Algorithms for Phylogenetic Footprinting

Mathieu Blanchette*, Benno Schwikowski†, and Martin Tompa
Department of Computer Science and Engineering
Box 352350
University of Washington
Seattle, WA 98195-2350 U.S.A.
206-543-9263
fax: 206-543-8331
{blanchem,benno,tompa}@cs.washington.edu

Abstract

Phylogenetic footprinting is a technique that identifies regulatory elements by finding unusually well conserved regions in a set of orthologous non-coding DNA sequences from multiple species. We introduce a new motif-finding problem, the Substring Parsimony Problem, which is a formalization of the ideas behind phylogenetic footprinting, and we present an exact dynamic programming algorithm to solve it. We then present a number of algorithmic optimizations that allow our program to run quickly on most biologically interesting datasets. We show how to handle data sets in which only an unknown subset of the sequences contain the regulatory element. Finally, we describe how to empirically assess the statistical significance of the motifs found. Each technique is implemented and successfully identifies a number of known binding sites, as well as several highly conserved but uncharacterized regions. The program is available at `http://bio.cs.washington.edu/software.html`.

Keywords: Phylogenetic footprinting, motif finding, regulatory elements, phylogeny, algorithm.

---

*Corresponding author
†Current address: The Institute for Systems Biology, 4225 Roosevelt Ave. NW, Suite 200, Seattle, WA 98105-0699,USA

1 Introduction and motivation

A major challenge of current genomics is to understand how gene expression is regulated. An important step towards this understanding is the capability to identify regulatory elements associated with a given gene. Most of these regulatory elements are relatively short stretches of DNA (5 to 25 nucleotides long), located in the non-coding sequence surrounding a gene. Most known regulatory elements are located 5' of the coding region, but some are also found in the 3' sequence, and even in introns. In all these cases, regulatory elements are located in otherwise non-functional sequences.

Phylogenetic footprinting (Tagle et al., 1988) is a technique that uses this functional/non-functional sequence dichotomy to identify regulatory elements. Functional sequences tend to evolve much slower than non-functional sequences, as they are subject to selective pressure. It is this difference in mutation rates that phylogenetic footprinting exploits. To identify regulatory elements associated with a given gene, one considers a set of orthologous non-coding sequences from a group of related species (for example, the non-coding sequence located 5' of the β-actin gene in ten different species of vertebrates). If these sequences contain unusually well conserved regions, it is a reasonable conjecture that these regions have some regulatory function. This approach was used with success to identify regulatory elements in various genes: *TNF*-α (Leung et al., 2000), *CFTR* (Vuillaumier et al., 1997), ϵ-*globin* (Gumucio et al., 1993; Tagle et al., 1988), γ-*globin* (Tagle et al., 1988), and *rbcL* (Manen et al., 1994), among others. See the excellent review by Duret and Bucher (1997) for more details.

This multi-species approach is to be contrasted with a much more common method for identifying regulatory elements, consisting of finding motifs in a set of non-coding sequences associated with *several* related genes from *a single* species. Over-represented patterns in these sequences are likely to correspond to binding sites of a common regulatory factor. This approach has been used successfully by a number of researchers, and several algorithms and statistical analyses have been developed (see, for example, Hertz and Stormo, 1999; Hughes et al., 2000; Roth et al., 1998; Sinha and Tompa, 2000; Tavazoie et al., 1999; van Helden et al., 1998). All these multi-gene approaches have an important inherent limitation: they will only find regulatory elements that are common to a number of genes of a given organism. Furthermore, the set of genes considered has to be obtained from experimental data (e.g. gene clusters obtained from expression array data). On the other hand, phylogenetic footprinting is capable of identifying regulatory elements that are specific to a single gene, as long as they are conserved across several species. The obvious drawback is that orthologous non-coding sequences from a sufficiently large number of species are not always available. In that sense, when phylogenetic footprinting was first proposed by Tagle *et al.* (1988), it was a little bit ahead of its time. However, the various genome projects are quickly producing sequences from a wide variety of organisms, and the data needed to perform phylogenetic footprinting are becoming more and more available.

Until recently, phylogenetic footprinting was performed by computing a global multiple alignment of the orthologous sequences, and by identifying highly conserved regions in the alignment. That approach led to significant discoveries, but is quite limited, in the following sense. First, multiple alignment is an NP-hard problem (Wang and Jiang, 1994) (although very good heuristics exist), so even if the optimal alignment could identify regulatory elements, we might not be able to compute it. More importantly, the upstream sequences considered are usually highly diverged (except for their regulatory elements). Since regulatory elements are very short compared to the sequences considered (say 10 nucleotides long in a 1000 nucleotide long sequence), it is likely that the noise of diverged non-functional sequences will overcome the short, conserved signal, in such a way that the alignment found might not align regulatory elements together. In that case, the regulatory elements would not appear to belong to conserved regions and would go undetected. Thus, when the sequences considered are moderately to highly diverged, multiple alignment is not the tool of choice.

A different approach that is likely to yield better results consists in using standard motif finding algorithms such as MEME (Bailey and Elkan, 1995), AlignAce (Hughes et al., 2000) or ANN-Spec (Workman and Stormo, 2000), or segment-based multiple alignment programs such as DIALIGN (Morgenstern et al., 1998). McCue et al. (2001) used a Gibbs sampler (Lawrence et al., 1993) to perform phylogenetic footprinting in bacterial sequences. Although we observe in this paper that these tools clearly outperform global multiple alignment, they still have an important shortcoming. Whether they optimize a consensus, sum-of-pairs, or information content criterion, none take into account the phylogenetic relationships of the given sequences. This can be problematic, for example in data sets containing a large number of closely related sequences, and a few more distant ones. If we ignore the phylogeny underlying the data, the group of closely related sequences will have an unduly high weight in the solution.

The method presented in this paper makes use of available information about the phylogenetic relationships among the species considered. Throughout this paper, we will assume that we are given a trusted phylogenetic tree. This poses little difficulties for most data sets, as the species for which genomic information is available are usually easily phylogenetically classifiable. However, the algorithms presented do not require the phylogenetic tree to be bifurcating, and multifurcating trees can be used in situations where the evolutionary relationships are unclear.

In this paper, we introduce a new type of motif-finding problem, the Substring Parsimony Problem, which is a formalization of the phylogenetic footprinting idea, and which makes use of the phylogenetic tree underlying the data. In the Substring Parsimony Problem, we are given a set of orthologous sequences $S_1, ..., S_n$ from n different species, together with the phylogenetic tree T relating these species, and an integer k. The problem is to find a set of substrings $s_1, ..., s_n$ of $S_1, ..., S_n$ respectively, each substring being of length k, such that the parsimony score of $s_1, ..., s_n$ on T is minimized. The substrings $s_1, ..., s_n$ correspond to the region that has undergone the fewest mutations, and if the conservation is sufficiently high, it is a good conjecture that these substrings are regulatory elements. We present a practical algorithm that solves the Substring Parsimony Problem *exactly*, in time linear in $n \cdot l$, where n is the total number of sequences and l is their length, but in time exponential in d, the parsimony score of the motifs sought. The Substring Parsimony Problem being NP-hard (Akutsu, 1998), (Blanchette, 2000), we cannot hope to eliminate the exponential behavior of the algorithm while still guaranteeing an optimal solution. Still, since we are generally interested in solutions with small parsimony scores, this yields a fast algorithm.

In Section 2, we give a formal description of the Substring Parsimony Problem, and describe a dynamic programming algorithm that gives an exact solution to the problem. We then describe algorithmic methods that improve the running time and space requirement by several orders of magnitude over the naive implementation of the dynamic programming algorithm.

The more species are available, the better we should be able to detect subtle signals. However, in some cases, some regulatory elements might have been lost during the evolution of some species. A solution to the Substring Parsimony Problem consists of one substring per given sequence, which means that if one or more of the species have lost the use of a regulatory element, we would most likely not find it. In Section 3, we relax this constraint by allowing sequences not to participate in the solution. This allows us to find regulatory elements conserved in only a subset of the input sequences.

With each algorithm, we report motifs found in various sets of orthologous sequences. We show that we are able to identify a large number of binding sites that have been experimentally verified. For example, many of the known binding sites upstream of the *β-actin* gene are identified by our algorithm using sequences from various vertebrates. Using the method presented in Section 3, we are able to identify three important regulatory elements for the *rbcS* gene, even though one is present only in a subset of the species considered. The sequences downstream of several genes also contain many highly conserved regions, most of which have not been characterized yet. The statistical significance of the motifs found is estimated through a procedure described in Section 4.

This paper is an extended version of two conference papers, Blanchette *et al.* (2000) and Blanchette (2001).

2 The substring parsimony problem

The Substring Parsimony Problem is a formalization of the phylogenetic footprinting idea:

SUBSTRING PARSIMONY PROBLEM

Given: a set of orthologous sequences $S_1, ..., S_n$ from n different species, the phylogenetic tree T relating these species, the length k of the motifs to look for, and an integer d.

Problem: find all sets of substrings $s_1, ..., s_n$ of $S_1, ..., S_n$ respectively, each of length k, such that the parsimony score of $s_1, ..., s_n$ on T is at most d.

Recall that the parsimony score of a set of sequences is the minimum total number of substitutions over the tree T needed to explain the observed sequences. It is defined as the minimum, over all possible labelings of the ancestral nodes of T with sequences of length k, of the sum over all edges e of the Hamming distance between the labels of the nodes connected by e. (The Hamming distance between two length k strings is the number of positions at which they differ.) Looking for sets of substrings that achieve a low parsimony score corresponds to searching for highly conserved regions. The present notion of parsimony scores allows only for substitutions, but more general mutations are considered in Section 2.6. Notice that the length k and score d of the motifs sought are given as input. In general, a few values of k might be tried by the user. For any fixed k, d can be chosen so that the motifs reported are statistically significant, in the sense described in Section 4.

2.1 A dynamic programming algorithm

We start by showing that the Substring Parsimony Problem can be solved optimally by a dynamic programming algorithm similar to that proposed by Sankoff and Rousseau (1975) for the computation of the parsimony score of a *fixed* set of strings. The algorithm assumes a rooted tree, so we will root our tree arbitrarily at an internal node r. (Since the parsimony score of a set of strings is independent of the position of the root of the tree, the choice of r will not affect the solution.) The algorithm then proceeds from the leaves up to the root. At each node u of the tree, we compute a table W_u containing 4^k entries, one for each possible sequence of length k. For a string s of length k, we define $W_u[s]$ as the best parsimony score that can be achieved for the subtree rooted at u, if u was to be labeled with s (i.e. if we force the ancestral sequence at u to be s). Let us denote by $C(u)$ the set of children of u, let $d(s,t)$ be the Hamming distance between sequences s and t, and let $\Sigma = \{\texttt{A},\texttt{C},\texttt{G},\texttt{T}\}$. The tables W can be computed by a dynamic programming algorithm:

$$W_u[s] = \begin{cases} 0 \quad \text{if } u \text{ is a leaf and } s \text{ is a substring of } S_u \\ +\infty \text{ if } u \text{ is a leaf and } s \text{ is not a substring of } S_u \\ \sum\limits_{v \in C(u)} \min\limits_{t \in \Sigma^k} (W_v[t] + d(s,t)) \text{ if } u \text{ is not a leaf} \end{cases} \tag{1}$$

Then, the score of the optimal solution to the Substring Parsimony Problem is given by $\min_{s \in \Sigma^k}(W_r[s])$. The straightforward implementation of this recurrence computes all W tables in time $O(n \cdot k \cdot (4^{2k} + l))$, where l is the average length of the input sequences $S_1, ..., S_n$. The main term in this expression comes from the fact that, for each of the $O(n)$ edges (u,v), for each of the 4^k possible values of s, and for each of the 4^k values of t, the recurrence calls for the computation of $d(s,t)$.

From that point, the ancestral sequences $s_{n+1}, ..., s_{|V|}$ and substrings $s_1, ..., s_n$ can be recovered by tracing back the recurrence, from the root down to the leaves, for each entry of W_r with score at most d. Maintaining appropriate pointers to keep track of the computation of the W tables, the set of solutions can be recovered in time linear in its size. Notice that the number of solutions may be exponential in n, although in non-repetitive biological sequences, the number of solutions is usually small (for d small), and the time to enumerate them is negligible compared to that of computing the W tables. Thus, in the remainder of this section, we focus our attention on reducing the complexity of the computation of the W tables. Indeed, the 4^{2k} factor in the complexity of the present algorithm makes it impractical to use for most interesting values of k.

2.2 Improved algorithm

We now show how to compute Equation (1) more efficiently, yielding an $O(n \cdot k \cdot (4^k + l))$ time algorithm, which makes it more practical for interesting biological purposes. We will need an auxiliary table $X_{(u,v)}$ for each edge (u, v) in the tree, where u is the parent of v. Let T' be the subtree consisting of u together with the subtree of T rooted at v. We define $X_{(u,v)}[s]$ as the best parsimony score that can be achieved on T', if u was to be labeled with s. That is, $X_{(u,v)}[s] = \min_{t \in \Sigma^k}(W_v[t] + d(s,t))$. Notice that once we have computed $X_{(u,v)}[s]$ for each string s and each child v of u, we obtain $W_u[s] = \sum_{v \in C(u)} X_{(u,v)}[s]$. We now show that the table $X_{(u,v)}$ can be computed in $O(k \cdot 4^k)$ instead of $O(k \cdot 4^{2k})$ time. The idea is similar to a breadth-first search of the space of sequences of length k. The search is divided into phases. At phase p, we consider a set of sequences B_p, called the boundary, which contains exactly the sequences s such that $X_{(u,v)}[s] = p$. Let $R_a = \{s : W_v[s] = a\}$ and let $N(t) = \{s \in \Sigma^k : d(s,t) = 1\}$ be the set of *neighbors* of string t. We start at phase 0 and set $B_0 = R_0$. To go from phase p to phase $p + 1$, we have

$$\begin{aligned} B_{p+1} &= R_{p+1} \cup \{s \in \Sigma^k : \exists t \in B_p \ s.t.\ s \in N(t)\} - \bigcup_{j \le p} B_j \\ &= R_{p+1} \cup \bigcup_{t \in B_p} N(t) - \bigcup_{j \le p} B_j \end{aligned} \tag{2}$$

Figure 1 illustrates this process. We continue this boundary expansion until all sequences have been visited and thus all $X_{(u,v)}[s]$ have been computed. Since each sequence t has exactly $3k$ neighbors and is part of the boundary only once, the computation of $X_{(u,v)}$ is done in $O(k \cdot 4^k)$ time, and the whole algorithm runs in $O(n \cdot k \cdot (4^k + l))$ time. (We assume that a string of length k fits in a single computer word.) *Figure 1 here*

2.3 Sibling bounds

We now present techniques that greatly reduce the computation time by avoiding the computation of useless entries in the W and X tables. First, note that if the only solutions of interest are those with parsimony score at most d, there is no need to compute any W_u entries that will have scores above d, as $W_u[s] > d$ implies that any global solution in which u is labeled with s will have score above d. That means that computing entries of $X_{(u,v)}$ with score above d is also useless, and so one can stop the boundary expansion after phase d. This simple observation improves the complexity of the algorithm to $O(n \cdot \min(l \cdot (3k)^d, k \cdot (4^k + l)))$, as each W and X table contains $O(l \cdot (3k)^d)$ entries. It is not difficult to see that the number of entries in some $X_{(u,v)}$ table, where v is a leaf, is $O(l \cdot (3k)^d)$, since each string in B_p has at most $3k$ neighbors in B_{p+1}. To see why it is also true when v is not a leaf, let y be an arbitrary leaf in the subtree rooted at v. Then $X_{(u,v)}[s] \le d$ implies that s must have Hamming distance at most d to some substring of S_y, which in turn implies that $X_{(x,y)}[s] \le d$. This means that $X_{(u,v)}$ has no more entries than $X_{(x,y)}$, which we showed is $O(l \cdot (3k)^d)$ since y is a leaf.

The bounding technique above also reduces the space requirement, as the tables W and X will now be very sparse, allowing us to use hash tables to store them efficiently. (We assume for the purposes of running time analysis that each hash table operation can be done in constant time.) We will refer to this bounding technique as d-bounding.

This idea can be pushed further to actually perform only $d/2$ unconstrained expansion phases for each edge. Remember that the $X_{(u,*)}[s]$ entries will eventually be added to form $W_u[s]$. Many entries $X_{(u,*)}[s]$ end up being rejected because they add up to more than d. To avoid that situation, we are going to compute the entries of the $X_{(u,*)}$ tables in parallel: first do phase zero in all tables, then phase one, etc. Then, after phase p, we know that if an entry $X_{(u,v)}[s]$ has not been computed yet, it must have a score of at least $p+1$. Thus, after phase p, we have the following bound:

$$W_u[s] \geq bound(u,s) = \sum_{v \in C(u)} \begin{cases} X_{(u,v)}[s] & \text{if } X_{(u,v)}[s] \text{ has been computed} \\ p+1 & \text{otherwise} \end{cases}$$

That means that at phase p, an entry $W_u[s]$ for which $bound(u,s) > d$ can't participate in the solution. Unfortunately, it is not true that entries failing this test can be omitted from the boundary expansion at the next phase. Indeed, it is possible that an entry s for which this bound is larger than d could have a neighbor t, visited in the next phase, for which the bound is less than or equal to d. For example, suppose u has children v, w and y, and $X_{(u,v)}[s] = d-1$ and $X_{(u,w)}[s] = X_{(u,y)}[s] = 1$, giving $bound(u,s) = d+1$. It is possible that s has a neighbor t with $X_{(u,v)}[t] = d$ and $X_{(u,w)}[t] = X_{(u,y)}[t] = 0$, giving $bound(u,t) = d$. Note however that $X_{(u,w)}[t] \geq X_{(u,w)}[s] - 1$, since otherwise s would have a smaller score $X_{(u,w)}[s]$ as a neighbor of t. This motivates the following rule for determining when an entry s need not be expanded. An entry s of $X_{(u,v)}[s] = p$ can only lead to entries with score at most d if

$$d \geq p + \max_{\substack{w \in C(u) \\ w \neq v}} \begin{cases} X_{(u,w)}[s] & \text{if } X_{(u,w)}[s] \text{ has been computed} \\ p+1 & \text{otherwise} \end{cases}$$

That means that any entry s for which this bound is not satisfied can be left out of the boundary. The reason is that, if $t \in N(s)$ and $X_{(u,v)}[t] = p+1$, then $X_{(u,w)}[t] \geq X_{(u,w)}[s]-1$, so $W_u[t] \geq X_{(u,v)}[t]+X_{(u,w)}[t] \geq X_{(u,v)}[s]+X_{(u,w)}[s] > d$. For phases $0, ..., d/2-1$, this bound is useless because it is always satisfied. However, for phases $d/2, ..., d$, it greatly reduces the size of the boundary considered. Indeed, at phase $d-i$, only those entries that have score at most i in all the siblings will be considered in the boundary. Thus, the size of the boundary at phase $d-i$ will be bounded by the size of the intersection of the boundaries of the siblings at phases $0, ..., i$. Consequently, only $d/2$ unconstrained expansion phases are peformed , and the overall time complexity of the algorithm is reduced to $O(n \cdot min(l \cdot k \cdot (3k)^{d/2}, k \cdot (4^k + l)))$. We will refer to this technique as sibling-bounding.

2.4 Parent bounds

The improvement above was obtained by computing the $X_{(u,*)}$ tables simultaneously for all children of u, and using those to constrain which entries s can lead to solutions with score at most d. We can push this idea one step further by using not only the siblings, but also the parent, to reduce the number of entries computed. To do so, we will need to compute in parallel all $X_{(u,v)}$ tables of all edges (u, v) in the entire tree, phase by phase. Furthermore, each non-terminal edge (v, u), where v is the parent of u, will have a new table associated with it: let $Y_{(u,v)}[s]$ be the best parsimony score achievable for the subgraph $((T - \text{subtree rooted at } u) \cup u)$, if u was to be labeled with s. Recall that the solutions to the Substring Parsimony Problem are independent of the node at which the tree is rooted. So, when computing entries of $X_{(u,*)}$, we will temporarily re-root the tree at u, so that $T -$ (subtree rooted at u) becomes one more subtree of u. The table associated with this new subtree is $Y_{(u,v)}$, and this extra table can be used to constrain the computation of $X_{(u,*)}$ as in Section 2.3, and the $X_{(u,*)}$ tables can be used to constrain the computation of $Y_{(u,v)}$. This new constraint does not reduce the asymptotic complexity of the algorithm, but in practice, it tends to reduce roughly by half the total number of entries computed. The drawback is that this new method requires the retention of all $X_{(u,v)}$ and $Y_{(u,v)}$ tables, whereas previously the $X_{(u,*)}$ tables could be discarded as soon as W_u was computed. We will call this technique parent-bounding.

2.5 Filtering substrings

The relatively tight bounds developed in Sections 2.3 and 2.4 make the overall complexity of the algorithm $O(n \cdot \min(l \cdot k \cdot (3k)^{d/2}, k \cdot (4^k + l)))$, significantly better than the original complexity of $O(n \cdot k \cdot (4^{2k} + l))$. Notice that the length l of the input sequences, which was completely dominated by the 4^{2k} term in the original algorithm, is now a determining factor in the running time. That is because the number of substrings inserted in the W tables at the leaves (Equation 1) directly affects the size of the boundaries explored. That suggests that reducing this number would proportionally reduce the whole running time. Indeed, if we could know in advance that a certain substring s has no chance to be part of a solution with score at most d, there would be no need to include it in W at the leaves.

There are several ways one can find out that a substring s can't produce interesting solutions. First, notice that in any solution $s_1, ..., s_n$ with parsimony score at most d, any pair of the chosen substrings must be at Hamming distance at most d. Thus, when considering whether to include substring s in W_u at a leaf u, one can find the best match of s in each of the $n-1$ other sequences. If any of these best matches has score worse than d, there is no need to consider s. This is a relatively weak bound, but for d small, it very effectively reduces the number of substrings considered.

Stronger bounds could also be used for filtering substrings. For example, if one builds an n-partite graph $G = (V, E)$, where V is the set of substrings of length k of the input sequences and $E = \{(s, t) : s \text{ and } t \text{ come from different sequences and } d(s, t) \leq d\}$, then the only substrings that can participate in an optimal solution are those that belong to an n-clique (see Pevzner and Sze (2000) for more on this idea). Obviously, this kind of filtering would be very computationally intensive and may take more time than it would save. Still, fast heuristics using these kinds of ideas seem possible. In our implementation, only the pairwise constraint was used.

2.6 General edit-distance metric

Until now, we have assumed that the only mutations allowed in the motifs sought were substitutions. However, the algorithms apply equally well to *any* set of edit operations, as follows:

Let $\mathcal{S} = (\{A, C, G, T\}^k, d)$ be a metric space where d is a string edit-distance based on a set of edit operations $\{\pi_1, ..., \pi_r\}$, each with unit cost. Then, the algorithm described above runs in $O(n \cdot \min(l \cdot r^{d/2+1}, r \cdot (4^k + l)))$. For example, if we allow substitutions, insertions, and deletions, there are $3k$ possible substitutions, $4(k+1)$ possible insertions and k possible deletions, so we get a complexity of $O(n \cdot \min(l \cdot (8k+4)^{d/2+1}, (8k+4) \cdot (4^k + l)))$. However, it may not make sense to look for motifs of length *exactly* k if insertions and deletions are allowed. Instead, we look for motifs of length *at least* k (and at most $k + d$).

One could also assign variable costs to each edit operation, for example to make insertions and deletions more expensive than substitutions, or to make mutations along short branches of the tree more expensive than those along longer branches, or to assign different weights to mutations at different motif positions. (See (Felsenstein, 1981) for an interesting connection between minimum weighted parsimony and maximum likelihood.) The boundary expansion in Section 2.2 would no longer be performed in phases, but rather table entries would be filled in one at a time, in increasing order of the entries of $X_{(u,v)}$. The complexity of the resulting algorithm would not be that mentioned above, but would be a rather complex expression, as the number of motifs at distance at most d from some string s is not easily quantifiable. In practice, however, allowing for non-unit mutation costs does not cause much increase in the running time. The resulting algorithm would then be a full-fledged algorithm for local multiple alignment on a tree.

2.7 Reporting solutions

The complete solution to the Substring Parsimony Problem is a set of sets of substrings having low parsimony score. This set could be quite large, in particular when the input sequences contain a conserved region that is much longer than k, in which case all length k substrings of the conserved region would be reported. To improve the readability of the output, we assemble the individual solutions using the following rule: if two sets of substrings overlap in exactly the same manner in each of the n input sequences, these substrings will be merged and displayed as only one solution. Notice that the resulting set of longer substrings may have a parsimony score larger than d, although each of their nucleotides belongs to at least one length k substring with score at most d.

2.8 Implementation and performance comparison

Table 1 here

The algorithms of Sections 2.2 to 2.7 were implemented in C++ in a program called FootPrinter, and the code is freely available at `http://bio.cs.washington.edu/software.html`. Table 1 presents the empirical running time and space, as well as the total number of table entries computed, when the methods described so far are applied. The data set used is the *c-myc* proto-oncogene 3' UTR sequences, from 10 different vertebrates, whose lengths vary between 450 and 900 nucleotides. The motifs sought were of length 12, with a parsimony score of at most 3 mutations. This data set is quite typical of the data available at present.

The time and memory requirements of the original algorithm (section 2.1) were too expensive to measure. Notice that when all bounding and filtering techniques are applied, we obtain a reduction in time and space of a factor of about 100 over using d-bounds only. As expected from the asymptotic complexity of the algorithm, the main factor that determines the running time is d, the score of the motifs sought, and, to a lesser extent, k, the length of the motif sought. In practice, one can easily find motifs with $k = 8$ and $d = 8$, or $k = 12$ and $d = 5$, or $k = 20$ and $d = 2$. Allowing insertions and deletions causes an increase in running time, in accordance with the asymptotic complexity. Still, the program runs quickly on a desktop machine.

3 Allowing for missing regulatory elements

Until now, we have assumed that the conserved regions we were looking for were functional in all sequences $S_1, ..., S_n$. That is, we have assumed that they had been subject to selective pressure over all branches of the tree T. That means that we were unable to find regulatory elements that were absent from some of the given species. There are many examples of data sets where some regulatory elements are common to only part of the species considered, and it would be useful to be able to find those as well, without knowing *a priori* which species contain those elements. In this section, we describe how an algorithm similar to that of Section 2.2 can handle this situation.

For this problem to make sense, we need a way to compare two solutions containing substrings from different subsets of species. For example, a motif containing m mutations in a group of ten highly diverged vertebrates is more interesting than one with the same number of mutations in a group of ten primates. What needs to be considered is the total amount of evolution (measured, for example, in million of years, in number of generations, or in overall mutation rate) that the motif has survived. Thus, in what follows, we will assume that we are not only given the phylogenetic tree T that relates the given species, but also the lengths of all its branches. If the branch lengths are unknown, they can be estimated as will be described in Section 4. (Estimating branch lengths is a notably difficult problem. However, our experience suggests that the quality of the results obtained by the method presented in this section does not critically depend on the accuracy of these estimates.) We will be looking for motifs that have a small parsimony score, but that span a large part of the tree. We assume that a regulatory element is either present at the root of the tree, or that it is acquired on exactly one branch of the tree (that is, a regulatory element is not acquired independently by two different phyla). Let the *length* of a tree be the sum of the lengths of its branches. The problem we want to solve here is the following:

SUBSTRING PARSIMONY PROBLEM WITH LOSSES

Given: a set of orthologous sequences $S_1, ..., S_n$ from n different species, the phylogenetic tree T relating these species and the length $\lambda(e)$ of each branch e of T, the length k of the motifs to look for, an integer d, and a set of thresholds $\{\delta_0, \delta_1, ..., \delta_d\}$.

Problem: find all sets of substrings $\{s_{i_1}, ..., s_{i_\zeta}\}$, $1 < \zeta \leq n$, where s_{i_j} is a k length substring of S_{i_j}, such that the parsimony score P of $\{s_{i_1}, ..., s_{i_\zeta}\}$ on the tree induced by the leaves $i_1, ..., i_\zeta$ is at most d, and such that the tree induced by the leaves $i_1, ..., i_\zeta$ is of length at least δ_P.

For example, one might be interested in motifs with a parsimony score of 2 that span at least 500 million years of evolution (i.e. $\delta_2 = 500$ Myrs), and in regions with a parsimony score of 3 that span at least 1000 million years of evolution (i.e. $\delta_3 = 1000$ Myrs).

The algorithm that solves this generalized problem is very similar in spirit to that of Section 2.2. Define a *leaf-induced subgraph* of a tree T as a connected subgraph of T in which all vertices of degree one are leaves in T. At each node u of T, we now have a set of tables $W_{u,a}$, for $a = 0, ..., d$, where $W_{u,a}[s]$ is defined as the maximal length of a connected leaf-induced subgraph of the subtree rooted at u, containing u, on which there exists a solution with parsimony score a, if u was to be labeled with s. For each edge (u, v) of the tree, we define $X_{(u,v),a}$ similarly. Let $Z_{v,a,s} = \{W_{v,b}[t] : b + d(s,t) = a, t \in \Sigma^k\}$. Then

$$X_{(u,v),a}[s] = \begin{cases} \max\{z \in Z_{v,a,s}\} + \lambda((u,v)) & \text{if } Z_{v,a,s} \neq \emptyset \\ -\infty & \text{otherwise} \end{cases} \quad \text{and}$$

$$W_{u,a}[s] = \begin{cases} 0 & \text{if } u \text{ is a leaf and } s \text{ is a substring of } S_u \\ -\infty & \text{if } u \text{ is a leaf and } s \text{ is not a substring of } S_u \\ \displaystyle\max_{\substack{q \leq |C(u)|, \\ \{c_1, c_2, ..., c_q\} \subseteq C(u), \\ \{b_1, b_2, ..., b_q\} \text{ partition of } a}} \sum_{i=1}^{q} X_{(u,c_i),b_i}[s] & \text{if } u \text{ is not a leaf} \end{cases}$$

The X tables can be computed by an algorithm very similar to that of Section 2.2. The set of substrings satisfying the given conditions are recovered by tracing back the recursion from any node u, any entry s, and any score a such that $W_{u,a}[s] \geq \delta_a$. The bounds developed in Sections 2.3 and 2.4 also apply with little modification, and filtering the input substrings could also be used, although the modifications required would be non-trivial. (In particular, a substring of sequence S_1 that has no good match with sequence S_2 can't be rejected, because S_2 might not participate in the solution.) Our current implementation uses only the sibling bounds.

4 Statistical significance of motifs

There exists a "best conserved region" in any set of sequences, whether or not these regions have actually been protected from mutations by selective pressure. To assess the significance of a best conserved region R from sequences $S_1, ..., S_n$ on tree T, one would like to do the following hypothesis testing:

$\mathbf{H_0}$: the probability of any nucleotide mutating in R is the same as that in the rest of the sequence, vs.

$\mathbf{H_1}$: the probability of any nucleotide mutating in R is less than that in the rest of the sequence.

If one knew the distribution of best parsimony scores found in sequences having evolved under H_0, one could readily obtain a significance score for a potential motif. The problem is that this distribution depends on a large number of factors, such as the sequence length l, the motif length k, the topology and branch lengths of T, the types and rates of mutations along each branch, etc. For this reason, it seems very difficult to calculate this distribution analytically, and we will thus approximate it empirically.

Let us assume that we had access to sequences $R_1, ..., R_n$ that were produced by a similar evolutionary process as $S_1, ..., S_n$, but where no region of the sequence was subject to any selective pressure. Then, one could compare the parsimony score of the best motif found in $S_1, ..., S_n$ to the one found in $R_1, ..., R_n$. If the former score were significantly lower than the latter, this might suggest that some regions of $S_1, ..., S_n$ were subject to selection. But where could we find such sequences $R_1, ..., R_n$? They might be orthologous sequences taken from the same organisms as $S_1, ..., S_n$, but in a region known to contain absolutely no functional sequence. Since it is unclear that such sequences even exist, and because it seems difficult to guarantee that the mutation rate of such sequences is the same as that of $S_1, ..., S_n$, this approach seems infeasible, and thus we will need to rely on simulated rather than biological sequences.

We generate sequences $R_1, ..., R_n$ so as to mimic the evolution of $S_1, ..., S_n$, but without selective pressure on any locus. The exact evolutionary history of $S_1, ..., S_n$ is unknown, so we first need to estimate it. We assume that throughout history, sequences have mutated according to a fixed model (in our case, an HKY substitution model (Hasekawa et al., 1985) together with insertions and deletions). The only unknown is then the length of the branches of the tree. To approximate these branch lengths, we first approximate a distance matrix M, where $M(i, j)$ is the true evolutionary distance between S_i and S_j. $M(i, j)$ can be approximated from the alignment score of S_i and S_j. Using the Fitch-Margoliash algorithm (Fitch and Margoliash, 1967), one then finds the branch lengths of T so as to fit M as closely as possible. This results in a complete approximation of the evolutionary history of $S_1, ..., S_n$. This history can then be used to generate the random sequences $R_1, ..., R_n$. This is done using the Rose program (Stoye et al., 1998), which simulates sequence evolution over a given tree. We ensure that the nucleotide frequency of $R_1, ..., R_n$ is similar to that of $S_1, ..., S_n$ by starting our simulation with a random ancestral sequence with the average nucleotide composition of $S_1, ..., S_n$. The HKY model maintains the base frequency through evolution.

The distribution of the best parsimony scores found in sequences having evolved under H_0 can now be approximated with the distribution of the scores found in these random sequences. For each data set in Table 2 and Table 3, we generated 100 sets of random sequences from which the best parsimony score distribution was approximated. For Table 2, we report the critical value $Z_{0.01}$ for the hypothesis testing, at level 0.01. Any motif with score less than $Z_{0.01}$ has probability less than 0.01 of having been generated under H_0. In Table 3, we report the approximated p-value of each of the motifs found.

5 Results

Table 2 here

Although the amount of sequences in the public domain is growing extremely fast, there are still relatively few genes for which the 5' or 3' non-coding regions have been sequenced in several species. Many of those sequences can be found in the ACUTS database (Duret et al., 1993). Table 2 reports highly conserved regions found by our algorithm in some of those sequences. We searched for motifs of length 10. Motifs longer than 10 were obtained by merging solutions that overlapped in all given sequences, as discussed in Section 2.7. The assessment of the statistical significance of the motifs found is discussed in Section 4. All motifs reported have a p-value less than 0.01.

Some of the regions identified by our algorithm are known regulatory elements, but the rest are novel conserved regions that may be worth further exploration. There is usually little known about regulatory elements in the 3' untranslated regions, but, as Duret *et al.* (1993) report, these regions contain many highly conserved regions. Interestingly, in two cases, we identified very long conserved regions in the downstream region of the given gene (data not shown). These turned out to be unannotated genes. In that sense, the algorithm could be used as an aid in gene finding.

Table 3 here

In Table 3, we illustrate the usefulness of allowing for regulatory element losses (see Section 3) using two chloroplast genes, *rbcS* and *rbcL*. These two genes contain regulatory elements that are known to be present in most, but not all, of the species considered here. All seven regions identified by our method correspond to actual functional elements. In *rbcS*, all three regulatory elements known to be present in at least 5 of the 10 input sequences were found, despite the fact that one of these regulatory elements is present only in a subset of species spanning less than 50% of the total tree length. In *rbcL*, four of the six regulatory elements known in tobacco are found, even though none of them are present in all species.

The rightmost column of Tables 2 and 3 indicates whether other motif finding techniques could have identified the motif. We tested four existing programs:

- CLUSTALW (Higgins et al., 1996) and DIALIGN 2 (Morgenstern, 1999): these are multiple alignment programs, which were followed by visual inspection of the alignment. A motif is said to be found if at least 75% of its instances are correctly aligned.
- MEME (Bailey and Elkan, 1995) and ANN-Spec (Workman and Stormo, 2000): these are motif-finding programs that make no use of the phylogeny underlying the data. A motif is said to be found by one of these two programs if it or one of its close variants (strings overlapping over at least 75% of their length) was correctly identified in at least 75% of the species considered. The top ten motifs reported by MEME and ANN-Spec were considered.

The first thing to notice is that CLUSTALW performs poorly for highly diverged sequences, as expected, for the reasons mentioned in Section 1. Because it is based on alignment of local features, DIALIGN performed extremely well and identified almost all regions FootPrinter reported, except in the case of the largest data set (*rbcL*, where the program did not terminate after one hour). MEME also performed quite well and correctly identified most regions that FootPrinter reported. This was to be expected, as most of these regions are extremely well conserved and thus fairly easy to find.

6 Conclusion and future work

In this paper, we presented an exact algorithm for phylogenetic footprinting. The algorithm is based on a simple dynamic programming formulation, but careful optimizations are needed to make the method efficient on interesting biological datasets. We described a principled way to handle data sets where regulatory elements are only present in a subset of the input sequences. All algorithms presented are guaranteed to yield all optimal solutions to the variety of problems addressed. Algorithms are implemented and run quickly on a desktop computer. Using them, we have been able to identify a number of experimentally determined binding sites, as well as a set of highly conserved regions with no function known yet.

Future work should be directed in three main directions. First, it seems possible to improve the bounds presented in Sections 2.3 and 2.4, as well as the filtering accuracy and time complexity. Second, an accurate assessment of the significance seems a difficult problem and our estimation of the statistical significance of the motifs found is based on a number of approximations that may be refined. Third, it appears that quite often, regulatory elements work in pairs (or even triples), at a relatively fixed distance from each other. This new kind of prior knowledge could easily be included in our algorithm and may allow us to find pairs of regulatory elements that were too weakly conserved to be detected by themselves. Other ways of injecting prior knowledge about the solutions of interest should also be considered.

From an application point of view, the next years should provide us with a bulk of new data to analyze. Interesting tasks include not only regulatory element prediction, but also gene and splice site detection, as well as protein and RNA structural motif finding.

7 Acknowledgements

This material is based upon work supported in part by a Natural Sciences and Engineering Research Council of Canada (NSERC) fellowship, by the German Academic Exchange Service (DAAD), by the National Science Foundation and DARPA under grant DBI-9601046, and by the National Science Foundation under grant DBI-9974498.

References

Akutsu, T. 1998. Hardness results on gapless local multiple sequence alignment. Technical Report 98-MPS-24-2, Information Processing Society of Japan.

Arguello-Astorga, G. and Herrera-Estrella, L. 1998. Evolution of light-regulated plant promoters. *Annu Rev Plant Physiol Plant Mol Biol* 49, 525–555.

Bailey, T. L. and Elkan, C. 1995. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning* 21, 51–80.

Blanchette, M. 2000. *An Exact Algorithm to Identify Motifs in Orthologous Sequences from Multiple Species.* Qualification project, University of Washington.

Blanchette, M. 2001. Algorithms for phylogenetic footprinting. In *RECOMB01: Proceedings of the Fifth Annual International Conference on Computational Molecular Biology*, 49–58. ACM Press, Montreal, Canada.

Blanchette, M., Schwikowski, B. and Tompa, M. 2000. An exact algorithm to identify motifs in orthologous sequences from multiple species. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, 37–45. AAAI Press, La Jolla, USA.

DePonti-Zilli, L., Seiler-Tuyns, A. and Paterson, B. 1988. A 40-base-pair sequence in the 3' end of the beta-actin gene regulates beta-actin mRNA transcription during myogenesis. *Proc Natl Acad Sci USA* 85:1389-93.

Duret, L. and Bucher, P. 1997. Searching for regulatory elements in human noncoding sequences. *Curr Op in Struct Biol* 7, 399–405.

Duret, L., Dorkeld, F. and Gauthier, C. 1993. Strong conservation of non-coding sequences during vertebrates evolution: potential involvement in post-transcriptional regulation of gene expression. *Nucleic Acids Research* 21, 10, 2315–2322.

Felsenstein, J. 1981. A likelihood approach to character weighting and what it tells us about parsimony and compatibility. *Biological Journal of the Linnean Society* 16, 183–196.

Fitch, W. M. and Margoliash, E. 1967. Construction of phylogenetic trees. *Science* 155, 279–284.

Frederickson, R. M., Micheau, M. R., Iwamoto, A. and Miyamoto, N. G. 1989. 5' flanking and first intron sequences of the human beta-actin gene required for efficient promoter activity. *Nucleic Acids Res* 17, 253–270.

Gumucio, D., Shelton, D., Bailey, W., Slightom, J. and Goodman, M. 1993. Phylogenetic footprinting reveals unexpected complexity in trans factor binding upstream from the ϵ-globin gene. *Proc Natl Acad Sci USA* 90, 6018–6022.

Hasekawa, M., Kishino, H. and Yano, T. 1985. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *J Mol Evol* 22, 160–174.

Hertz, G. Z. and Stormo, G. D. 1999. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics* 15, 563–577.

Higgins, D. G., Thompson, J. D. and Gibson, T. J. 1996. Using CLUSTAL for multiple sequence alignments. In Doolittle, R. F., ed., *Computer Methods for Macromolecular Sequence Analysis*, volume 266 of *Methods in Enzymology*, 383–401. Academic Press, New York.

Hughes, J., Estep, P., Tavazoie, S. and Church, G. 2000. Computational identification of cis-regulatory elements associated with groups of functionally related genes in Saccharomyces cerevisiae. *J Mol Biol* 296(5), 1205–14.

Kislauskis, E., Zhu, X. and Singer, R. 1994. Sequences responsible for intracellular localization of beta-actin messenger RNA also affect cell phenotype. *J Cell Biol* 127, 441–451.

Lawrence, C. E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F. and Wootton, J. C. 1993. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science* 262, 208–214.

Leung, J., McKenzie, E., Uglialoro, A., Florez-Villanueva, P., Sorkin, B., Yunis, E., Hartl, D. and Goldfeld, A. 2000. Identification of phylogenetic footprints in primate tumor necrosis factor-α promoters. *Proc Natl Acad Sci USA* 97, 12, 6614–6618.

Manen, J. F., Savolainen, V. and Simon, P. 1994. The *atpB* and *rbcL* promoters in plastid DNAs of a wide dicot range. *Journal of Molecular Evolution* 38, 577–582.

McCue, L., Thompson, W., Carmack, C., M.P., R., Liu, J., Derbyshire, V. and Lawrence, C. 2001. Phylogenetic footprinting of transcription factor binding sites in proteobacterial genomes. *Nucleic Acids Research* Feb 1;29(3), 774–82.

Morgenstern, B. 1999. DIALIGN 2: Improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics* 15, 211–218.

Morgenstern, B., Frech, K., Dress, A. and Werner, T. 1998. DIALIGN: Finding local similarities by multiple sequence alignment. *Bioinformatics* 14, 290–294.

Pevzner, P. and Sze, S. 2000. Combinatorial approaches to finding subtle signals in DNA sequences. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, 269–278. AAAI Press.

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

Sankoff, D. and Rousseau, P. 1975. Locating the vertices of a Steiner tree in arbitrary metric space. *Mathematical Programming* 9, 240–246.

Sinha, S. and Tompa, M. 2000. A statistical method for finding transcription factor binding sites. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, 344–354. AAAI Press, San Diego, CA.

Stoye, J., Evers, D. and Meyer, F. 1998. Rose: generating sequence families. *Bioinformatics* 14:2, 157–163.

Tagle, D., Koop, B., Goodman, M., Slightom, J., Hess, D. and Jones, R. 1988. Embryonic ϵ and γ globin genes of a prosimian primate (Galago crassicaudatus) nucleotide and amino acid sequences, developmental regulation and phylogenetic footprints. *J Mol Biol* 203, 439–455.

Tavazoie, S., Hughes, J. D., Campbell, M. J., Cho, R. J. and Church, G. M. 1999. Systematic determination of genetic network architecture. *Nature Genetics* 22, 281–285.

van Helden, J., André, B. and Collado-Vides, J. 1998. Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *Journal of Molecular Biology* 281, 827–842.

Vuillaumier, S., Dixmeras, I., Messai., H., Lapoumeroulie, C., Lallemand, D., Gekas, J., Chebab, F., Perret, C., Elion, J. and Denamur, E. 1997. Cross-species characterization of the promoter region of the cystic fibrosis transmembrane conductance regulator gene reveals multiple levels of regulation. *Biochem J* 327, 652–662.

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

Workman, C. T. and Stormo, G. D. 2000. ANN-Spec: a method for discovering transcription factor binding sites with improved specificity. In *Pacific Symposium of Biocomputing*, 467–78.

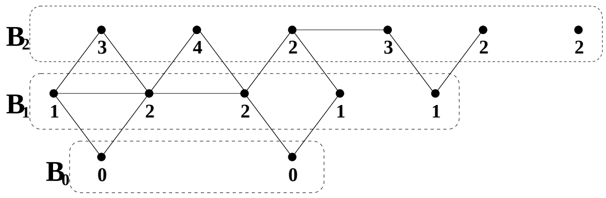

Figure 1: Illustration of the progression of the boundary B. Each dot corresponds to a substring. The number associated with dot x is $W_u[x]$. Arcs connect neighboring substrings.

Table 1: PERFORMANCE IMPROVEMENTS OBTAINED FROM BOUNDING AND FILTERING TECHNIQUES

	No indel			With indels		
Technique	Time	Space	#Entries	Time	Space	#Entries
No bounds	-	-	$\sim 160 \times 10^6$	-	-	$\sim 1 \times 10^9$
d-bounds only	∼500s	∼1000M	$\sim 20 \times 10^6$	∼10000s	∼5000M	$\sim 150 \times 10^6$
Sibling bounds only	32s	40M	810130	727s	340M	6999597
Sibling+parent bounds	24s	39M	414207	573s	315M	3642586
All bounds+Filtering	4s(+10s filtering)	8M	60962	80s(+90s)	51M	559305

Performance obtained when the various bounding and filtering techniques described in Sections 2.3, 2.4, 2.5 are used. Data set: *c-myc* proto-oncogene 3' UTR sequences, k=12, d=3, n=10. Sequence length varies between 450 and 900 nucleotides. The solution consists of 3 distinct conserved substrings. The program ran on a Pentium III 550 MHz, with a 512M RAM.

Table 2: HIGHLY CONSERVED REGIONS FOUND IN SETS OF ORTHOLOGOUS SEQUENCES

Region	Species	$Z_{0.01}$	Motif	Score	Ref.	Other methods
β-actin 5'	1,2,5,9,11	2	ccaatcagcg (-934)	1	<1a>	A,D,M
(1091 bp)	(1200Myrs)		ttgccttttatggc (-915)	0	<1b>	M
			ttgccttttatggtaat (-160)	0	<1c>	D,M
			cttcctttgtc (-131)	0		D,M
β-actin 3'	1,2,3,4,5,6,	5	ttggcatggctt (70)	1		D,M,W
(636 bp)	9,11,12,13		ttgactcaggat (127)	1		D,M,W
	(1900Myrs)		aaaaactggaac (141)	1		A,D,M,W
			agtcattccaaat (242)	2		D,M,W
			tgtaaattatgt (374)	2		A,D,M
			cctgtacactgac (545)	0	<2>	A,D,M,W
γ-actin 3'	1,5,6,7,8,10	2	tcatgctagcctc (78)	0		A,D,M,W
(427 bp)	(1100Myrs)		aaactggaataag (94)	0		A,D,M,W
			tgtgcagggtat (353)	1		D,M

Highly conserved sequences found by our algorithm in non-coding regions. The sequence and position of the motifs reported correspond to those in human. Motif positions are from the start codon for 5' sequences, and from the stop codon for 3' sequences. All substrings of length 10 of the reported motifs contain at most the reported number of mutations. Poly-Ts are not reported. $Z_{0.01}$ is the critical value for the hypothesis testing of Section 4, at level 0.01. Motifs with score less than $Z_{0.01}$ have p-value less than 0.01. **References:** <1a> Bound by unknown factor (Frederickson et al., 1989), <1b> Bound by *SRF* (Frederickson et al., 1989), <1c> subcellular localization signal (Kislauskis et al., 1994), <2> responsible for down-regulation during myogenesis (DePonti-Zilli et al., 1988), **Species:** (1) human, (2) common carp, (3) hamster, (4) rabbit, (5) rat, (6) mouse, (7) cow, (8) frog, (9) chicken, (10) goose, (11) grass carp, (12) medaka fish, (13) gilthead sea bream . **Other methods** (see Section 5): A: Ann-spec, D: Dialign2, M: Meme3, W: ClustalW.

Table 3: HIGHLY CONSERVED REGIONS FOUND IN SUBSETS OF SETS OF CHLOROPLAST ORTHOLOGOUS SEQUENCES

Region	Species	Motif	Score	Ind. length	P-value	Ref.	Oth. meth.
rbcS 5'	10 embryophyta(1)	tatatatag (-105)	4	100%	<0.01	<1a>	D,M
		agataaga (-327)	5	100%	<0.01	<1b>	A,M
		cacgtggc (-303)	1	49%	0.38	<1c>	A,D,M
rbcL 5'	44 Magnoliophyta(2)	ggatttacatata(-326)	0	64%	<0.01	<2a>	A,M
		tatacaata (-199)	1	79%	<0.01	<2b>	W
		tcgagtagacct (-59)	0	66%	<0.01	<2c>	M,W
		gaattcttaattcatgag...					
		...ttgtagggaggga (-39)	0	75%	<0.01	<2d>	A,M,W

Highly conserved regions found in a subset of the input sequences. All substrings of length 8 (for *rbcS*) and 9 (for *rbcL*) of the reported motifs contain at most the reported number of mutations on the induced subtree. The motifs and positions reported are those in tobacco. **Species :** (1) tobacco, spinach, tomato, ice plant, rape, cotton, pea, wheat, duckweed, corn. (2) 1 fern, 1 conifer, 5 lilliopsidae, 6 magnoliales, 11 asteridae, 1 buxaceae, 19 rosidae. **Induced length**: Relative length of the tree induced by the species containing the motif. **References:** From (Arguello-Astorga and Herrera-Estrella, 1998): <1a> TATA-box, <1b> I-box , <1c> G-box, absent from corn, wheat, duckweed and rape. From (Manen et al., 1994): <2a> *atpB* promoter (*atpB* is located upstream of *rbcL*, on the opposite strand), absent from spruce, corn, rice, fern and *Spermacoceas surgens*, and mutated in two other species. <2b> *rbcL* promoter, similar to -10 promoter, absent from cotton, fern, and *cuscuta*, and mutated in three other species. <2c> 5' part of leader region, absent from corn and *cuscuta*, and mutated in two other species. <2d> 3' part of leader region, absent from *cuscuta*, and mutated in four other species. **Other methods**: see Table 2.