P. A. Pezner and S. Sze. *ISMB*, 8:269–278, 2000.
15. P. Pavlidis, et al. *Pac. Symp. on Biocomp.*, 2001:151–63, 2001.
16. J. W. Fickett and A. G. Hatzigeorgiou. *Genome Res.*, 7:861–878, 1997.
17. M. Q. Zhang. *Genome Res.*, 8:319–326, 1998.