Algorithms for Finding Maximal-Scoring Segment Sets* (extended abstract)

Miklós Csűrös

Département d'informatique et de recherche opérationnelle
Université de Montréal
C.P. 6128 succ. Centre-Ville, Montréal, Québec, H3C 3J7, Canada
csuros@iro.umontreal.ca

Abstract. We examine the problem of finding maximal-scoring sets of disjoint regions in a sequence of scores. The problem arises in DNA and protein segmentation, and in post-processing of sequence alignments. Our key result states a simple recursive relationship between maximal-scoring segment sets. The statement leads to an algorithm that finds such a k-set of segments in a sequence of length n in $O(nk)$ time. We describe linear-time algorithms for finding optimal segment sets using different criteria for choosing k, as well as an algorithm for finding an optimal set of k segments in $O(n \log n)$ time, independently of k. We apply our methods to the identification of non-coding RNA genes in thermophiles.

1 Introduction

Suppose that $w_1, w_2, \ldots, w_n \in \mathbb{R}$ is an arbitrary sequence of scores with $n > 0$. A *segment* S is a set of consecutive integers: $S = [a, b] = \{a, a+1, \ldots, b\}$. The *score of a segment* S is the sum of the scores indexed by the segment's elements: $w(S) = \sum_{i \in S} w_i$. A classic example of algorithm design is Jon Bentley's Programming Pearl [1] for finding a segment with maximum score. Such a segment can be found in linear time by scanning the scores once. This paper considers a natural generalization of the maximum-scoring segment problem. Namely, we are interested in finding k disjoint segments with maximum total score. A *k-cover* $\mathcal{C} = \{S_1, \ldots, S_k\}$ is a non-intersecting family of segments. The score of a k-cover $\mathcal{C}$ is the sum of its elements' score: $w(\mathcal{C}) = \sum_{S \in \mathcal{C}} w(S)$. It is useful to define the *indicator vector* $(z_1, \ldots, z_n)$ of a cover $\mathcal{C}$: $z_i = 1$ if $i \in \cup_{S \in \mathcal{C}} S$ and $z_i = 0$ otherwise. Using this notation, $w(\mathcal{C}) = \sum_{i=1}^{n} w_i z_i$. A k-cover is *maximal* if it has maximum score among all k-covers. We define the 0-cover as the empty set with score 0.

A cover may define a segmentation, which alternates high- and low-scoring regions, i.e., segments within and outside the cover. Segmentation methods have been extensively used in the analysis of protein and DNA sequences [2]. Various scoring schemes permit the identification of charge clusters and hydrophobic profiles for proteins [3], determination of isochores in DNA sequences [4, 5], discovery

* Work supported by NSERC grant 250391-02.

of CpG islands [5, 6], and even gene finding [7]. Different methods include maximum likelihood estimation [4], Hidden Markov Models [8, 7], entropy-based [5], and various "moving window" techniques. Segmentation methods are also used to remove low-scoring regions from sequence alignments [9].

Our key result is Theorem 1, which states the incremental nature of maximal covers. This theorem leads to an algorithm that finds a k-cover with maximum score for $k \leq K$ in $O(nK)$ time where K is an upper bound on the cover size. Section 3 describes the algorithms for finding maximal covers using different optimality criteria, as well as an algorithm for finding a maximal k-cover in $O(n \log n)$ time. Section 4 deals with the problem of identifying GC-rich regions in AT-rich genomes, which coincide with non-coding RNA genes in thermophiles. Section 5 discusses related results and concludes the paper.

Theorem 1. *Let $\mathcal{C}_k$ be a maximal k-cover for $k \in [0, n-1]$. There exists a maximal $(k+1)$-cover $\mathcal{C}_{k+1}$ which satisfies one of the following conditions.*

(1) There exists such a segment $[a, b]$ that $\mathcal{C}_{k+1} = \mathcal{C}_k \cup \{[a, b]\}$; or
(2) there exist $a, b, c, d \in [1, n]$ for which $a \leq c < d \leq b$, and $\mathcal{C}_{k+1} = \mathcal{C}_k \cup \{[a, c], [d, b]\} \setminus \{[a, b]\}$.

Theorem 2. *Let $\mathcal{C}_k$ be a maximal k-cover for $k \in [1, n]$. There exists a maximal $(k-1)$-cover $\mathcal{C}_{k-1}$ which satisfies one of the following conditions.*

(1) There exists such a segment $[a, b] \in \mathcal{C}_k$ that $\mathcal{C}_{k-1} = \mathcal{C}_k \setminus \{[a, b]\}$; or
(2) there exist $a, b, c, d \in [1, n]$ for which $a \leq c < d \leq b$, and $\mathcal{C}_{k-1} = \mathcal{C}_k \cup \{[a, b]\} \setminus \{[a, c], [d, b]\}$.

Theorem 1 shows that $\mathcal{C}_{k+1}$ is obtained either (1) by adding a new segment to $\mathcal{C}_k$, or (2) by removing the middle of a segment in $\mathcal{C}_k$. By Theorem 2, the converse is also true: a maximal $(k-1)$-cover can be created from a k-cover by merging two segments, or by removing one. Theorem 2 implies also that *all* maximal covers can be produced by consecutive applications of operations (1) and (2) of Theorem 1. The theorems' proofs are omitted here due to space constraints. The theorems have two immediate consequences. First, Corollary 1 below shows that the score of maximal k-covers is a concave function of k. Secondly, Theorem 1 implies a simple algorithm for calculating successive maximal covers, which we describe in §3.1.

Corollary 1. *Let $1 < k < n$. Let $\mathcal{C}_{k-1}$, $\mathcal{C}_k$, and $\mathcal{C}_{k+1}$ be maximal $(k-1)$-, k-, and $(k+1)$-covers, respectively. Then $w(\mathcal{C}_{k+1}) - w(\mathcal{C}_k) \leq w(\mathcal{C}_k) - w(\mathcal{C}_{k-1})$.*

Proof. Omitted. □

2 Scores based on probabilistic models

2.1 Maximum likelihood estimation of segments

Let $X_1, \ldots, X_n$ be a sequence of independent random letters from an alphabet $\Sigma = \{\sigma_1, \ldots, \sigma_r\}$. The distribution of every X_i is one of two known distributions, specified by the probabilities $p(\sigma_j)$ and $q(\sigma_j)$. A *changed segment*

is a segment $[a, b]$ of indices where $\mathbb{P}\{X_i = \sigma_j\} = q(\sigma_j)$ for all $i \in [a, b]$. A segment $[a, b]$ is *unchanged* if $\mathbb{P}\{X_i = \sigma_j\} = p(\sigma_j)$ for all $i \in [a, b]$. Maximum likelihood estimation of changed segments turns into a maximal cover problem. Let $x_i \colon i \in [1, n]$ be the observed values of X_i. Let $\mathcal{C}$ be a non-intersecting set of hypothetical changed segments. Let $\mathbf{z} = (z_1, \ldots, z_n)$ be the indicator vector for $\mathcal{C}$. The likelihood function is $f\big(\mathbf{x} \bigm| \mathbf{z}, \mathbf{p}, \mathbf{q}\big) = \prod_{i=1}^{n} (p(x_i))^{1-z_i} (q(x_i))^{z_i}$. Define

$$w_i = \log(q(x_i)) - \log(p(x_i)). \tag{1}$$

(Throughout the paper, log denotes natural logarithm.) The log-likelihood can be written as $\log f\big(\mathbf{x} \bigm| \mathbf{z}, \mathbf{p}, \mathbf{q}\big) = \sum_{i=1}^{n} \log p(x_i) + \sum_{i=1}^{n} w_i z_i$. The first term is the log-likelihood of the null hypothesis that there are no changed segments. The second term is the log-likelihood ratio (LLR) of the alternative hypothesis defined by $\mathcal{C}$. Accordingly, a maximal k-cover maximizes the LLR among hypotheses with k changed segments if the scores are set by Eq. (1).

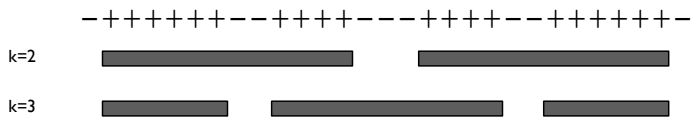

Fig. 1. Maximal k-covers with minimum segment lengths $m_1 = 6$ and $m_0 = 1$. A '+' denotes $w_i = 1$ and '-' denotes $w_i = -1$. An equivalent scoring scheme is realized when $\Sigma = \{0, 1\}$, and $p(0) = 1 - q(0)$ in Eq. (1).

Fu and Curnow [4] examine the problem of finding a k-set of changed segments that maximizes the LLR with restrictions on the minimum lengths of changed and unchanged segments specified by thresholds m_1 and m_0, respectively. Fu and Curnow state a theorem (with an incomplete proof) that is similar to Theorem 1: "Given one set of best k segments [k-cover in our terminology], we can find one set of best $(k + 1)$ segments if it exists, by either adding the best segment which does not overlap with any of the k best segments or by splitting and expanding one of the best k segments." Their claim, however, does not seem to hold in general, as the relationship between maximal covers may be complicated if a minimum length is imposed on the segments. Figure 1 shows an example where more than one segment change between consecutive maximal covers. Alternatively, if segments can be of arbitrary length, then by Theorem 1, there is no need for expansions.

2.2 Selecting the cover size: complexity penalties

Unless it is warranted by the problem at hand, the reason for restricting segment lengths is to avoid overfitting: the cover $\{[i, i] \colon w_i > 0\}$ maximizes the likelihood but it hardly captures any meaningful pattern in the data. We suggest that one

should instead penalize the cover size. Define the *complexity-penalized score* of a cover $\mathcal{C}$ by $\tilde{w}(\mathcal{C}) = w(\mathcal{C}) - r(|\mathcal{C}|)$ where $r\colon \mathbb{N} \mapsto [0,\infty)$ is a monotone increasing penalty function. The optimal cover has maximum complexity-penalized score. First we describe a penalty based on the minimum description length (MDL). According to the MDL principle [10], one favors the cover $\mathcal{C}$ which minimizes the length of encoding the data and $\mathcal{C}$. Let $\mathbf{z}$ be the indicator vector for $\mathcal{C}$. Given $\mathbf{z}$, every x_i can be encoded in $b(z_i, x_i)$ bits on average, where $b(0,\sigma) = -\log p(\sigma)$ and $b(1,\sigma) = -\log q(\sigma)$. The cover itself can be specified by the endpoints of its segments using $2|\mathcal{C}|\log_2 n$ bits. The total codelength equals $\ell(\mathbf{x},\mathcal{C}) = \sum_{i=1}^{n} b(z_i,x_i) + 2|\mathcal{C}|\log_2 n = \ell(\mathbf{x},\emptyset) - \frac{w(\mathcal{C})-2|\mathcal{C}|\log n}{\log 2}$. The MDL cover thus maximizes $w(\mathcal{C}) - 2|\mathcal{C}|\log n$, and corresponds to the penalty $r(k) = 2k\log n$. (A more efficient encoding can rely on the fact that there are $\binom{n}{2k}$ possible k-covers. When $k = o(n)$, a k-cover can be encoded in $\log_2\binom{n}{2k} \approx 2k\log_2 n - 2k\log_2(2k)$ bits. The corresponding penalty equals $r(k) = 2k(\log n - \log(2k))$.)

2.3 A penalty based on statistical significance

As an alternative to the MDL approach, a penalty can be defined based on statistical significance, measured by the probability that a segment has a large score under the null hypothesis that there are no changed segments. The distribution of the maximum segment score, i.e., the score $w^{(1)}$ of the maximal 1-cover, under the null hypothesis has been extensively studied [11, 12]. Karlin *et al.* [12] prove that $w^{(1)} \to \log n$ almost surely as $n \to \infty$, and that for all x,

$$\lim_{n\to\infty} \mathbb{P}\big\{w^{(1)} - \log n \le x\big\} = \exp\big(-Ce^{-x}\big), \tag{2}$$

where C is independent of n and x, and is defined by a rapidly converging infinite sum. (For the general case of assigning score u_j to every letter σ_j with $\sum_{j=1}^{r} p(\sigma_j)u_j < 0$, $w^{(1)} \to \lambda^{-1}\log n$ where λ is the unique positive solution of $\sum_{j=1}^{r} p(\sigma_j)\exp(\lambda u_j) = 1$. When $u_j = \log\frac{q(\sigma_j)}{p(\sigma_j)}$, $\lambda = 1$ is a solution.) This result provides a means to select α, in order to search for the cover $\mathcal{C}$ that is optimal according to a penalty function $r(k) = \alpha k$. The following theorem characterizes the segment scores in $\mathcal{C}$.

Theorem 3. *Fix $\alpha > 0$, and let $\mathcal{C} = \{[a_1,b_1],\ldots,[a_k,b_k]\}$ be a cover that maximizes $\tilde{w}(\mathcal{C}) = w(\mathcal{C}) - \alpha|\mathcal{C}|$. If $\mathcal{C} \neq \emptyset$, then the following holds. For all $i \in [1,k]$, $w([a_i,b_i]) \ge \alpha$, and there does not exist a,b with $a_i < a \le b < b_i$ and $w([a,b]) < -\alpha$. For all $i \in [1,k-1]$, if $b_i + 1 < a_{i+1}$, then $w([b_i+1, a_{i+1}-1]) \le -\alpha$. For all $i \in [0,k]$, there does not exist a,b for which $b_i < a \le b < a_{i+1}$ and $w([a,b]) > \alpha$, where $b_0 = 0$ and $a_{k+1} = n+1$.*

Proof. Straightforward. □

By Theorem 3, every changed segment in $\mathcal{C}$ has score at least α and no subsegment of an unchanged segment has a score above α. Consequently, by setting $\alpha = x + \log n$ with an appropriately chosen x we can ensure that every

changed segment has significant statistical support, and that a maximal set of such segments is selected. In particular, Eq. (2) implies that for large n, $\mathcal{C}$ is non-empty with probability $1-\exp(-Cne^{-\alpha})$ under the null hypothesis. Accordingly, for a given [small] $0 < p < 1$, we can use

$$\alpha \geq \log n + \log \frac{C}{-\log(1-p)} \approx \log n + \log \frac{C}{p}, \tag{3}$$

in order to get a non-empty optimal cover with at most p probability. By switching the roles of changed and unchanged segments, a similar argument can be made to measure the statistical support for unchanged segments.

2.4 Two-state Hidden Markov Models

Our last example of penalizing cover size is that of segmentation by a Hidden Markov Model (HMM). Extending the maximum likelihood framework, we impose that the random sequence $X_1, \ldots, X_n$ is generated by a two-state HMM [8, 7]. The two states correspond to changed and unchanged segments. A run of the HMM results in a state sequence $Z_1, \ldots, Z_n$ forming a Markov chain, and the sequence of emitted characters $X_1, \ldots, X_n$. If $Z_i = 0$, then X_i is drawn according to the unchanged segments' distribution $\mathbf{p}$, otherwise it is drawn according to $\mathbf{q}$. The most likely state sequence $\mathbf{z} = (z_1, \ldots, z_n)$ for a given observation sequence $\mathbf{x} = (x_1, \ldots, x_n)$ defines a segmentation of $[1, n]$ into changed and unchanged segments, i.e., segments where $z_i = 1$ vs. segments where $z_i = 0$. Clearly, $\mathbf{z}$ is the indicator vector for a cover. The likelihood function equals $f(\mathbf{x} \mid \mathbf{z}) = \pi(z_1)\Big(\prod_{i=1}^{n}(p(x_i))^{1-z_i}(q(x_i))^{z_i}\Big)\Big(\prod_{i=2}^{n}\tau(z_{i-1} \to z_i)\Big)$, where π are the starting probabilities and τ are the transition probabilities for the states' Markov chain. There exists a well-known method for finding the most likely state sequence, known as the Viterbi algorithm [13], but formulating it as a maximal cover problem enables us to consider further variations with restrictions on the number of state changes (§3.1) or on state durations (§3.2). The LLR of a state sequence $\mathbf{z}$ (viewed as indicator for a cover $\mathcal{C}$) with respect to the null hypothesis that all $z_i = 0$ can be written in the form $\sum_{i=1}^{n} w_i z_i - \alpha|\mathcal{C}|$, where

$$w_i = \log \frac{q(\sigma)}{p(\sigma)} + \log \frac{\tau(1 \to 1)}{\tau(0 \to 0)} + \delta_i; \tag{4a}$$

$$\alpha = -\log \frac{\tau(0 \to 1)}{\tau(0 \to 0)} - \log \frac{\tau(1 \to 0)}{\tau(0 \to 0)} + \log \frac{\tau(1 \to 1)}{\tau(0 \to 0)}, \tag{4b}$$

and $\delta_i = 0$ for every $i \in [2, n-1]$, otherwise it hides correction terms: $\delta_1 = -\log \frac{\tau(0\to1)}{\tau(0\to0)} + \log \frac{\pi(1)}{\pi(0)}$ and $\delta_n = -\log \frac{\tau(1\to0)}{\tau(0\to0)}$. Consequently, segmentation by the most likely state sequence in a two-state HMM is an instance of finding an optimal cover using linear complexity penalties.

3 Algorithms

3.1 An algorithm for finding a maximal cover

By Theorem 1, a maximal $(k+1)$-cover can be found by updating a maximal k-cover. For each k, one needs to find the segment that can be either added or removed to increase the cover score by the largest amount. The idea is employed by the algorithm MaxCover, which is an adaptation of Bentley's algorithm [1]. (In fact, Bentley credits Joseph Kadane of CMU with the design.)

Algorithm MaxCover
Input: w_i scores for $i \in [1, n]$, K maximum cover size
Output: indicator vector for a k-cover with maximum score for $0 \le k \le K$
C1 initialize $z_i \leftarrow 0$ for $i = 1, \ldots, n$
C2 **for** $k \leftarrow 1, \ldots, K$ **do**
C3 set $i_0 \leftarrow 1$; $w \leftarrow 0$; $S \leftarrow$ null; $w_{\max} \leftarrow 0$
C4 **for** $i \leftarrow 1, \ldots, n$ **do**
C5 **if** $i > 1$ and $z_{i-1} \neq z_i$ **then** $i_0 \leftarrow i$; $w \leftarrow 0$
C6 $w \leftarrow w + w_i$ *//current candidate is* $[i_0, i]$ *with score* w
C7 **if** ($z_i = 0$ and $w \le 0$) or ($z_i = 1$ and $w \ge 0$) **then** $i_0 \leftarrow i+1; w \leftarrow 0$
C8 **else if** $|w| > w_{\max}$ **then** $w_{\max} \leftarrow |w|$; $S \leftarrow [i_0, i]$
C9 **if** $w_{\max} = 0$ **then return** $(z_1, \ldots, z_n)$ **else** set $z_i \leftarrow 1 - z_i$ for all $i \in S$
C10 **return** $(z_1, \ldots, z_n)$

The algorithm scans the scores w_i once for every $k \in [1, K]$ in Lines C4–C8. For every k, the algorithm calculates the maximum increase $w_{\max}$ in cover score that can be achieved by removing a sub-segment or adding a segment (the segment S).

Lemma 1. *The algorithm* MaxCover *finds a cover that has maximum score among covers with at most K segments in $O(nK)$ time.*

Proof. The proof of the running time is straightforward. The proof of correctness is analogous to that of [1]; it is omitted due to space constraints. □

3.2 Algorithms for linear complexity penalties

Suppose that we want to find the cover $\mathcal{C}$ that maximizes the penalized score $\tilde{w}(\mathcal{C}) = w(\mathcal{C}) - \alpha|\mathcal{C}|$ with some $\alpha \ge 0$. The MDL approach of §2.1 sets $\alpha = 2 \log n$; the statistical significance framework (setting α by Eq. (3)), and the HMM approach of §2.4 also use linear penalty functions.

Let $\mathcal{C}_0 = \emptyset, \mathcal{C}_1, \mathcal{C}_2, \ldots$ be a series of maximal k-covers. By Corollary 1, a cover $\mathcal{C}^*$ maximizing $\tilde{w}$ is the first $\mathcal{C}_k$ for which $w(\mathcal{C}_{k+1}) - w(\mathcal{C}_k) < \alpha$. It is easy to modify MaxCover to find $\mathcal{C}^*$. The only necessary change is in Line C9, where $\mathbf{z}$ needs to be returned if $w_{\max} \le \alpha$. MaxCover then finds $\mathcal{C}^*$ in $O(nK)$ time if it is invoked with $K \ge |\mathcal{C}^*|$. In what follows we develop a faster algorithm.

For all $i \in [1, n]$, define $W^0(i)$ as the maximum of $\tilde{w}$ for covers of $[1, i]$ which do not include i. Define $W^1(i)$ as the maximum of $\tilde{w}$ for covers of $[1, i]$ which do include i.

Lemma 2. *For all* $i > 1$, $W^0(i) = \max\{W^0(i-1), W^1(i-1)\}$, *and* $W^1(i) = w_i + \max\{W^0(i-1) - \alpha, W^1(i-1)\}$.

Proof. Straightforward by using the definition. □

The lemma implies a dynamic programming algorithm. In case of the two-state HMM, the algorithm is equivalent to the Viterbi algorithm [13]. We design a more general method that respects minimum segment length constraints. Specifically, we want to find a cover that maximizes $\tilde{w}$ with the stipulation that changed segments must have lengths at least m_1 and unchanged segments must have lengths at least m_0.

For all $j = 0, 1$, $m \in [1, m_j]$, and $i \in [m, n]$, define $\mathcal{C}^j_{i,m}$ as covers of $[1, i]$ that maximize $\tilde{w}$ while satisfying the requirements for all segment lengths, except for the last one: $\mathcal{C}^0_{i,m}$ is a cover that ends with an unchanged segment of length at least m, and $\mathcal{C}^1_{i,m}$ ends with a changed segment of length at least m.

Lemma 3. *Let* $W^0_{\text{short}}(i) = \tilde{w}(\mathcal{C}^0_{i,1})$, $W^0_{\text{long}}(i) = \tilde{w}(\mathcal{C}^0_{i,m_0})$, $W^1_{\text{short}}(i) = \tilde{w}(\mathcal{C}^1_{i,1})$, *and* $W^1_{\text{long}}(i) = \tilde{w}(\mathcal{C}^1_{i,m_1})$. *For all* $i > 1$, $W^0_{\text{short}}(i) = \max\{W^0_{\text{short}}(i-1), W^1_{\text{long}}(i-1)\}$ *and* $W^1_{\text{short}}(i) = w_i + \max\{W^0_{\text{long}}(i-1) - \alpha, W^1_{\text{short}}(i-1)\}$. *For all* $i \in [m_0, n]$, $W^0_{\text{long}}(i) = W^0_{\text{short}}(i - m_0 + 1)$, *and for all* $i \in [m_1, n]$, $W^1_{\text{long}}(i) = W^1_{\text{short}}(i - m_1 + 1) + \sum_{j=i-m_1+2}^{i} w_i$.

Proof. Straightforward by using the definition. □

Lemma 3 implies a dynamic programming algorithm (referred to as MinLength-Cover), which finds an optimal cover subject to length restrictions. The algorithm runs in $O(n)$ time. The case $\alpha = 0$ is equivalent to the original problem of Fu and Curnow [4], that of finding a segmentation that satisfies the length restrictions.

3.3 A fast algorithm for finding a maximal cover

So far we concentrated on computing maximal covers using Theorem 1 or selecting one cover using linear complexity penalties. It is also possible to calculate maximal covers by employing Theorem 2. The main idea is to find the cover that comprises all runs of positive scores and then produce smaller maximal covers consecutively. Below we develop the idea formally. A segment $[i, j]$ is a *positive run* if $w([i, j]) > 0$ and for all $k \in [i, j]$, $w_k \geq 0$. A segment $[i, j]$ is a *negative run* if $w([i, j]) < 0$ and for all $k \in [i, j]$, $w_k \leq 0$. When not all scores are zero, we can decompose $[1, n]$ into an alternating series of maximal negative and positive runs. Let $\mathcal{T} = (T_1, T_2, \ldots, T_m)$ be the resulting series. Let M be the number of positive runs in $\mathcal{T}$. Clearly, the set $\{T \in \mathcal{T}\colon w(T) > 0\}$ is a maximal M-cover. In fact, M is the cover size until which the score of maximal covers increases.

The sequence $\mathcal{T}$ can be calculated in $O(n)$ time. Applying Theorem 2, we produce maximal covers of size less than M one by one. In every step, we need to identify three consecutive segments T_{i-1}, T_i, T_{i+1} that can be merged at the

expense of the smallest decrease in the cover score. Such a triple is found by selecting i for which the absolute value $|w(T_i)|$ is minimal. Algorithm MaxCover-Fast shown here implements the idea.

Algorithm MaxCover-Fast
Input: w_i scores for $i \in [1, n]$, K cover size
F1 Let $\mathcal{T}$ be the sequence of alternating maximal runs
F2 **for** $M = |\{T \in \mathcal{T}\colon w(T) > 0\}|$ **downto** K **do**
F3 // *at this point* $\mathcal{T} = ([a_1, b_1], [a_2, b_2], \ldots, [a_m, b_m])$ *where* $a_{i+1} = b_i + 1$
F4 Choose $[a_i, b_i]$ from $\mathcal{T}$ with minimum $|w([a_i, b_i])|$, $1 < i < m$
F5 Set $\mathcal{T} \leftarrow \mathcal{T} \cup \{[a_{i-1}, b_{i+1}]\} \setminus \{[a_{i-1}, b_{i-1}], [a_i, b_i], [a_{i+1}, b_{i+1}]\}$
F6 **return** the set $\{T \in \mathcal{T}\colon w(T) > 0\}$

Lemma 4. *Algorithm* MaxCover-Fast *finds a maximal K-cover if not all scores are zero, and it is invoked with a K that is not larger than the number M of maximal positive runs. The algorithm can be implemented in such a way that it terminates in $O(n + M \log M)$ time.*

Proof. (Sketch.) An invariant that implies the correctness is that in Line F4, $\mathcal{T}$ alternates segments with positive and negative scores. In order to see that, notice that $\Big|w([a_i, b_i])\Big| \leq \min\Big\{\Big|w([a_j, b_j])\Big|\colon j = i \pm 1\Big\}$ in Line F5. Thus, $w([a_{i-1}, b_{i+1}])$, $w([a_{i-1}, b_{i-1}])$, and $w([a_{i+1}, b_{i+1}])$ have the same sign. The algorithm's correctness now follows from Theorem 2. A balanced search tree can be augmented to track the segments in $\mathcal{T}$. Elements of $\mathcal{T}$ are stored at the tree leaves, ordered by the absolute values of the scores. In order to avoid selecting the first or the last segment in Line F4, those two segments are stored with scores $\pm\infty$, preserving only their scores' signs. In addition, leaves are equipped with pointers to preceding and succeeding segments. It is thus possible to perform Line F4 in $O(\log M)$ time, to find neighboring segments in $O(1)$ time, and to update $\mathcal{T}$ in Line F5 in $O(\log M)$ time. Hence the algorithm runs in $O(n + M \log M)$ time. □

MaxCover-Fast can be modified to find an optimal cover $\mathcal{C}^*$ for an arbitrary monotone increasing complexity penalty function. Since maximal covers' scores stop increasing at M, $|\mathcal{C}^*| \leq M$. The algorithm has to track the cover score: at each merging operation in Line F5, the score decreases by $|w([a_i, b_i])|$. All maximal covers of size $\leq M$ are inspected, and the one maximizing $\tilde{w}$ is reported at the end. Consequently, the optimal cover can be found in $O(n \log n)$ time.

4 Non-coding RNA genes in AT-rich thermophiles

A frequently used statistic for DNA sequences is the *GC-content*, which is the relative frequency of `G` and `C` in a region. In a recent application, GC-content was used to detect non-coding RNA genes [7, 14] in genomes of thermophile Archaebacteria such as *Methanocaldococcus jannaschii*. The optimal growth temperature of thermophile Prokaryotes strongly correlates with the GC-content of

transfer and ribosomal RNA genes [15–17]. (For the genome-wide GC-content, however, there does not seem to exist a similar dependence [17].) *M. jannaschii* is a prime candidate for identifying RNA genes on GC-content alone, since while the GC-content of the genome is 31%, known RNA genes have a much higher GC-content of 60–70%. Klein *et al.* [7] trained a two-state HMM in which the states modeled GC-poor and GC-rich regions. They computed the most likely state sequence, in order to select a set of GC-rich segments. After filtering out known genes, they selected the segments with a minimum length of 50, which resulted in nine candidate RNA genes, denoted Mj1–Mj9. They validated four of them by showing that they are transcribed. They identified a fifth gene Mj6a, missed by the HMM, based on sequence similarity. Two candidates (Mj5 and Mj8) are less likely to be RNA genes as they overlap with putative protein coding regions. Schattner [14] also used GC-content and other statistics to identify RNA genes in *M. jannaschii*. He used a moving window, within which the log-likelihood was calculated using essentially the same equations as in §2.1.

We tested our algorithms on *M. jannaschii* (1.66 Mbp, GenBank accession `NC_000909.1`). Using Eq. (1), we employed the scores $w_i = -0.66$ if the corresponding nucleotide was `A` or `T`, and $w_i = 0.72$ for `G` or `C`. The scores are based on the genome's overall GC-content, and the 65% GC-content in seven tRNA genes between positions 850000 and 870000. Using MaxCover, we computed maximal k-covers: see Fig. 2. The smallest maximal cover that includes all tRNAs has size $k = 38$. That cover also includes all rRNAs, as well as RNase P RNA and SRP 7S (Signal Recognition Particle) genes. In addition, three novel genes of [7] are also included. The false positive rate can be assessed by the fact that only two intervals overlap with protein-coding genes: Mj5 and Mj8. The maximal 46-cover contains all RNA genes of [7], including Mj6a, not discovered by either the HMM or the sliding windows of [14].

We evaluated different penalty functions r: $r(k) = 2k \log n$ (MDL1) or $r(k) = 2k(\log n - \log(2k))$ (MDL2); Eq. 4b (HMM[1]); and Eq. 3 for significance ($P = 0.1$ and $P = 0.01$). As shown in Fig. 2, the MDL penalties are too severe, and even HMM segmentation stops at $P = 0.01$. There is no need to be very conservative in this case, as the gene candidates identified by the segmentation are further analyzed by different methods. Accordingly, we selected $\alpha = 14$ for the complexity penalty (P-value 0.11 by Eq. (3)), and imposed a minimum segment length of 40. MinLength-Cover finds a 48-cover, which includes all known RNA genes (even Mj6a), five protein-coding genes, and the segment [334439,334485] not identified by either [14] or [7], which is classified as a pseudogene by tRNAscan [18].

We carried out similar experiments with a number of thermophile Prokaryotes. In the maximum likelihood framework of §2.1, one can readily predict the success of gene finding. A linear penalty α set by Eq. (3) can be compared to expected scores of changed segments. A changed segment of length ℓ has expected score $E(\ell) = \ell D$ where D is the relative entropy between the distributions. The threshold $\ell_{\min} = \alpha/D$ thus indicates the minimum detectable gene lengths. By this reasoning, we found that among thermophiles for which whole

[1] If an HMM is used, the $\log \frac{\tau(1\to 1)}{\tau(0\to 0)}$ terms are negligible in Eq. (4a).

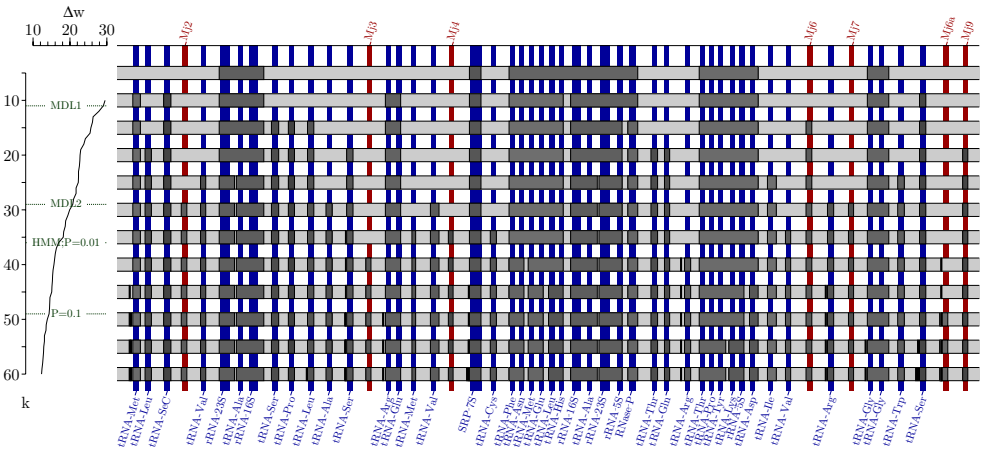

Fig. 2. Segmentation of M. jannaschii. The right-hand side shows maximal k-covers in increments of five. Horizontal grey bands correspond to covers, and darker boxes indicate the segments. Vertical bars show RNA genes with known function (bottom) and those in [7] (top). (Locations are mapped to display coordinates by a piecewise logarithmic function.) The left-hand side plots the score increase $\Delta w(k) = w(\mathcal{C}_k) - w(\mathcal{C}_{k-1})$. The optimal cover for a linear penalty α is the last one with $\Delta w(k) \geq \alpha$.

genome sequences are avaiilable, *N. equitans* [19], *S. tokodaii* [20], *S. solfataricus*, *M. maripaludis* and *P. horikoshii* have low $\ell_{\min}$ values. The analysis of the results is in progress, and is beyond this paper's scope. We summarize here some initial findings. We found that a maximal 58-cover of the *N. equitans* genome (`NC_005213.1`) includes all tRNAs found by tRNAscan [18] (even pseudogenes), all rRNAs, four protein-coding genes, and 12 unannotated segments. Seven of the unannotated segments[2] produce no significant BLAST hits in Genbank. The *S. tokodaii* genome (`NC_003106.2`) contains many repetitive sequences. Its segmentation identifies all annotated RNA genes (including a tRNA that is missed by tRNAscan), and includes 24 transposition-related sequences, and six hypothetical protein-coding genes. There are eight unannotated segments, four of which are repeated more than once in the genome. BLAST finds no similarities between the remaining four[3] segments and known archaeal DNA sequences. We hypothesize that they correspond to non-coding RNA genes. A fifth segment [326016,326321] overlaps in only 11 bp with a hypothetical protein-coding gene. Based on a structural alignment to known RNase P RNA genes in Crenarchaea [21], it is the so far unpublished RNase P RNA gene in *S. tokodaii* (J. W. Brown, personal communication).

[2] The following segments: [221790,221843], [308960,309006], [339411,339562], [427536,427595], [487434,487490], [487645,487722], [488412,488458].

[3] Segments [1322116,1322329] [2020954,2021005], [2089025,2089145], and [2091849,2092094].

5 Discussion

We presented algorithms that calculate optimal covers according to different criteria, in linear or $O(n \log n)$ time for an input of size n. Even a recent review [2] of DNA segmentation methods considered the cover selection problem, based on [4], as one that can only be solved in $O(n^2)$ time. Such a running time may be a serious drawback in the analysis of long DNA sequences.

A related problem, that of finding a maximal *chain* of covers, can also be solved in linear time. A chain of covers is formed by $\mathcal{C}_0, \mathcal{C}_1, \ldots$ where every $\mathcal{C}_k$ is a k-cover, and $\mathcal{C}_k \subset \mathcal{C}_{k+1}$ for all k. A maximal chain of covers $\mathcal{C}_0^*, \mathcal{C}_1^*, \ldots$ is defined recursively: $\mathcal{C}_0^* = \emptyset$, and for every $k > 0$, $\mathcal{C}_k^*$ is a k-cover that has maximum score satisfying $\mathcal{C}_{k-1}^* \subset \mathcal{C}_k^*$. In other words, successive elements are generated using only Case (1) of Theorem 1. Ruzzo and Tompa [6] describe a linear-time algorithm that finds the last $\mathcal{C}_k^*$ in which all segments have positive scores. In the maximal likelihood framework of §2.1, looking for a maximal chain may give unsatisfactory results. Specifically, it may be the case that two changed segments with large scores are separated by a short unchanged segment, and all three get lumped together in one of the covers. Subsequent covers do not change the situation, regardless of the middle segment's score. In Fig. 1, all positive scores are included in $\mathcal{C}_1^*$. Theorem 3 shows that maximal covers may give more sound segmentation results than do maximal chains.

Zhang *et al.* [9] examine the problem of producing pairwise sequence alignments without low-scoring regions. An alignment is a sequence of n columns, each assigned a score. The score of a subalignment, defined by a segment $[a, b] \subseteq [1, n]$ is the sum of its columns scores. Disjoint subalignments thus form a cover. Standard alignment procedures [22] have essentially the same shortcomings as maximal cover chains in that they may include subalignments of arbitrarily low score. In order to avoid such situations, Zhang *et al.* [9] propose that low-scoring regions should be removed from the alignment. In particular, they aim to find a cover $\mathcal{C}$, for which no subsegment of a $S \in \mathcal{C}$ has score less than $-X$ for a threshold $X \geq 0$. They prove that such covers for decreasing values of X form a hierarchy similar to that of maximal covers described by Theorems 1 and 2. They also provide a linear time algorithm implied by the hierarchy that finds such a cover for a given X. In light of Theorem 3, such covers are succinctly characterized by a linear penalty function $r(k) = Xk$. We pointed out the connection between the threshold X and various statistical notions of complexity, as well as the interpretation of the optimal cover as the most likely state sequence in a Markov model. Penalized-Cover offers a simple, efficient alternative to the algorithm of [9] for eliminating low-scoring regions from alignments. MinLength-Cover also provides the option of imposing minimum subalignment lengths.

Acknowledgments. This work has benefited from conversations with Balázs Kégl and Stefan Wolf. I am very grateful to James W. Brown for confirming my identification of the *S. tokodaii* RNase P gene, and to Hans-Georg Müller for reading the manuscript. I found a simpler version of Theorem 1 in collaboration with Réka Szabó: it appeared first, along with the MaxCover algorithm, in my Masters thesis, written under the direction of Gábor Lugosi at the Technical University of Budapest in 1994.

References

1. Bentley, J.: Programming pearls: algorithm design techniques. Comm. ACM **27** (1984) 865–873
2. Braun, J.V., Müller, H.G.: Statistical methods for DNA sequence segmentation. Statist. Sci. **13** (1998) 142–162
3. Karlin, S., Brendel, V.: Chance and significance in protein and DNA analysis. Science **257** (1992) 39–49
4. Fu, Y.X., Curnow, R.N.: Maximum likelihood estimation of multiple change points. Biometrika **77** (1990) 563–573
5. Li, W., Bernaola-Galván, P., Haghighi, F., Grosse, I.: Applications of recursive segmentation to the analysis of DNA sequences. Comput. Chem. **26** (2002) 491–510
6. Ruzzo, W.L., Tompa, M.: A linear time algorithm for finding all maximal scoring subsequences. In: Proc. 7th Intl. Conf. Intelligent Systems in Molecular Biology, AAAI Press (1999) 234–241
7. Klein, R.J., Misulovin, Z., Eddy, S.R.: Noncoding RNA genes identified in AT-rich hyperthermophiles. Proc. Natl. Acad. Sci. USA **99** (2002) 7542–7547
8. Churchill, G.A.: Stochastic models for heterogeneous DNA sequences. Bull. Math. Biol. **51** (1989) 79–94
9. Zhang, Z., Berman, P., Wiehe, T., Miller, W.: Post-processing long pairwise alignments. Bioinformatics **15** (1999) 1012–1019
10. Barron, A., Rissanen, J., Yu, B.: The Minimum Description Length principle in coding and modeling. IEEE Trans. Inform. Theory **44** (1998) 2743–2760
11. Karlin, S., Altschul, S.F.: Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. Proc. Natl. Acad. Sci. USA **87** (1990) 2264—2268
12. Karlin, S., Dembo, A., Kawabata, T.: Statistical composition of high-scoring segments from molecular sequences. Ann. Statist. **18** (1990) 571–581
13. Rabiner, L.R.: A tutorial on Hidden Markov Models and selected applications in speech recognition. Proc. IEEE **77** (1989) 257–286
14. Schattner, P.: Searching for RNA genes using base composition statistics. Nucleic Acids Res. **30** (2002) 2076–2082
15. Galtier, N., Lobry, J.: Relationships between genomic G+C content, RNA secondary structures, and optimal growth temperature in Prokaryotes. J. Mol. Evol. **44** (1997) 632–636
16. Wang, H.C., Hickey, D.A.: Evidence for strong selective constraint acting on the nucleotide composition of 16S ribosomal RNA genes. Nucleic Acids Res. **30** (2002) 2501–2507
17. Bao, Q., et al.: A complete sequence of the T. tengcongensis genome. Genome Res. **12** (2002) 689–700
18. Lowe, T.M., Eddy, S.R.: tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. Nucleic Acids Res. **25** (1997) 955–964
19. Waters, E., et al.: The genome of Nanoarchaeum equitans: insights into early archaeal evolution and derived parasitism. Proc. Natl. Acad. Sci. USA **100** (2003)
20. Kawarabayashi, Y., et al.: Complete genome sequence of an aerobic thermoacidophilic crenarchaeon, Sulfolobus tokodaii strain7. DNA Research **8** (2001) 123–140
21. Brown, J.W.: The ribonuclease P database. Nucleic Acids Res. **27** (1999) 314
22. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. J. Mol. Biol. **147** (1981) 195–197