Maximum-Scoring Segment Sets

Miklós Csűrös

***Abstract*— We examine the problem of finding maximum-scoring sets of disjoint segments 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 maximum-scoring segment sets. The statement leads to fast algorithms for finding such segment sets. We apply our methods to the identification of non-coding RNA genes in thermophiles.**

Key words: segmentation, change point estimation, noncoding RNA, thermophiles

I. 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 segment of maximum score can be found in linear time [2] 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 set of segments. The score of a k-cover $\mathcal{C}$ is the sum of its elements' scores: $w(\mathcal{C}) = \sum_{S \in \mathcal{C}} w(S)$. 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 [3]. Various scoring schemes permit the identification of charge clusters and hydrophobic profiles for proteins [4], determination of isochores in DNA sequences [5], [6], discovery of CpG islands [6], [7], and even gene finding [8]. Different methods include maximum likelihood estimation [5], Hidden Markov Models [8], [9], entropy-based [6], and various "moving window" techniques. Segmentation methods are also used to remove low-scoring regions from sequence alignments [10].

Our key result is Theorem 1 of Section II, 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 III examines maximal cover problems arising in statistical contexts where probabilistic principles guide the choice of scores and cover sizes. Section IV 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 V deals with the problem of identifying GC-rich regions in AT-rich genomes, which often coincide with non-protein-coding RNA genes in thermophiles. Section VI discusses related results and concludes the paper.

Research supported by grants 250391-02 from the Natural Sciences and Engineering Research Council of Canada and 2003-SC-87439 from the Fonds québécois de la recherche sur la nature et les technologies.

A preliminary abstract appeared at WABI 2004 [1].

Départment d'informatique et de recherche opérationnelle, Université de Montréal, C.P. 6128, succ. Centre-Ville, Montréal, Qué. H3C 3J7, Canada.

II. Relationships between consecutive maximal covers

In this section we examine characteristics of maximal covers, and prove that there is a simple recursive relationship between them. Theorem 1 shows that a maximal $(k+1)$-cover can be obtained from any maximal k-cover, either (1) by adding a new segment to it, or (2) by removing the middle of a segment in it. By Theorem 2, the converse is also true: a maximal $(k-1)$-cover can be created from any maximal k-cover by merging two segments, or by removing one. Theorem 2 further implies that *all* maximal covers can be produced by consecutive applications of operations (1) and (2) of Theorem 1. Figure 1 illustrates these relations between successive maximal covers.

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 $\mathcal{C}_{k-1} = \mathcal{C}_k \cup \{[a, b]\} \setminus \{[a, c], [d, b]\}$.

Proof: [*Theorem 1*] For $k = 0$, the theorem holds trivially. Let $k > 0$. Let $\mathcal{C}_k$ be a maximal k-cover. Define $\Delta_1 = \max\{w([a, b]) \colon [a, b] \cap \cup_{S \in \mathcal{C}_k} = \emptyset\}$, and $\Delta_2 = \max\{(w([a, c]) + w([d, b]) - w([a, b])) \colon [a, b] \in \mathcal{C}_k, a \leq c < d \leq b\}$. Let $\Delta = \max\{\Delta_1, \Delta_2\}$. The theorem claims that the score of a maximal $(k+1)$-cover equals $w(\mathcal{C}_k) + \Delta$. (Obviously, a maximal $(k+1)$-cover cannot have score less than $w(\mathcal{C}_k) + \Delta$ since the definitions of Δ_1 and Δ_2 correspond to the set of $(k+1)$-covers that can be produced by applying operation (1) or (2).) For the sake of contradiction, assume that there exists a $(k+1)$-cover $\mathcal{C}'_{k+1}$ with $w(\mathcal{C}'_{k+1}) > w(\mathcal{C}_k) + \Delta$.

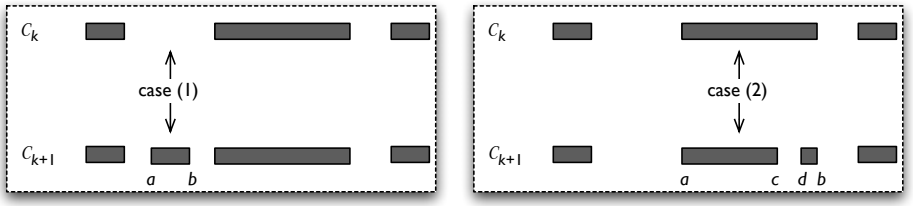

Fig. 1. Relations between maximal covers (Theorems 1 and 2).

We show that then it is possible to construct a k-cover $\mathcal{C}'_k$ with larger score than that of $\mathcal{C}_k$, which contradicts the maximality of $\mathcal{C}_k$.

Since there are more segments in $\mathcal{C}'_{k+1}$ than in $\mathcal{C}_k$, it must be the case that either there exists a segment $S' \in \mathcal{C}'_{k+1}$ which does not overlap any segment of $\mathcal{C}_k$, or that there exists a segment $[a,b] \in \mathcal{C}_k$ which overlaps two consecutive segments $[a',c],[d,b'] \in \mathcal{C}'_{k+1}$. In the first case, the k-cover $\mathcal{C}'_k$ is obtained from $\mathcal{C}'_{k+1}$ by removing S'. Necessarily, $w(S') \leq \Delta_1 \leq \Delta$, and thus

$$w(\mathcal{C}'_k) = w(\mathcal{C}'_{k+1}) - w(S') \geq w(\mathcal{C}'_{k+1}) - \Delta > w(\mathcal{C}_k),$$

a contradiction.

In the second case, $\mathcal{C}'_k$ is produced from $\mathcal{C}'_{k+1}$ by fusing the two segments: $\mathcal{C}'_k = \mathcal{C}'_{k+1} \cup \{[a',b']\} \setminus \{[a',c],[d,b']\}$. Clearly,

$$\begin{aligned} w([a',c]) + w([d,b']) - w([a',b']) \\ = w([a,c]) + w([d,b]) - w([a,b]) \\ \leq \Delta_2 \leq \Delta. \end{aligned}$$

Hence,

$$\begin{aligned} w(\mathcal{C}'_k) &= w(\mathcal{C}'_{k+1}) \\ &\quad - \big(w([a',c]) + w([d,b']) - w([a',b'])\big) \\ &\geq w(\mathcal{C}'_{k+1}) - \Delta > w(\mathcal{C}_k), \end{aligned}$$

again a contradiction. ∎

Proof: [*Theorem 2*] The proof uses the same logic as the previous proof. Consider $k > 1$, otherwise the claim is trivially true. Let $\mathcal{C}_k = \{[a_1,b_1],\ldots,[a_k,b_k]\}$ be a maximal k-cover with $a_1 < a_2 < \cdots < a_k$. Let $\Delta_1 = \min\big\{w([a_i,b_i])\colon i \in [1,k]\big\}$, and let $\Delta_2 = \min\big\{w([a_i,b_i]) + w([a_{i+1},b_{i+1}]) - w([a_i,b_{i+1}])\colon i \in [1,k-1]\big\}$. Define $\Delta = \min\{\Delta_1,\Delta_2\}$. The theorem implies that a maximal $(k-1)$-cover has score $w(\mathcal{C}_k) - \Delta$. For the sake of contradiction, suppose that there exists a $(k-1)$-cover $\mathcal{C}'_{k-1}$ with $w(\mathcal{C}'_{k-1}) > w(\mathcal{C}_k) - \Delta$. We show that then it is possible to construct a k-cover $\mathcal{C}'_k$ with larger score than that of $\mathcal{C}_k$, which contradicts the maximality of $\mathcal{C}_k$.

Since there are more segments in $\mathcal{C}_k$ than in $\mathcal{C}'_{k-1}$, it must be the case that either there exists a segment $S \in \mathcal{C}_k$ which does not intersect any segment of $\mathcal{C}'_{k-1}$, or that there exists a segment $[a',b'] \in \mathcal{C}'_{k-1}$ which overlaps two consecutive segments $[a_i,b_i],[a_{i+1},b_{i+1}]$. In the first case, the k-cover $\mathcal{C}'_k$ is obtained from $\mathcal{C}'_{k-1}$ by adding S. Necessarily, $w(S) \geq \Delta_1 \geq \Delta$, and thus

$$w(\mathcal{C}'_k) = w(\mathcal{C}'_{k-1}) + w(S) \geq w(\mathcal{C}'_{k-1}) + \Delta > w(\mathcal{C}_k),$$

a contradiction.

In the second case, $\mathcal{C}'_k$ is produced from $\mathcal{C}'_{k-1}$ by splitting the segment $[a',b']$: $\mathcal{C}'_k = \mathcal{C}'_{k-1} \cup \{[a',b_i],[a_{i+1},b']\} \setminus \{[a',b']\}$. Clearly,

$$\begin{aligned} &w([a',b_i]) + w([a_{i+1},b']) - w([a',b']) \\ &= w([a_i,b_i]) + w([a_{i+1},b_{i+1}]) - w([a_i,b_{i+1}]) \geq \Delta_2 \geq \Delta. \end{aligned}$$

Hence,

$$\begin{aligned} w(\mathcal{C}'_k) &= w(\mathcal{C}'_{k-1}) \\ &\quad + \big(w([a',b_i]) + w([a_{i+1},b']) - w([a',b'])\big) \\ &\geq w(\mathcal{C}'_{k-1}) + \Delta > w(\mathcal{C}_k), \end{aligned}$$

again a contradiction. ∎

The theorems have some important consequences. First, Theorem 1 implies a simple algorithm for calculating successive maximal covers, which we describe in §IV-A. Secondly, Theorem 2 allows for a fast algorithm that finds a maximal cover in $O(n \log n)$ time, described in §IV-C. Finally, Corollary 1 below shows that the score of maximal k-covers is a concave function of k.

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

$$w(\mathcal{C}_{k+1}) - w(\mathcal{C}_k) \leq w(\mathcal{C}_k) - w(\mathcal{C}_{k-1}). \tag{1}$$

Proof: Without loss of generality, we can assume that the covers satisfy the relationships described by Theorem 1. Let $\mathcal{C}_k = \{[a_1,b_1],\ldots,[a_k,b_k]\}$ with $a_1 \leq b_1 < a_2 \leq b_2 < \cdots < a_k \leq b_k$.

Case 1.1: There exist $a,b \in [1,n]$ and $i \in [1,k]$ such that $\mathcal{C}_{k-1} = \mathcal{C}_k \setminus \{[a_i,b_i]\}$ and $\mathcal{C}_{k+1} = \mathcal{C}_k \cup \{[a,b]\}$. We need to prove that $w([a_i,b_i]) \leq w([a,b])$. Since $\mathcal{C}_{k+1}$ contains both segments, they may not overlap. Hence Eq. (1) must hold, otherwise $\mathcal{C}_{k-1} \cup \{[a,b]\}$ is a better k-cover than $\mathcal{C}_k$.

Case 1.2: There exist $i,j \in [1,k]$ and $b,a \in [a_j,b_j]$ such that $\mathcal{C}_{k-1} = \mathcal{C}_k \setminus \{[a_i,b_i]\}$ and $\mathcal{C}_{k+1} = \mathcal{C}_k \cup \{[a_j,b],[a,b_j]\} \setminus \{[a_j,b_j]\}$. Clearly, if $i \neq j$ then Eq. (1) must be satisfied otherwise $\mathcal{C}_{k+1} \setminus \{[a_i,b_i]\}$ is a better k-cover than $\mathcal{C}_k$. If $i = j$, then $w([a_i,b]) + w([a,b_i]) - w([a_i,b_i]) \leq w([a_i,b_i])$ must hold: otherwise, $w([a_i,b]) > w([a_i,b_i])$ or $w([a,b_i]) > w([a_i,b_i])$ and thus $\mathcal{C}_{k-1} \cup \{[a_i,b]\}$ or $\mathcal{C}_{k-1} \cup \{[a,b_i]\}$ is a higher-scoring k-cover than $\mathcal{C}_k$.

Case 2.1: There exist $i \in [1,k-1]$ and $a,b \in [1,n]$ such that $\mathcal{C}_{k-1} = \mathcal{C}_k \cup \{[a_i,b_{i+1}]\} \setminus \{[a_i,b_i],[a_{i+1},b_{i+1}]\}$ and $\mathcal{C}_{k+1} = \mathcal{C}_k \cup \{[a,b]\}$. If $[a,b]$ does not overlap $[a_i,b_{i+1}]$, then Eq. (1) must hold, or else $\mathcal{C}_{k-1} \cup \{[a,b]\}$ is a better k-cover than $\mathcal{C}_k$. Suppose that $[a,b]$ intersects $[a_i,b_{i+1}]$ and thus

$b_i < a \le b < a_{i+1}$. We need to prove that

$$w([a,b]) \le w([a_i,b_i]) + w([a_{i+1},b_{i+1}]) - w([a,b_{i+1}]). \quad (*)$$

Now, $w([a_i,b]) \le w([a_i,b_i])$ and $w([a,b_{i+1}]) \le w([a_{i+1},b_{i+1}])$, otherwise $[a_i,b_i]$ can be replaced by $[a_i,b]$ or $[a_{i+1},b_{i+1}]$ can be replaced by $[a,b_{i+1}]$ in $\mathcal{C}_k$, resulting in a higher score. By adding the two inequalities, we get $w([a_i,b]) + w([a,b_{i+1}]) \le w([a_i,b_i]) + w([a_{i+1},b_{i+1}])$. The left-hand side equals $w([a_i,b_{i+1}]) + w([a,b])$, and thus $w([a_i,b_{i+1}]) + w([a,b]) \le w([a_i,b_i]) + w([a_{i+1},b_{i+1}])$, which is tantamount to (*).

Case 2.2: There exist $i \in [1,k-1]$, $j \in [1,k]$, and $b,a \in [a_j,b_j]$ such that $\mathcal{C}_{k-1} = \mathcal{C}_k \cup \{[a_i,b_{i+1}]\} \setminus \{[a_i,b_i],[a_{i+1},b_{i+1}]\}$ and $\mathcal{C}_{k+1} = \mathcal{C}_k \cup \{[a_j,b],[a,b_j]\} \setminus \{[a_j,b_j]\}$. Similarly to Case 1.1, the removal of the subsegments can be carried out in any order, and thus Eq (1) must hold. ∎

REMARK. Kearns *et al.* [11] consider a problem in a machine learning context, which turns out to be a special case of finding maximal covers. They consider empirical error minimization for a learning sample with one-dimensional observations, which is equivalent to finding a maximal cover when all scores equal $+1$ or -1. Theorem 2 for this special case, and an implied algorithm similar to that of §IV-C, appear in [11]; our proof is an adaptation of theirs to general scores.

III. SCORES BASED ON PROBABILISTIC MODELS

In this section we consider the case when scores are based on probabilistic models. First, §III-A shows that in a statistical context, maximal likelihood estimation of change points [5], [12] leads to a maximal cover problem. Secondly, §III-B examines the problem of cover size selection, based on the minimum description length principle, and other statistical notions of complexity. We continue by arguing for a cover size selection method based on statistical significance in §III-C. Finally, §III-D shows how Hidden Markov Model-based segmentation fits into the framework of maximal covers.

A. Maximum likelihood estimation of segments

Let $X_1,\dots,X_n$ be a sequence of independent random letters from an alphabet $\Sigma = \{\sigma_1,\dots,\sigma_r\}$. 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 $\mathbf{z} = (z_1,\dots,z_n)$ be the indicator vector for $\mathcal{C}$. The likelihood function is $f(\mathbf{x} \mid \mathbf{z},\mathbf{p},\mathbf{q}) = \prod_{i=1}^n (p(x_i))^{1-z_i}(q(x_i))^{z_i}$. Define

$$w_i = \log(q(x_i)) - \log(p(x_i)). \quad (2)$$

(Throughout the paper, log denotes natural logarithm.) The log-likelihood can be written as

$$\log f(\mathbf{x} \mid \mathbf{z},\mathbf{p},\mathbf{q}) = \sum_{i=1}^n \log p(x_i) + \sum_{i=1}^n w_i z_i.$$

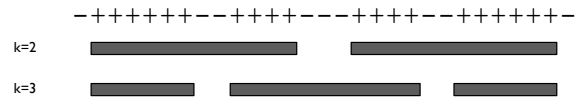

Fig. 2. Maximal k-covers with minimum segment lengths $m_1 = 6$ and $m_0 = 1$. A '+' denotes $w_i = 1$ and '-' denotes $w_i = -1$. An equivalent scoring scheme is realized when $\Sigma = \{0,1\}$, and $p(0) = 1 - q(0)$ in Eq. (2).

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. (2).

Notice that the framework is easily extended to infinite alphabets, or even to continuous distributions.

Fu and Curnow [5] 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. They aim to maximize the LLR by using scores defined by Equation (2), and by finding a k-cover $\{[a_1,b_1],\dots,[a_k,b_k]\}$ for which $\forall i \in [1,k]\colon b_i - a_i + 1 \ge m_1$, $\forall i \in [2,k]\colon a_i - b_{i-1} + 1 \ge m_0$, and either $a_1 = 1$ or $a_1 > m_0$, and either $b_k = n$ or $n - b_k \ge m_0$. Fu and Curnow state a theorem (with an incorrect 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 2 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.

B. 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 minimum description length (MDL). According to the MDL principle [13], [14], 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 $b(0,\sigma) = -\log_2 p(\sigma)$ and $b(1,\sigma) = -\log_2 q(\sigma)$. 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 $\binom{n+1}{2k}$ possible k-covers in which segments do not abut. (If $\mathcal{C} = \{[a_1,b_1],\ldots,[a_k,b_k]\}$ with $b_i < a_{i-1}$ for all $i = 1,\ldots,k-1$, then there is a one-to-one correspondence between combinations of $2k$ points from $[1,n+1]$ and selections of $\{a_1, b_1+1, a_2, b_2+1,\ldots,a_k,b_k+1\}$.) When $k = o(n)$ and $n \gg 1$, a k-cover can be encoded in $\log_2\binom{n+1}{2k} \approx 2k\log_2 n - 2k\log_2(2k)$ bits. The corresponding penalty equals $r(k) = 2k(\log n - \log(2k))$.

Other MDL-related penalties such as Akaike's AIC [15] or the Bayesian Information Criterion (BIC) of Schwarz [16] lead to similar penalty functions that are linear in the cover size. For a model of dimension d, AIC recommends using d, and BIC recommends using $\frac{d}{2}\log n$ as penalty. In our case, the model dimension is $d = 2k$, the number of parameters needed to specify a k-cover. Notice that BIC gives half as large of a penalty as MDL does.

C. A penalty based on statistical significance

As an alternative to the MDL approach, a penalty can be defined using 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 [17]–[19]. It is known [17], [18] that $w^{(1)} \to \log n$ almost surely as $n \to \infty$, and that for all x,

$$\lim_{n\to\infty} \mathbb{P}\{w^{(1)} - \log n \le x\} = \exp\bigl(-Ce^{-x}\bigr), \qquad (3)$$

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 $\mathcal{C}$: every changed segment in $\mathcal{C}$ scores at least α, no subsegment of an unchanged segment scores above α, and every unchanged segment attains a score of at most $(-\alpha)$, and no subsegment of a changed segment scores below $(-\alpha)$.

Theorem 3: Fix $\alpha \ge 0$, 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 $i \in [1,k-1]$, $b_i + 1 < a_{i+1}$, and $w([b_i+1, a_{i+1}-1]) \le -\alpha$. 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 $a_{k+1} = n+1$.

A consequence of Theorem 3 is that by setting $\alpha = x + \log n$ with an appropriately chosen x we can ensure that every changed segment has significant statistical support, and that maximal set of such segments is selected. In particular, Eq. (3) implies that for large n, $\mathcal{C}$ is non-empty with probability $1 - \exp(-Cne^{-\alpha})$ under the null hypothesis. Accordingly, for a given $0 < p < 1$, we can use

$$\alpha \ge \log n + \log\frac{C}{-\log(1-p)} \approx \log n + \log\frac{C}{p}, \qquad (4)$$

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.

Proof: [*Theorem 3*] If there exists $S \in \mathcal{C}$ for which $w(S) < \alpha$ then by removing that segment from $\mathcal{C}$, we obtain a $(k-1)$-cover $\mathcal{C}'$ with $\tilde{w}(\mathcal{C}') > \tilde{w}(\mathcal{C})$, which is a contradiction. If there exists $b_i+1 = a_{i+1}$ or $w([b_i+1, a_{i+1}-1]) > -\alpha$, then by replacing $[a_i,b_i]$ and $[a_{i+1},b_{i+1}]$ with $[a_i,b_{i+1}]$, we obtain a $(k-1)$-cover that has larger penalized score $\tilde{w}$ than $\tilde{w}(\mathcal{C})$.

If for some i there exist $a_i < a \le b < b_i$ with $w([a,b]) < -\alpha$, then the cover $\mathcal{C}' = \mathcal{C}\cup\{[a_i,a-1],[b+1,b_i]\}\setminus\{[a_i,b_i]\}$ is a $(k+1)$-cover with score $w(\mathcal{C}') > w(\mathcal{C})+\alpha$, and thus $\tilde{w}(\mathcal{C}') > \tilde{w}(\mathcal{C})$, which is a contradiction. If for some i there exist $b_i < a \le b < a_{i+1}$ with $w([a,b]) > \alpha$, then the cover $\mathcal{C}' = \mathcal{C}\cup\{[a,b]\}$ is a $(k+1)$-cover with score $w(\mathcal{C}') > w(\mathcal{C})+\alpha$, which is again a contradiction. ∎

D. 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], [9]. 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

$$f(\mathbf{x} \mid \mathbf{z}) = \pi(z_1)\Bigl(\prod_{i=1}^{n}(p(x_i))^{1-z_i}(q(x_i))^{z_i}\Bigr)\Bigl(\prod_{i=2}^{n}\tau(z_{i-1}\to z_i)\Bigr),$$

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 [20], but formulating it as a maximal cover problem enables us to consider further variations with restrictions on the number of state changes (§IV-A) or on state durations (§IV-B). The LLR of a state sequence $\mathbf{z}$ (viewed as an 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

$$w_i = \log \frac{q(\sigma)}{p(\sigma)} + \log \frac{\tau(1 \to 1)}{\tau(0 \to 0)} + \delta_i; \tag{5a}$$

$$\alpha = -\log \frac{\tau(0 \to 1)}{\tau(0 \to 0)} - \log \frac{\tau(1 \to 0)}{\tau(0 \to 0)} + \log \frac{\tau(1 \to 1)}{\tau(0 \to 0)}, \tag{5b}$$

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.

At first sight, it may seem that the scores and the complexity penalty do not depend on n. In practice, however, the state transition parameters $\boldsymbol{\tau}$ often incorporate such a dependence. For example, Viterbi training [20] alternates an expectation and a minimization step until convergence is reached. First, $\mathbf{z}$ is calculated given the model parameters, then the parameters are readjusted given $\mathbf{z}$. Let $\ell = \sum_{i=1}^{n} z_i$ be the total length of changed segments, and let k be the number of changed segments when a fix point is reached in the optimization. For simplicity, assume that $z_1 = z_n = 0$. The state transition probabilities are estimated as $\tau(0 \to 1) = \frac{k}{n-\ell-1}$, $\tau(0 \to 0) = \frac{n-\ell-k-1}{n-\ell-1}$, $\tau(1 \to 0) = \frac{k}{\ell}$, and $\tau(1 \to 1) = \frac{\ell-k}{k}$. If $k \ll \ell \ll n$, then the complexity penalty is $\alpha \approx \log n + \log \ell - 2 \log k$. The formula depends on the total length of segments, and thus does not fit our framework in which penalties depend solely on the number of segments in the cover. Nevertheless, it shows that HMM penalties fall between MDL ($\alpha = 2 \log n$) and BIC penalties ($\alpha = \log n$).

IV. Algorithms

A. 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, shown in Figure 3, which is an adaptation of Jon Bentley's classic algorithm [2]. (In fact, Bentley credits Joseph Kadane of CMU with the design.)

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: First we prove the bound on the running time. Line C1 initializes $(z_1, \ldots, z_n)$ in $O(n)$ time. The loop body C5–C8 is executed in $O(1)$ time for every $k = 1, \ldots, K$ and $i = 1, \ldots, n$. The **else** branch of Line C9 is executed in $O(n)$ time for every k.

Concerning the algorithm's correctness, in the trivial case when $w_i \le 0$ for all i, the algorithm returns $\mathcal{C} = \emptyset$ correctly. We prove the lemma for non-trivial cases by induction. The induction hypothesis is that $(z_1, \ldots, z_n)$ correspond to a maximal $(k-1)$-cover $\mathcal{C}$ for a $k \in [1, K-1]$ at the beginning of the loop on k in Line C3.

We start with a few observations on how the algorithm works. Line C5 ensures that $[i_0, i]$ is a viable segment in Lines C6–C8 in the sense that $[i_0, i]$ is entirely within a segment of $\mathcal{C}$ or it does not intersect any segment of $\mathcal{C}$. Lines C5 and C6 track the score of $[i_0, i]$, so that in Line C6, $w = w([i_0, i])$. The segment S can be updated in Line C8 only if $|w| > 0$. Consequently, if the algorithm does not terminate in Line C9, then it creates a k-cover.

Now we prove that the algorithm creates a maximal k-cover. Define the set of segments $\mathcal{A}^+$ that do not intersect any element of $\mathcal{C}$. Let $w^+ = \max\{w(T) \colon T \in \mathcal{A}^+\}$. Similarly, let $\mathcal{A}^-$ be the set of subsegments of the segments in $\mathcal{C}$, and let $w^- = \max\{-w(T) \colon T \in \mathcal{A}^-\}$. Clearly, w^+ is the maximum score increase that can be achieved by adding a segment, and w^- is the maximum score increase that can be achieved by splitting a segment. We claim that if $w^* = \max\{w^+, w^-\} > 0$, then $w_{\max} = w(S) = w^*$ in Line C9.

Let $[a_0, b_0] \in \mathcal{A}^+$ be the segment for which $w([a_0, b_0]) = w^+$, b_0 is minimal, and there does not exist $[a, b_0]$ with $a > a_0$ and $w([a, b_0]) = w^+$. Similarly, let $[a_1, b_1] \in \mathcal{A}^-$ be the segment for which $w([a_1, b_1]) = -w^-$, b_1 is minimal, and there does not exist $[a, b_1]$ with $a > a_1$ and $w([a, b_1]) = -w^-$. We show that $S = [a_0, b_0]$ if $w^+ = w^-$ and $b_0 < b_1$, or if $w^+ > w^-$. It is clear that if $i_0 = a_0$ and $i = b_0$ in Line C8, then S is set to $[a_0, b_0]$, $w_{\max}$ is set to w^+ and their values do not change anymore until finishing with the loop on i. Hence we have to show that $i_0 = a_0$ when $i = b_0$. When $i = a_0$, $i_0 \le i$ after Line C5. In fact, we must have $i_0 = a_0$, or else $w[i_0, a_0 - 1] > 0$ and the segment $[i_0, b_0]$ would have a score $w([i_0, b_0]) = w([i_0, a_0 - 1]) + w([a_0, b_0]) > w^+$. Moreover, i_0 cannot change while $a_0 < i \le b_0$ since then $w([a_0, i_0]) \le 0$ right before i_0 is updated the first time in Line C5, and thus when $i = b_0$, we have $w([i_0, i]) \ge w^+$ contradicting the definitions of a_0, b_0.

One can prove in an analogous manner that $S = [a_1, b_1]$ if $w^+ = w^-$ and $b_1 < b_0$, or if $w^- > w^+$. Therefore, $\mathcal{C}$ is updated in Line C9 correctly for every k if there is a k-cover that has a larger score than the maximal $(k-1)$-cover. By Corollary 1, we can stop the first time that there does not exist such a cover. ∎

B. Algorithms for linear complexity penalties

Suppose that we want to find a 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 §III-A sets $\alpha = 2 \log n$; the statistical significance framework (setting α by Eq. (4)), and the HMM approach of §III-D 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

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** $(z_1, \ldots, z_n)$

Fig. 3. Algorithm MAXCOVER.

maximum of $\tilde{w}$ for covers of $[1,i]$ which do include i. The following lemma shows the recursions for calculating $W^j(i)$.

Lemma 2: For all $i > 1$,

$$W^0(i) = \max\{W^0(i-1), W^1(i-1)\};$$
$$W^1(i) = w_i + \max\{W^0(i-1) - \alpha, W^1(i-1)\}.$$

Proof: Let $i > 1$ be arbitrary and consider the covers realizing $W^0(i-1)$ and $W^1(i-1)$. Since both of them are covers of $[1,i]$ and do not include i, $W^0(i) \ge \max\{W^0(i-1), W^1(i-1)\}$. However, strict inequality is not possible since the cover realizing $W^0(i)$ is a cover of $[1, i-1]$, and thus its penalized score cannot be larger than the maximum of $\tilde{w}$ among covers of $[1, i-1]$. In a similar spirit, consider the covers of $[1,i]$ obtained by adding $[i,i]$ to a cover realizing $W^0(i-1)$, and by extending the last interval of a cover realizing $W^1(i-1)$. Since their penalized scores equal $(w_i + W^0(i-1) - \alpha)$ and $(w_i + W^1(i-1))$, respectively, $W^1(i) \ge w_i + \max\{W^0(i-1) - \alpha, W^1(i-1)\}$. Again, strict inequality is not possible since one can remove i from the cover of $[1,i]$ that realizes $W^1(i)$ and obtain a cover of $[1, i-1]$ that way. ∎

The lemma implies a dynamic programming algorithm that runs in $O(n)$ time. In case of the two-state HMM, the algorithm is equivalent to the Viterbi algorithm [20]. 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. In order to arrive at a solution by dynamic programming, we consider covers of $[1,i]$ in increasing order of i. In contrast to Lemma 2, we need to track not only whether i is included, but also whether the last segment satisfies the minimum length requirements. As a consequence, Lemma 3 below gives recursions for four variables per prefix.

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[1]: $\mathcal{C}^0_{i,m}$ 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. The following lemma shows the recursions for calculating the weights of these covers.

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 $W^1_{\text{long}}(i) = \tilde{w}(\mathcal{C}^1_{i,m_1})$. Then $W^0_{\text{short}}(1) = 0$, $W^1_{\text{short}}(1) = w_1 - \alpha$, and the following recursions hold.

$$W^0_{\text{short}}(i) = \max\{W^0_{\text{short}}(i-1), W^1_{\text{long}}(i-1)\};$$
$$W^1_{\text{short}}(i) = w_i + \max\{W^0_{\text{long}}(i-1) - \alpha, W^1_{\text{short}}(i-1)\};$$
$$W^0_{\text{long}}(i) = W^0_{\text{short}}(i - m_0 + 1);$$
$$W^1_{\text{long}}(i) = W^1_{\text{short}}(i - m_1 + 1) + \sum_{j=i-m_1+2}^{i} w_j.$$

Lemma 3 implies a dynamic programming algorithm (see Fig. 4), 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 [5], that of finding a segmentation that satisfies the length restrictions. The case $m_0 = m_1 = 1$ is the problem of finding optimal complexity-penalized covers, and thus $W^0_{\text{long}}(i) = W^0_{\text{short}}(i), W^1_{\text{long}}(i) = W^1_{\text{short}}(i)$.

Proof: [*Lemma 3*] For simplicity, we assume the uniqueness of the covers $\mathcal{C}^j_{i,m}$. The cover $\mathcal{C}^0_{i,1}$ either includes $(i-1)$, and thus $\mathcal{C}^0_{i,1} = \mathcal{C}^1_{i-1,m_1}$, otherwise $\mathcal{C}^0_{i,1} = \mathcal{C}^0_{i-1,1}$. If the cover $\mathcal{C}^1_{i,1}$ includes $(i-1)$, then $\mathcal{C}^1_{i,1}$ is obtained by extending the last segment of $\mathcal{C}^1_{i-1,1}$. Otherwise, $\mathcal{C}^1_{i,1} = \mathcal{C}^0_{i-1,m_0} \cup \{[i,i]\}$.

It is clear that $\mathcal{C}^0_{i,m} = \mathcal{C}^0_{i-1,m-1} = \cdots = \mathcal{C}^0_{i-m+1,1}$ when $i, m > 1$. The cover $\mathcal{C}^1_{i,m}$ is obtained from $\mathcal{C}^1_{i-m+1,1}$ by extending the last segment with $[i-m+2, i]$. ∎

C. 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 $k \in [i,j]$, $w_k \ge 0$. A segment $[i,j]$ is a *negative run* if $w([i,j]) < 0$ and for all $k \in [i,j]$, $w_k \le 0$. When not all scores are zero, we can decompose $[1,n]$ into an alternating

[1]More precisely, $\mathcal{C}^j_{i,m}$ satisfy the following conditions. Either $\mathcal{C}^0_{i,m} = \emptyset$, or $\mathcal{C}^0_{i,m} = \{[a_1,b_1], \ldots, [a_k,b_k]\}$, where $0 < a_1 \le b_1 < a_2 \le b_2 < \cdots < a_k \le b_k \le i-m$, $\forall t \in [1,k]$: $b_t \ge a_t + m_1 - 1$, $\forall t \in [2,k]$: $a_t > b_{t-1} + m_0$, and $a_1 = 1$ or $a_1 > m_0$. Either $\mathcal{C}^1_{i,m} = \emptyset$, or $\mathcal{C}^1_{i,m} = \{[a_1,b_1], \ldots, [a_k,b_k]\}$, where $0 < a_1 \le b_1 < a_2 \le b_2 < \cdots < a_k \le b_k = i$, $\forall t \in [1,k-1]$: $b_t \ge a_t + m_1 - 1$, $a_k \le b_k - m + 1$, $\forall t \in [2,k]$: $a_t > b_{t-1} + m_0$, and $a_1 = 1$ or $a_1 > m_0$.

Algorithm MINLENGTH-COVER
Input: w_i scores for $i \in [1,n]$; $\alpha > 0$ complexity penalty; $m_0, m_1 \le n$ minimum segment lengths
Output: z_i indicators for a cover $\mathcal{C}$ that maximizes $w(\mathcal{C}) - \alpha(|\mathcal{C}|)$

L1 Initialize $t_0(i), t_1(i) \leftarrow$ null for $i = 2, \ldots, n$ // *(for traceback)*
L2 $U_0(1) \leftarrow 0$; $U_1(1) \leftarrow w_1 - \alpha$ // *($U_j = W^j_{\text{short}}$ in Lemma 3)*
L3 **if** $m_0 > 1$ **then** $V_0(1) \leftarrow -\infty$ **else** $V_0(1) \leftarrow U_0(1)$ // *($V_j = W^j_{\text{long}}$ in Lemma 3)*
L4 **if** $m_1 > 1$ **then** $V_1(1) \leftarrow -\infty$ **else** $V_1(1) \leftarrow U_1(1)$
L5 **if** $m_1 > 1$ **then** $s \leftarrow w_1$ **else** $s \leftarrow 0$ // *if $m_1 = 1$ then $s = 0$ always*
L6 **for** $i = 2, \ldots, n$ **do**
L7 $\quad s \leftarrow s + w_i$
L8 $\quad$ **if** $i \ge m_1$ **then** $s \leftarrow s - w_{i-m_1+1}$ // *($s = \sum_{j=i-m_1+2}^{i} w_j$)*
L9 $\quad$ **if** $U_0(i-1) > V_1(i-1)$ **then** $U_0(i) \leftarrow U_0(i-1)$; $t_0(i) \leftarrow 0$ **else** $U_0(i) \leftarrow V_1(i-1)$; $t_0(i) \leftarrow 1$
L10 $\quad$ **if** $V_0(i-1) - \alpha > U_1(i-1)$ **then** $U_1(i) \leftarrow w_i + V_0(i-1) - \alpha$; $t_1(i) \leftarrow 0$ **else** $U_1(i) \leftarrow w_i + U_1(i-1)$; $t_1(i) \leftarrow 1$
L11 $\quad$ **if** $i \ge m_0$ **then** $V_0(i) \leftarrow U_0(i - m_0 + 1)$ **else** $V_0(i) \leftarrow -\infty$
L12 $\quad$ **if** $i \ge m_1$ **then** $V_1(i) \leftarrow U_1(i - m_1 + 1) + s$ **else** $V_1(i) \leftarrow -\infty$
L13 **end for**
L14 **if** $V_0(n) > V_1(n)$ **then** $z_n \leftarrow 0$; $m \leftarrow m_0$ **else** $z_n \leftarrow 1$; $m \leftarrow m_1$
L15 **for** $i \leftarrow n-1, n-2, \ldots, 1$ **do**
L16 $\quad$ **if** $m > 1$ **then** $z_i \leftarrow z_{i+1}$; $m \leftarrow m - 1$
L17 $\quad$ **else if** $m = 1$ and $z_{i+1} = 0$ **then**
L18 $\quad\quad$ **if** $t_0(i+1) = 0$ **then** $z_i \leftarrow 0$ **else** $z_i \leftarrow 1$, $m \leftarrow m_1$
L19 $\quad$ **else if** $m = 1$ and $z_{i+1} = 1$ **then**
L20 $\quad\quad$ **if** $t_1(i+1) = 1$ **then** $z_i \leftarrow 1$ **else** $z_i \leftarrow 0$, $m \leftarrow m_0$
L21 **end for**
L22 **return** $(z_1, \ldots, z_n)$

Fig. 4. Algorithm MINLENGTH-COVER

series of 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 $\mathcal{T}$: $M = \lceil m/2 \rceil$ or $M = \lfloor m/2 \rfloor$. 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. After that point, maximal covers have the same score as long as the positive runs can be split, and we arrive to the cover $\{[i] \colon w_i \ge 0\}$. Further increases in the cover size decrease the score, since negative scores need to be included.

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 in Fig. 5 implements the idea.

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.

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, $|\mathcal{C}^*| \le M$. 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 $\le M$ are inspected, and the one maximizing $\tilde{w}$ is reported at the end. Consequently, the optimal cover can be found in $O(n + M \log M)$ time.

It is worth pointing out, that $M = \Theta(n)$ is not unlikely, in which case the running time is $O(n \log n)$. For instance, consider the maximum likelihood framework of §III-A and assume that $p(\sigma) \ne q(\sigma)$ for all letters $\sigma \in \Sigma$, and thus all scores are non-zero. If all letters are drawn from the unchanged distribution $\mathbf{p}$, then the expected value of M is $(1 + 2(n-1)\mu(1-\mu))$ where $\mu = \sum_{\sigma \colon p(\sigma) > q(\sigma)} p(\sigma)$ is the probability of a negative score in an arbitrary position. In practice, it is likely that $\mu(1-\mu)$ is not negligible and thus $M = \Theta(n)$ with high probability. In such a case MAXCOVER-FAST may not enjoy much advantage over MAXCOVER. In the examples of bacterial genome segmentations in Section V, n is in the range of several hundreds of thousands to a few millions, and M/n is typically 40–45%. At the same time, only up to a few hundred segments are interesting. A practical intermediate solution is to use one of the dynamic programming algorithms of §IV-B with a fairly permissible penalty to identify a maximal cover, and then use that cover as the starting point $\mathcal{T}$ in Line F1.

Proof: [*Lemma 4*] 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 $\Big|w([a_i, b_i])\Big| \le \min\Big\{\Big|w([a_j, b_j])\Big| \colon j = i \pm 1\Big\}$ 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

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 **while** $|\{T \in \mathcal{T}\colon w(T) > 0\}| > K$ **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 $\{T \in \mathcal{T}\colon w(T) > 0\}$

Fig. 5. Algorithm MAXCOVER-FAST

in $O(n + M \log M)$ time. ■

V. 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` (guanine) and `C` (cytosine) in a region. In a recent application, GC-content was used to detect non-coding RNA [21] genes in genomes of thermophile Archaebacteria such as *Methanocaldococcus jannaschii* [8], [22]. The GC-content of transfer and ribosomal RNA genes strongly correlates with the optimal growth temperature of thermophile Prokaryotes [23]–[26]. The linear trend is broken at lower growth temperatures (around 60°C) [27], and there does not seem to exist a similar dependence for the genome-wide GC-content [26]. *M. jannaschii* is a prime candidate for identifying RNA genes by 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.* [8] 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 [22] 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 §III-A. In each position, starting with a window of 25 nucleotides, the window was extended as long as its score surpassed a pre-defined threshold, up to a length of 100 characters. An extended window was reported as a changed segment, if it reached the length of 40 nucleotides. This approach identified 19 candidates, denoted cnr1–cnr19. Four of these candidates correspond to those experimentally verified by [8], and two additional ones are also found by the HMM screen. One candidate (cnr10) is the RNase P RNA gene [28] of *M. jannaschii*.

We tested our algorithms on *M. jannaschii* (1.66 Mbp, GenBank accession `NC_000909.1`). Using Eq. (2), we employed the scores $w_i = -0.66$ if the corresponding nucleotide was `A` (adenine), `T` (thymine), or `W` (weak), and $w_i = 0.72$ for `G`, `C`, or `S` (strong). (An additional twelve positions with ambiguous characters had score 0.) 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. This latter set was also used by [22] as the basis for the parameter settings. Using MAXCOVER, we computed maximal k-covers: see Fig. 6. 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 [8] 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 [8], including Mj6a, not discovered by either the HMM or the sliding windows of [22]. The 46-cover intersects only one more protein-coding gene than the 38-cover.

We evaluated different penalty functions r: $r(k) = 2k \log n$ (MDL1) or $r(k) = 2k(\log n - \log(2k))$ (MDL2); Eq. 5b (HMM[2]); and Eq. 4 for significance ($P = 0.1$ and $P = 0.01$). As shown in Fig. 6, 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. (4)), 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 [22] or [8], which is classified as a pseudogene by tRNAscan [29]. Eight candidates of [22] are not in the cover, as they score below 14 (three of them have a score less than twelve, corresponding to a P-value of at least 0.6).

We carried out similar experiments with a number of thermophile Prokaryotes. In the maximum likelihood framework of §III-A, one can readily predict the success of gene finding. A linear penalty α set by Eq. (4) can be compared to expected scores of changed segments. A changed segment of length ℓ has expected score $E(\ell) = \ell D$ where $D = \sum_{\sigma \in \Sigma} q(\Sigma) \log \frac{q(\sigma)}{p(\sigma)}$ is the relative entropy between the distributions. The threshold $\ell_{\min} = \alpha / D$ thus indicates the minimum detectable gene length. By this reasoning, we found that among thermophiles for which whole genome sequences are available, *N. equitans* [30], *S. tokodaii* [31], *S. solfataricus*, *M. maripaludis* and *P. horikoshii* have low $\ell_{\min}$ values (see Fig. 7), allowing for the detection of transfer RNAs and longer RNA genes.

We summarize here some initial findings. We found

[2]If an HMM is used, the $\log \frac{\tau(1 \to 1)}{\tau(0 \to 0)}$ terms are negligible in Eq. (5a).

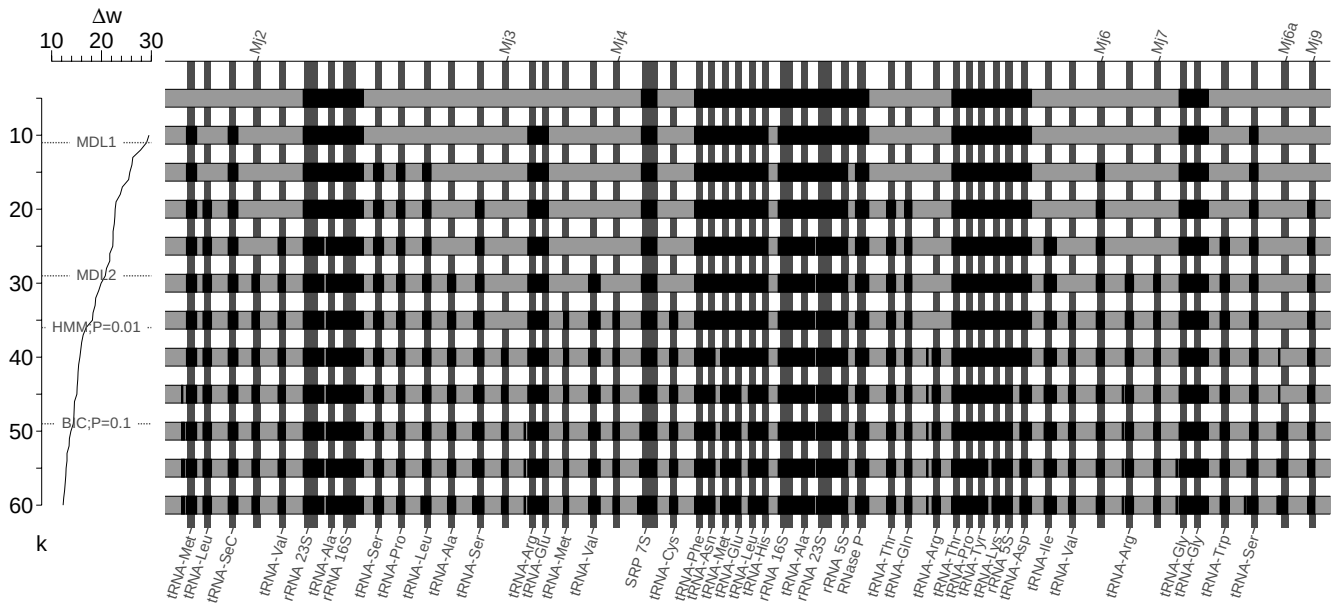

Fig. 6. Segmentation of *M. jannaschii*. The right-hand side shows maximal k-covers in increments of five. Horizontal grey bands correspond to covers, and darker boxes indicate the segments. Vertical bars show RNA genes with known function (bottom) and those in [8] (top). (Distances between loci are not drawn to scale: physical genomic coordinates 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 $\Delta w(k) \geq \alpha$. Penalty policies: (MDL1) $r(k) = 2k \log n$, (MDL2) $r(k) = 2k(\log n - \log(2k))$, (HMM) r defined by Eq. 5b, (BIC) $r(k) = k \log n$, (P=x) significance level x.

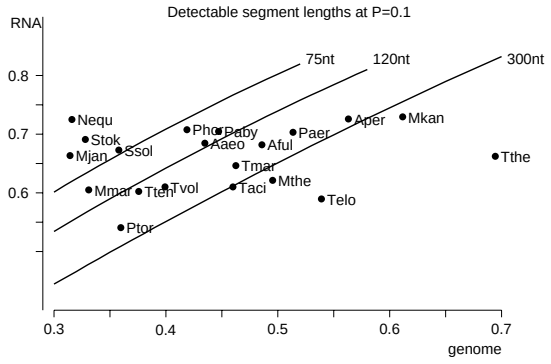

Fig. 7. Length of detectable RNA genes in thermophiles. Genome GC-content is shown on the X-axis, and tRNA GC-content is shown on the Y-axis. Organisms: Aaeo – *A. aeolicus*, Aful – *A. fulgidus*, Aper – *A. pernyx*, Mjan – *M. jannaschii*, Mkan – *M. kandleri*, Mmar – *M. maripaludis*, Mthe – *M. thermoautotrophicum*, Nequ – *N. equitans*, Paby – *P. abyssi*, Paer – *P. aerophilum*, Phor – *P. horikoshii*, Ptor — *P. torridus*, Ssol – *S. solfataricus*, Stok – *S. tokodaii*, Taci – *T. acidophilum*, Telo – *T. elongatus*. Tmar – *T. maritima*, Tten – *T. tengcongensis*, Tthe – *T. thermophilus*, Tvol – *T. volcanium*. We calculated GC-contents for *N. equitans*, *M. kandleri*, *M. maripaludis*, *P. aerophilum*, *P. torridus*, *S. tokodaii*, *T. elongatus*, and *T. thermophilus* from public genome sequences and annotations, and used the values computed by [26] for others. Lines show GC-content pairs for $\ell_{\min} = 75, 150, 300$ at $P = 0.1$ and genome size of 2 Mbp.

that a maximal 58-cover of the *N. equitans* genome (`NC_005213.1`) includes all tRNAs found by tRNAscan [29] (even pseudogenes), all rRNAs, four protein-coding genes, and 12 unannotated segments. Seven of the unannotated segments[3] produce no significant BLAST hits in Genbank. The *S. solfataricus* genome (`NC_002754.1`) contains many repetitive elements with a high GC-content. Our segmentation by a 116-cover includes 68 transposons or transposases, and two hypothetical protein-coding genes. All other segments are annotated as RNA genes. 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[4] 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 [28], it is very likely to be the so far unpublished RNase P RNA gene in *S. tokodaii* (J. W. Brown, personal communication). Figure 8 shows its alignment to the corresponding gene in *S. solfataricus*.

ML segmentation of *M. maripaludis* (`NC_005791.1`) gives only 16 GC-rich regions, one of which is unannotated, but further inspection [29] reveals that it is likely to be a Selenocysteine tRNA gene. Other structural RNA genes such as RNase P [28] and SRP [32] can be located in other thermophiles with a larger $\ell_{\min}$, based on GC-content alone. For instance, a segmentation of the *T. tengcongensis* genome (`NC_003869.1`) includes the unannotated region [1699151,1699436] which corresponds to the RNase P RNA

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

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

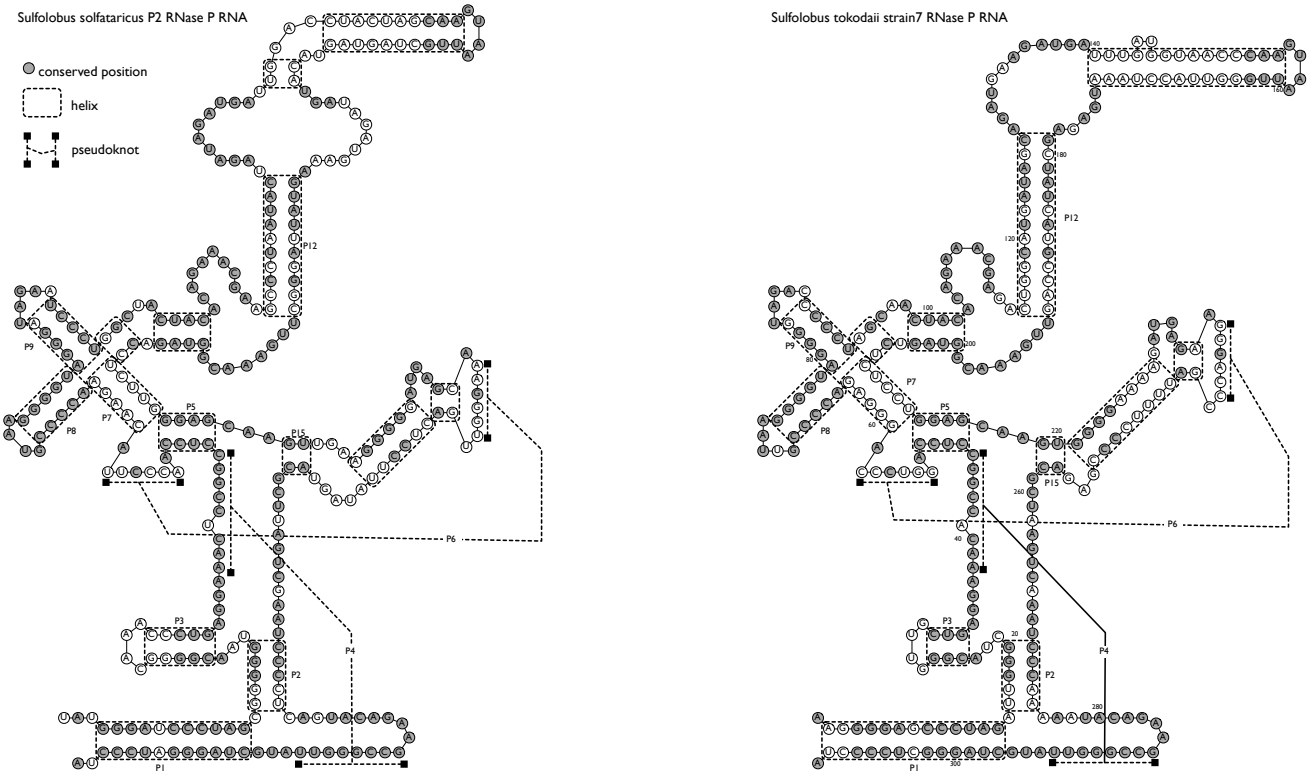

Fig. 8. Comparison of *S. solfataricus* and *S. tokodaii* RNase P RNA genes. *S. solfataricus* sequence and structure are from the RNase P RNA database [28]. The *S. tokodaii* sequence is the segment [326014,326323] in the genome, found by ML segmentation. Secondary structure for *S. tokodaii* is predicted from alignment and covariation.

gene of the organism mentioned in the article [26].

VI. 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. A relatively recent review [3] of DNA segmentation methods considered the cover selection problem, based on [5], 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. In fact, optimal cover problems with arbitrary, even non-additive, segment scoring can be solved in $O(n^2)$ time. For instance, [33] considers the problem of finding a cover with minimum total variance. A k-cover with maximum score can be found by dynamic programming in $O(n^2 k)$ time using $O(nk)$ space if a general segment scoring function is used [33], [34].

A related problem, that of finding a maximal *chain* of covers, can 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 [7] 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 §III-A, 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. For instance, $\mathcal{C}_1^*$ includes all positive scores in the example of Fig. 2. Theorem 3 shows that maximal covers may give more sound segmentation results than do maximal chains. It is also worth pointing out that looking for a cover $\mathcal{C}$ that maximizes $w(\mathcal{C}) - \alpha|\mathcal{C}|$, where w is set according to log-likelihood ratios, gives a symmetric solution in the sense that the role of changed and unchanged segments can be reversed. With the possible exception of segments at the extremities of $[1, n]$ that can have scores between $(-\alpha)$ and α, an equally good segmentation is found whether we use the scores w_i of Eq. (2) or $(-w_i)$. Maximal chains do not exhibit a similar symmetry.

In addition to maximal chains, results related to maximal covers are described by Huang [35], Lin *et al.* [36] and Zhang *et al.* [10]. A linear-time algorithm for finding a segment with maximum score satisfying a minimum length requirement is given in [35]. A linear-time algorithm for finding a segment with maximum score that satisfies both minimum and maximum length restrictions is described in [36]. Lin *et al.* [36] give an algorithm that finds a segment with maximum average score of some minimum length L in $O(n \log L)$ time. The algorithm can be employed to find a chain of covers in which segments have large average scores [37].

Zhang *et al.* [10] examine the problem of producing pairwise sequence alignments without low-scoring regions. An alignment is viewed as 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 [38] have essentially the same shortcomings as maximal cover chains in that they may include subalignments of arbitrarily low score (termed the "mosaic effect" in [39]). In order to avoid such situations, Zhang *et al.* [10] 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. The dynamic programming algorithm of Lemma 2 offers a simple, efficient alternative to the algorithm of [10] for eliminating low-scoring regions from alignments. MINLENGTH-COVER additionally provides the option of imposing minimum subalignment lengths.

In the context of Hidden Markov Models, Algorithms MAXCOVER and MINLENGTH-COVER find most likely state sequences under restrictions on the number of state changes, and on state durations, respectively. At first sight, it seems that the HMM approach has the advantage over the maximum likelihood approach of §III-A in that the model's parameters can be set by a training algorithm. There is no reason, however, for not doing the same in conjunction with ML segmentation. For instance, Viterbi training can be imitated, by alternating a parameter estimation step and a segmentation step. In the former, the character frequencies are calculated in changed and unchanged segments, and α is set based on statistical significance from Equation (4). In the segmentation step, a segmentation is computed using the newly set parameters.

From a practical viewpoint, our algorithms complement each other. The fast dynamic programming algorithms of §IV-B can identify a plausible maximal cover, while the algorithms MAXCOVER, MAXCOVER-FAST enable the exploration of maximal covers in the neighborhood with increasing or decreasing sizes, respectively. The experiments of Section V illustrate that our algorithms used in a maximum likelihood framework yield higher sensitivity and flexibility for DNA segmentation than does a two-state HMM, while matching the speed of the latter. They are very space efficient at the same time. The scores can be computed on demand from the input sequence. The dynamic programming algorithms need to store only the traceback values, as the few values of W^i that are accessed can be tracked in a small data structure. Every nucleotide can be stored in two bits (or four to allow for ambiguities), and the data encoding the cover and the traceback values can be stored in three bits per position. Thus MAXCOVER and the algorithms of §IV-B can segment a DNA sequence of length n using $n + O(1)$ bytes.

Acknowledgments: I am very grateful to James W. Brown for confirming the identification of the *S. tokodaii* RNase P gene and for computing its secondary structure. 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] M. Csűrös, "Algorithms for finding maximal-scoring segment sets (extended abstract)," in *Algorithms in Bioinformatics, Fourth Workshop*, I. Jonassen and J. Kim, Eds., LNCS 3240, Heidelberg: Springer-Verlag, 2004, pp. 62–73.

[2] J. Bentley, "Programming pearls: algorithm design techniques," *Comm. ACM*, vol. 27, no. 9, pp. 865–873, 1984.

[3] J. V. Braun and H.-G. Müller, "Statistical methods for DNA sequence segmentation," *Statist. Sci.*, vol. 13, no. 2, pp. 142–162, 1998.

[4] S. Karlin and V. Brendel, "Chance and significance in protein and DNA analysis," *Science*, vol. 257, no. 5066, pp. 39–49, 1992.

[5] Y.-X. Fu and R. N. Curnow, "Maximum likelihood estimation of multiple change points," *Biometrika*, vol. 77, no. 3, pp. 563–573, 1990.

[6] W. Li, P. Bernaola-Galván, F. Haghighi, and I. Grosse, "Applications of recursive segmentation to the analysis of DNA sequences," *Comput. Chem.*, vol. 26, pp. 491–510, 2002.

[7] W. L. Ruzzo and M. Tompa, "A linear time algorithm for finding all maximal scoring subsequences," in *Proc. 7th Intl. Conf. Intelligent Systems in Molecular Biology*. AAAI Press, 1999, pp. 234–241.

[8] R. J. Klein, Z. Misulovin, and S. R. Eddy, "Noncoding RNA genes identified in AT-rich hyperthermophiles," *Proc. Natl. Acad. Sci. USA*, vol. 99, no. 11, pp. 7542–7547, 2002.

[9] G. A. Churchill, "Stochastic models for heterogeneous DNA sequences," *Bull. Math. Biol.*, vol. 51, no. 1, pp. 79–94, 1989.

[10] Z. Zhang, P. Berman, T. Wiehe, and W. Miller, "Post-processing long pairwise alignments," *Bioinformatics*, vol. 15, no. 12, pp. 1012–1019, 1999.

[11] M. Kearns, Y. Mansour, A. Y. Ng, and D. Ron, "An experimental and theoretical comparison of model selection methods," *Machine Learning*, vol. 27, pp. 7–50, 1997.

[12] D. V. Hinkley and E. A. Hinkley, "Inference about the change-point in a sequence of binomial variables," *Biometrika*, vol. 57, no. 3, pp. 477–488, 1970.

[13] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *Ann. Statist.*, vol. 11, no. 2, pp. 416–431, 1983.

[14] A. Barron, J. Rissanen, and B. Yu, "The Minimum Description Length principle in coding and modeling," *IEEE Trans. Inform. Theory*, vol. 44, no. 6, pp. 2743–2760, 1998.

[15] H. Akaike, "A new look at statistical model identification," *IEEE Trans. Autom. Control*, vol. 19, no. 6, pp. 716–723, 1974.

[16] G. Schwarz, "Estimating the dimension of a model," *Ann. Statist.*, vol. 6, no. 2, pp. 461–464, 1978.

[17] D. L. Iglehart, "Extreme values in the G1/G/1 queue," *Ann. Math. Stat.*, vol. 43, no. 2, pp. 627–635, 1972.

[18] S. Karlin, A. Dembo, and T. Kawabata, "Statistical composition of high-scoring segments from molecular sequences," *Ann. Statist.*, vol. 18, no. 2, pp. 571–581, 1990.

[19] S. Karlin and S. F. Altschul, "Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes," *Proc. Natl. Acad. Sci. USA*, vol. 87, no. 6, pp. 2264–2268, 1990.

[20] L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[21] S. R. Eddy, "Noncoding RNA genes and the modern RNA world," *Nature Reviews Genetics*, vol. 2, pp. 919–929, 2001.

[22] P. Schattner, "Searching for RNA genes using base composition statistics," *Nucleic Acids Res.*, vol. 30, no. 9, pp. 2076–2082, 2002.

[23] N. Galtier and J. Lobry, "Relationships between genomic G+C content, RNA secondary structures, and optimal growth temperature in Prokaryotes," *J. Mol. Evol.*, vol. 44, pp. 632–636, 1997.

[24] N. Galtier, N. Tourasse, and M. Gouy, "A nonhyperthermophilic common ancestor to extant life forms," *Science*, vol. 283, pp. 220–221, 1999.

[25] H.-C. Wang and D. A. Hickey, "Evidence for strong selective constraint acting on the nucleotide composition of 16S ribosomal RNA genes," *Nucleic Acids Res.*, vol. 30, no. 11, pp. 2501–2507, 2002.

[26] Q. Bao, Y. Tian, W. Li, Z. Xu, Z. Xuan, S. Hu, W. Dong, J. Yang, Y. Chen, Y. Xue, Y. Xu, X. Lai, L. Huang, X. Dong, Y. Ma, L. Ling, H. Tan, R. Chen, J. Wang, J. Yu, and H. Yang, "A complete sequence of the *T. tengcongensis* genome," *Genome Res.*, vol. 12, no. 5, pp. 689–700, 2002.

[27] N. F. W. Saunders, T. Thomas, P. M. G. Curmi, J. S. Mattick, E. Kuczek, R. Slade, J. Davis, P. D. Franzmann, D. Boone, K. Rusterholtz, R. Feldman, C. Gates, S. Bench, K. Sowers, K. Kadner, A. Aerts, P. Dehal, C. Detter, T. Glavina, S. Lucas, P. Richardson, F. Larimer, L. Hauser, M. Land, and R. Cavicchioli, "Mechanisms of thermal adaptation revealed from the genomes of the Antarctic Archaea *Methanogenium frigidum* and *Methanococcoides burtonii*," *Genome Res.*, vol. 13, no. 7, pp. 1580–1588, 2003.

[28] J. W. Brown, "The ribonuclease P database," *Nucleic Acids Res.*, vol. 27, no. 1, p. 314, 1999.

[29] T. M. Lowe and S. R. Eddy, "tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence," *Nucleic Acids Res.*, vol. 25, no. 5, pp. 955–964, 1997.

[30] E. Waters, M. J. Hohn, I. Ahel, D. E. Graham, M. D. Adams, M. Barnstead, K. Y. Beeson, L. Bibbs, R. Bolanos, M. Keller, K. Kretz, X. Lin, E. Mathur, J. Ni, M. Podar, T. Richardson, G. G. Sutton, M. Simon, D. Soll, K. O. Stetter, J. M. Short, and M. Noordewier, "The genome of *Nanoarchaeum equitans*: insights into early archaeal evolution and derived parasitism," *Proc. Natl. Acad. Sci. USA*, vol. 100, no. 22, pp. 12 984–12 988, 2003.

[31] Y. Kawarabayashi, Y. Hino, H. Horikawa, K. Jin-no, M. Takahashi, M. Sekine, S. Baba, A. Ankai, H. Kosugi, A. Hosoyama, S. Fukui, Y. Nagai, K. Nishijima, R. Otsuka, H. Nakazawa, M. Takamiya, Y. Kato, T. Yoshizawa, T. Tanaka, Y. Kudoh, J. Yamazaki, N. Kushida, A. Oguchi, K. Aoki, S. Masuda, M. Yanagii, M. Nishimura, A. Yamagishi, T. Oshima, and H. Kikuchi, "Complete genome sequence of an aerobic thermoacidophilic crenarchaeon, *Sulfolobus tokodaii* strain7," *DNA Research*, vol. 8, no. 4, pp. 123–140, 2001.

[32] M. Alm Rosenblad, J. Gorodkin, B. Knudsen, C. Zwieb, and T. Samuelsson, "SRPDB (Signal Recognition Particle Database)," *Nucleic Acids Res.*, vol. 31, pp. 363–364, 2003.

[33] T. R. Bement and M. S. Waterman, "Locating maximum variance segments in sequential data," *Mathematical Geology*, vol. 9, no. 1, pp. 55–61, 1977.

[34] I. E. Auger and C. E. Lawrence, "Algorithms for the optimal identification of segment neighborhoods," *Bull. Math. Biol.*, vol. 51, no. 1, pp. 39–54, 1989.

[35] X. Huang, "An algorithm for identifying regions of a DNA sequence that satisfy a content requirement," *Comput. Appl. Biosci.*, vol. 10, pp. 219–225, 1994.

[36] Y.-L. Lin, T. Jiang, and K.-M. Chao, "Efficient algorithms for locating the length-constrained heaviest segments with applications to biomolecular sequence analysis," *J. Comp. Syst. Sci.*, vol. 65, pp. 570–586, 2002.

[37] Y.-L. Lin, X. Huang, T. Jiang, and K.-M. Chao, "MAVG: locating non-overlapping maximum average segments in a given sequence," *Bioinformatics*, vol. 19, no. 1, pp. 151–152, 2003.

[38] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *J. Mol. Biol.*, vol. 147, pp. 195–197, 1981.

[39] A. N. Arslan, O. Eğecioğlu, and P. A. Pevzner, "A new approach to sequence comparison: normalized sequence alignment," *Bioinformatics*, vol. 17, no. 4, pp. 327–337, 2001.