MULTISCALE DNA PARTITIONING: STATISTICAL EVIDENCE FOR SEGMENTS

Andreas Futschik[1,*], Thomas Hotz[2,1], Axel Munk[3,4], Hannes Sieling[3]

[1]Department of Applied Statistics, JKU University Linz, A-4040 Linz, Austria.

[2]Institute of Mathematics, Technische Universität Ilmenau, D-98693 Ilmenau, Germany.

[3]Institute for Mathematical Stochastics and Felix Bernstein Institute for Mathematical Statistics in Biosciences, Georgia Augusta University of Goettingen, D-37077 Göttingen, Germany.

[4]Max Planck Institute for Biophysical Chemistry, D-37077 Goettingen, Germany.

Contact:

andreas.futschik@jku.at, thomas.hotz@tu-ilmenau.de, munk@math.uni-goettingen.de, hsielin@uni-goettingen.de

[1]to whom correspondence should be addressed

Abstract

Motivation:

DNA segmentation, i.e. the partitioning of DNA in compositionally homogeneous segments, is a basic task in bioinformatics. Different algorithms have been proposed for various partitioning criteria such as GC content, local ancestry in population genetics, or copy number variation. A critical component of any such method is the choice of an appropriate number of segments. Some methods use model selection criteria, and do not provide a suitable error control. Other methods that are based on simulating a statistic under a null model provide suitable error control only if the correct null model is chosen.

Results:

Here, we focus on partitioning with respect to GC content and propose a new approach that provides statistical error control: as in statistical hypothesis testing, it guarantees with a user specified probability $1-\alpha$ that the number of identified segments does not exceed the number of actually present segments. The method is based on a statistical multiscale criterion, rendering this as segmentation method which searches segments of any length (on all scales), simultaneously. It is also very accurate in localizing segments: under bench-mark scenarios, our approach leads to a segmentation that is more accurate than the approaches discussed in the comparative review of [18]. In our real data examples, we find segments that often correspond well to features taken from standard UCSC genome annotation tracks.

Availability:

Our method is implemented in function *smuceR* of the R-package *stepR*, available from *http://www.stochastik.math.uni-goettingen.de/smuce*.

1. **Introduction**

It has been observed a long time ago [30] that DNA sequences are not composed homogeneously and that bases fluctuate in their frequency. These inhomogeneities often have an evolutionary or a functional interpretation, and can be relevant for the subsequent analysis of sequence data. Since it correlates with many features of interest, the GC content, i.e. the relative frequency of the bases G and C, is one of the most commonly studied sequence properties.

Large scale regions, typically 300kb [5], of approximately homogeneous GC content have been called isochores. In view of the somewhat vague notion of "approximate homogeneity", and conceptual criticism in studies such as [13], or [18], there is less interest in isochores nowadays. There is no doubt about variation in GC content along genomes however, and search is done instead for domains of any length exhibiting distinct local GC content; see for instance [19]. Several factors can influence the GC content of a region. At larger scales, it correlates with the density of genes, with gene rich regions typically exhibiting an elevated GC content compared to regions of low gene density. At smaller scales, there is fluctuation in GC content for instance due to repetitive elements and GpC islands. The GC content is also known to vary between exons and introns. Indeed, especially for long introns, their lower GC content seems to play a role in splice site recognition [1]. There is also a correlation between GC content and the local recombination rate [24]. For a further discussion of features correlated to the GC content, see [23].

In gene expression studies, regions of homogeneous GC content are of interest as the GC content of a region affects the number of reads mapped to this region. For DNA and RNA-seq experiments with the Illumina Genome Analyzer platform, this has been for instance investigated in [2] and [29].

Segmentation algorithms aim to partition a given DNA sequence into stretches that are homogeneous in their composition, but differ from neighboring segments. The classical approach of using moving windows is very simple and available for instance as an option with the UCSC and Ensembl genome browsers. It has some disadvantages, however. For instance, the choice of the window size is difficult as it defines implicitly a fixed scale at which segments primarily will be detected. Further, the involved smoothing blurs abrupt changes. Without additional statistical criteria, the method also does not tell us whether differences between neighboring windows are statistically significant. Therefore several more sophisticated approaches have been proposed. These methods include hidden Markov Models ([10, 11]) and walking Markov Models ([20]). There are also change point methods available, see for instance [8]. A Bayesian approach that relies on the Gibbs sampler has been proposed by [26]. An older approach based on information criteria can be found in [28]. Furthermore, recently developed methods based on information criteria have been shown to perform particularly well, see [18], [19]. A review of segmentation methods can be found in [7], and for a more recent comparative evaluation of the more popular approaches see [18].

In this paper we focus on binary segmentation, where the four letter alphabet of a DNA sequence is converted into a two letter code. For GC content, we set the response to be “1” for G or C at a position, and zero for A/T; we use Y_i to denote the response at position i, and summarize the responses for a sequence of length n by

$$Y = (Y_1, Y_2, \dots, Y_n). \tag{1}$$

We model the responses Y_i to be independent and Bernoulli $\mathrm{Bin}(1, \pi_i)$ distributed, and also assume that there is a partition $0 = \tau_0 < \tau_1 < \dots < \tau_K = n$ into K segments on which the π_i are piecewise constant, i.e. $\pi_i = p_j$ for $i \in I_j$. Here, $I_j := (\tau_{j-1}, \tau_j]$ denotes the j'th segment with response probability p_j for $1 \leq j \leq K$ while K is unknown and can be arbitrarily large.

A segmentation algorithm provides estimates $\hat{K}$ for the number of segments, for the internal segment boundaries,

$$0 = \hat{\tau}_0 < \hat{\tau}_1 < \hat{\tau}_2 < \dots < \hat{\tau}_{\hat{K}-1} < \hat{\tau}_{\hat{K}} = n, \tag{2}$$

and for the response probabilities, $\hat{p}_j$, on the estimated segments $\hat{I}_j := (\hat{\tau}_{j-1}, \hat{\tau}_j]$.

In the following we will identify a segmentation with (p, I), where $p = (p_1, \dots, p_K)$ and $I = (I_1, \dots, I_K)$.

Our proposed algorithm provides a parsimonious estimate $\hat{K}$ for K: $\hat{K}$ will not exceed the actual number of segments K, except for a small user specified error probability α; as a default value, we suggest $\alpha = 5\%$, the error probability also chosen in our simulations and data analyses. Relaxing this significance level to a larger value, $\alpha = 20\%$ say, will typically lead to more identified segments, but at the cost of statistical accuracy.

2. **Method**

Our proposed multiscale segmentation provides estimates for the number of segments and their boundaries at the same time. We employ a certain multiscale statistic which will ensure that the estimator fits the data well on all segments simultaneously, i.e. the number of segments is not underestimated with high probability. This estimator is based on [21] who proposed a general *statistical multiscale change-point estimator* (SMUCE) for exponential family models. Exponential families include many classes of well known distributions, such as the Gaussian (normal) class, the Poisson class or the Bernoulli class, which is of particular interest for this paper.

In the Gaussian setting, a related estimator has also been suggested in [15]. Forerunners of SMUCE are based on a penalized likelihood with a penalty that depends on the number of jumps, see e.g. [6,8,32,33] and the introduction in [21] for a brief survey. The underlying multiscale statistic is based on the work of [16], see also [17] and [31]. For a general description of the approach underlying the present work, its statistical interpretation, statistical optimality, and theoretical results we again refer to [21]. To ease understanding, in the following we elaborate in greater details on the case of binary Bernoulli observations with success probabilities given as piecewise constant segments, as this model underlies the methodology for the segmentation problem at hand. In contrast to other approaches such as Hidden Markov Models, we require neither explicit nor implicit distributional assumptions on the segments and their lengths.

Let $\ell(Y_i, \pi_i)$ denote the likelihood of Y_i under the parameter π_i, i.e. $\ell(Y_i, \pi_i) = \pi_i$ if $Y_i = 1$ and $\ell(Y_i, \pi_i) = 1 - \pi_i$ else. We then define for a fixed interval $(i, j]$ with $0 \leq i < j \leq n$ the local likelihood ratio statistic

$$T_{(i,j]}(p_0) = \log\left(\max_{\tilde{p}_0\in[0,1]} \prod_{l=i+1}^{j} \frac{\ell(Y_l, \tilde{p}_0)}{\ell(Y_l, p_0)}\right). \tag{3}$$

Roughly, this statistic indicates how well the data on the sub-interval $(i, j]$ are described by the constant response probability $p_0 \in [0, 1]$ of some segment under consideration as opposed to choosing some $\tilde{p}_0 \in [0, 1]$ freely for that sub-interval.

As a goodness-of-fit measure for the segmentation (p, I) we consider the scale-calibrated multiresolution statistic

$$T_n(p, I) = \max_{1\leq l\leq K} \max_{(i,j]\subseteq I_l} \left(\sqrt{2T_{(i,j]}(p_l)} - \sqrt{2\log\left(\frac{e \cdot n}{j - i}\right)}\right). \tag{4}$$

(Here "e" denotes Euler's number.) This statistic may be interpreted as follows: for all segments I_l in I the response probability is assumed to be constant, and for every interval $(i, j]$ within such a segment, $T_n(p, I)$ measures whether the data are well described by the constant response probability p_l on that interval. It thus checks the quality of fit on all scales simultaneously, hence the name. Note that the log-penalty term depends on the length $j - i$ of the interval that is currently checked for deviations from the model. Indeed, it takes the number of disjoint intervals of the considered length into account, thereby adjusting for multiple testing. For our final estimate we determine

(1) the minimal number of segments $\hat{K}$, such that there exists a segmentation $(\hat{p}, \hat{I})$ with $\hat{K}$ segments satisfying the multiresolution constraint $T_n(\hat{p}, \hat{I}) \leq q$ for some pre-determined significance threshold q, and
(2) the segmentation $(\hat{p}, \hat{I})$ with maximal likelihood among all segmentation having $\hat{K}$ segments and satisfying the multiresolution constraint.

To be more precise, let C_K denote the set of segmentation with K segments. Then, our estimate in the second step is

$$(\hat{p},\hat{I}) = \underset{(p,I)\in C_{\hat{K}},T_n(p,I)\leq q}{\operatorname{argmax}} \ell(Y;p,I), \tag{5}$$

where argmax denotes a position $(\hat{p},\hat{I})$ at which the maximum is obtained, and $\ell(Y;p,I)$ denotes the likelihood of all data, if the segmentation (p,I) with $\hat{K}$ segments were true, i.e.

$$\ell(Y;p,I) = \prod_{1\leq l\leq \hat{K}} \prod_{i\in I_l} \ell(Y_i,p_l). \tag{6}$$

Following [21], the general class of such estimators in exponential families has been denoted as SMUCE (Statistical Multiscale Change point Estimator). We adopt this terminology and will denote the estimator in (5) for the Bernoulli and binomial case as B-SMUCE.

The threshold parameter q determines the parsimony of the estimator, the larger q the fewer segments will be included into B-SMUCE. Hence, the choice of q is crucial. A statistically attractive feature of B-SMUCE is that q can be chosen as the $(1-\alpha)$-quantile of the distribution of $T_n(p,I)$ under the hypothesis that (p,I) is the true model. In [21] it has been proven that this choice ensures that the number of segments is not *overestimated* with probability at least $1-\alpha$.

To be a little more precise, in [21] Theorem 2.1 was shown under some mild technical assumptions that for any (p,I) the asymptotic distribution of the multiresolution statistic $T_n(p,I)$ can be bounded by the asymptotic distribution of the statistic for a signal with only one segment. Moreover, the latter distribution converges to the limit distribution of (4) for the case of i.i.d. zero-mean Gaussian observations. Therefore, we may (and we do in the following) simply use Monte Carlo simulations with i.i.d. zero-mean Gaussian data to determine bounds on the quantiles of the distribution of $T_n(p,I)$. In fact, in simulations we found the approximate quantiles so obtained to be rather conservative (i.e. the preassigned error probability α was not exceeded) even for small sample sizes. This adds support to the the basic inequality

$$P(\hat{K} > K) \leq P(T_n(p,I) > q) \leq \alpha \tag{7}$$

in Section 1.2. of (Frick et al. 2013) which renders SMUCE to be a method which statistically controls the error to *overestimate* the number of segments in the binary case, i.e. it provides the statistical validity of B-SMUCE in the above sense (7). The other way around, Theorem 2.2 in [21] provides an exponential deviation bound for the error to *underestimate* the true number of segments which explicitly depends on the smallest segment length and signal strength. Under prior information on these quantities these two inequalities together even allow to give a guarantee for the probability $P(\hat{K}=K)$ of specifying the number of segments *correctly*. Moreover, in the Gaussian case, it can be shown that SMUCE attains optimal detection rates (and even constants) over a large range of segment lengths, see [21], Theorems 2.6, 2.7. Our simulations suggest a similar performance in the binary/binomial case, although we do not have a rigorous proof of this statement.

2.1. **Details of the Algorithm**

We follow [21] and use a pruned version of dynamic programming to compute our estimated segmentation $\hat{I}$ and levels $\hat{p}$. This is possible since our multiresolution statistic T_n considers only subintervals of the candidate segments of constant response probability. For related ideas, where dynamic programming has been used for other segmentation estimators, see e.g. [22], [6], [15] and in particular for pruning [27]. To describe the algorithm for computing B-SMUCE, we need the following notation, identifying a segmentation with (p, I) again: for an interval $(i, j]$ we define the local costs of a response probability $p_0 \in [0, 1]$ as

$$c_{(i,j]}(p_0) = \begin{cases} -\prod_{l=i+1}^{j} \ell(Y_l, p_0) & \text{if } T_{(i,j]}(p_0) \leq q, \\ \infty & \text{otherwise.} \end{cases} \tag{8}$$

Let $c^*_{(i,j]} = \min_{p_0 \in [0,1]} c_{(i,j]}(p_0)$ denote the minimal costs on $(i, j]$ for a constant response probability under the multiresolution constraint, while $p_{(i,j]} = \operatorname{argmin}_{p_0 \in [0,1]} c_{(i,j]}(p_0)$ denotes the corresponding optimal estimate.

Let us for the moment only consider the observations $Y_1, \dots, Y_i$ for fixed $1 \leq i \leq n$, and denote by $c^*_{i,K}$ the optimal overall costs on $(0, i]$ using K segments, i.e.

$$c^*_{i,K} = \operatorname*{argmax}_{(p,I) \in C_{K,i}, T_i(p,I) \leq q} \prod_{1 \leq l \leq K} \prod_{j \in I_l} \ell(Y_j, p_l), \tag{9}$$

cf. (5), where $C_{K,i}$ denotes the set of segmentation of $(0, i]$ with K segments; if no segmentation $(p, I) \in C_{K,i}$ fulfills the multiresolution constraint $T_i(p, I) \leq q$, we let $c^*_{i,K} = \infty$.

The algorithm for B-SMUCE is then based on the observation that for $K > 0$

$$c^*_{i,K} = \min_{1 \leq l \leq i} \left(c^*_{l,K-1} + c^*_{(l,i]} \right). \tag{10}$$

In dynamic programming this is called the Bellman equation; it is the main ingredient for an efficient implementation, see line 11 in Algorithm 1.

Algorithm 1 Dynamic Programming Algorithm for B-SMUCE

```
 1: $\hat{K} \leftarrow 0$, $\hat{I}_0 \leftarrow \emptyset$, $\hat{p}_0 \leftarrow \emptyset$, $i \leftarrow 1$
 2: while $i \leq n$ do
 3:   if $\hat{K} = 0$ then
 4:     $l^* \leftarrow 0$
 5:     $c^*_{i,0} \leftarrow c^*_{(0,i]}$
 6:   else
 7:     for $l = i-1, \ldots, 1$ do
 8:       if $c_{(l,i]} = \infty$ then
 9:         goto 14
10:       else
11:         $c^l_i \leftarrow c^*_{l,\hat{K}-1} + c^*_{(l,i]}$
12:       end if
13:     end for
14:     $l^* \leftarrow \operatorname{argmin}_{l \leq j < i} c^j_i$
15:     $c^*_{i,\hat{K}} \leftarrow c^{l^*}_i$
16:   end if
17:   if $c^*_{i,\hat{K}} = \infty$ then
18:     $\hat{K} \leftarrow \hat{K} + 1$
19:     goto 3
20:   end if
21:   $\hat{I}_i \leftarrow \big(\hat{I}_{l^*}, (l^*, i]\big)$
22:   $\hat{p}_i \leftarrow \big(\hat{p}_{l^*}, p_{(l^*,i]}\big)$
23: end while
24: return $\hat{K}$, $\hat{I}_n$, $\hat{p}_n$
```

Note that we introduced two rules that permit for early stopping of loops: if $c^*_{(l,i]} = \infty$, i.e. if the hypothesis of constancy on $(l, i]$ is rejected, then consequently this also happens on any larger interval, i.e. for any smaller l; this justifies lines 8–9. Similarly, if K segments are insufficient to fulfill the multiresolution constraint on $(0, i]$ then a fortiori so for any larger i, whence lines 17–20. To the best of our knowledge, these shortcuts that are possible due to the specific structure of the multiscale constraint, have not been used so far. Additional improvements were used in our implementation; these, however, are rather technical and thus omitted from the present paper.

3. **Results**

We evaluated our segmentation approach both on simulated data and on data taken from the human genome and the long known λ-phage. In our simulations, we used the benchmark scenarios proposed in [18].

Since an extensive comparison of popular DNA segmentation algorithms under these benchmark scenarios is already available in [18], we provide a comparison of our approach with the method that performed best in [18], namely the one based on the Jensen-Shannon divergence. This recursive approach (called D_{JS}) splits one of the current segments in each step. This is done by adding a new break point such that the improvement in Jensen-Shannon divergence is maximized. The algorithm stops when the improvement does not reach a threshold value obtained via simulations. Here, we used the Matlab implementation *Djsegmentation.m* of the algorithm which is publicly available as part of ISOPLOTTER 2.4 (http://code.google.com/p/isoplotter/). There 5.8×10^{-5} is taken as threshold, a value obtained from simulating long (1Mb) homogeneous sequences. While this value seems to work well for the considered benchmark scenarios and might also be useful to prevent false positives when searching for long homogeneous sequences, it might be less suitable for balancing false positives and false negatives under other scenarios. Therefore a modified version (called ISOPLOTTER) of D_{JS} has been proposed briefly after [19] that uses critical values dependent both on the segment length and the standard deviation of GC content. We therefore also report on the performance of ISOPLOTTER 2.4 (again under the default parameter settings), and provide detailed results in the supplement.

To facilitate comparison and to accelerate the computations for longer sequences, we binned the data and applied our algorithm to the resulting binomial frequencies. We choose the bin size equal to 32 which is the default value with the D_{JS} and IsoPlotter software, and has also been used in [18]. While binning the data clearly improves the speed, it should be noted that it is not essential for the algorithm to work.

The first scenario considered there involves sequences consisting of fixed-size homogeneous domains. Although not realistic in practice, this setup has been proposed by [18] as a minimum standard: a criterion that does not perform well on such data cannot be expected to perform well under more complex settings. The second scenario consists of sequences with domains of random length generated according to a power-law distribution. These sequences are reported to mimic mammalian genomes well, see [12].

3.1. **Performance measures**

We measured performance both by a qualitative criterion proposed by [18], and by a new quantitative criterion. For the qualitative criterion, we classify an identified segment as true positive, if both segment boundaries are identified correctly within an error margin of 5000 bases, or 5% of the segment length, whichever is smaller. Thus an identified segment is considered to be a false positive, unless both detected boundaries were within 5000 bases (or 5%) from the boundaries of a true segment. Similarly, actual segments were taken as false negatives, if they were not detected correctly within the permitted tolerance. Let now tp, fp, and fn denote the number of true positives, false positives and false negatives respectively. Following [18], we define a sensitivity rate as

$$r_s := \frac{tp}{tp + fn}, \tag{11}$$

and a precision rate as

$$r_p := \frac{tp}{tp + fp}. \tag{12}$$

We investigate the performance of our proposed approach based on these criteria.

Furthermore, we defined two quantitative criteria that better reflect the accuracy of detecting segment boundaries. We denote them by *false negative sensitive localization error* (FNSLE) and *false positive sensitive localization error* (FPSLE). To introduce the FNSLE, consider a true segment $I_j := (\tau_{j-1}, \tau_j]$, and let $m_j = \frac{\tau_{j-1}+\tau_j}{2}$ denote the midpoint of the segment. We define the best matching estimated segment as the segment $\hat{I}_l = (\hat{\tau}_{l-1}, \hat{\tau}_l]$ with $m_j \in \hat{I}_l$. The FNSLE for segment I_j is then defined as the mean distance

$$e_j^{(FNSL)} = \frac{1}{2}\big(|\tau_{j-1} - \hat{\tau}_{l-1}| + |\tau_j - \hat{\tau}_l|\big) \tag{13}$$

between true and estimated boundaries.

The overall FNSLE is then defined as the mean FNSLE taken over all true segments

$$e^{(FNSL)} := \frac{1}{K}\sum_{j=1}^{K} e_j^{(FNSL)}. \tag{14}$$

Analogously, the FPS localization error can be defined by measuring how closely an estimated segment matches to one of the true segments. By starting with an estimated segment $\hat{I}_l$ and its midpoint $\hat{m}_l$ we look for the true segment satisfying $\hat{m}_l \in I_j$. With analogously defined errors $e_l^{(FPSL)}$, we call

$$e^{(FPSL)} := \frac{1}{\hat{K}}\sum_{l=1}^{\hat{K}} e_l^{(FPSL)}. \tag{15}$$

the overall FPSLE.

These measures for the error may be interpreted as follows: assume the estimated segmentation agrees with the true segmentation in the number of segments, and that all boundaries have been determined with an error smaller than half the length of each neighboring segment. Then FPSLE and FNSLE agree: they essentially give the average distance between true and estimated boundaries. These error measures behave differently however, if the numbers of true and detected segments do not coincide: assume that the estimated segmentation is the true segmentation except that it has incorrectly split one true segment into two estimated segments. Then the FNSLE increases by the length of that true segment minus the length of the longer estimated segment divided by $2K$, i.e. a spurious split is treated like an error in localizing that boundary. The FPSLE, however, will get rather large, as the length of that true segment divided by $2\hat{K}$ gets added. Similarly, if a true boundary is not detected, the FNSLE will be larger.

3.2. **Simulations**

3.2.1. **Segments of Equal Length**

We first implemented scenario I of [18]. Thus we simulated sequences that consist of 10 segments of equal length. We considered the following eight different segment lengths: $10kb$, $50kb$, $100kb$, $200kb$, $300kb$, $500kb$, $1Mb$, $5Mb$. Thus the longest sequences had total length $50Mb$. For each sequence, we selected a global probability $\bar{p}_{(t)}$ for the response "1" at a position according to a uniform distribution on $[0.1, 0.9]$. Then we randomly modified this probability for each sequence segment j by taking $p_j = \bar{p}_{(t)} + \sigma Z_j$. Here Z_j denotes a standard normal random number, and σ was chosen from $\{0, 0.025, 0.05, 0.075, 0.1\}$. The p_j were conditioned to lie within $[0, 1]$, i.e. if p_j did not turn out to be a proper probability, a new random number was generated. The individual observations Y_i within a given segment I_j were then chosen as independent Bernoulli random variables with expected value p_j.

We simulated 100 sequences for each combination of segment length and heterogeneity σ of the segment specific response probabilities.

In [18], a detection threshold was introduced, and neighboring true segments for which the value of p_j differed by less than this threshold have been merged and considered as a single segment in the subsequent performance analysis. We did not use such a threshold, however, since we did not want to penalize high sensitivity. Indeed, a correct detection of two neighboring segments with unequal but too similar levels p_j would be counted as an error, if such a detection threshold was used.

The average sensitivity and precision rates of B-SMUCE and the method based on the Jensen-Shannon entropy (D_{JS}) are displayed in Figures 1 and 2 respectively. Especially for shorter segments, B-SMUCE performs better than D_{JS}, with higher sensitivity and precision. IsoPlotter performed worse than D_{JS} under the considered scenarios and gave up to 40 segments on average for the very long sequences. B-SMUCE is able to detect also short segments while controlling the number of spuriously detected segments. Notice however that in the case of a homogeneous sequence without partitioning into segments ($\sigma = 0$), D_{JS} always obtained the correct answer whereas B-SMUCE sometimes introduced spurious segment boundaries. This is to be expected, as the error of introducing spurious boundaries has been set to $\alpha = 5\%$ under such a model. Furthermore, under all scenarios, too many boundaries were estimated by B-SMUCE in less than 5% of the simulations, as predicted. We consider the ability to control this error to be a particular strength of our approach.

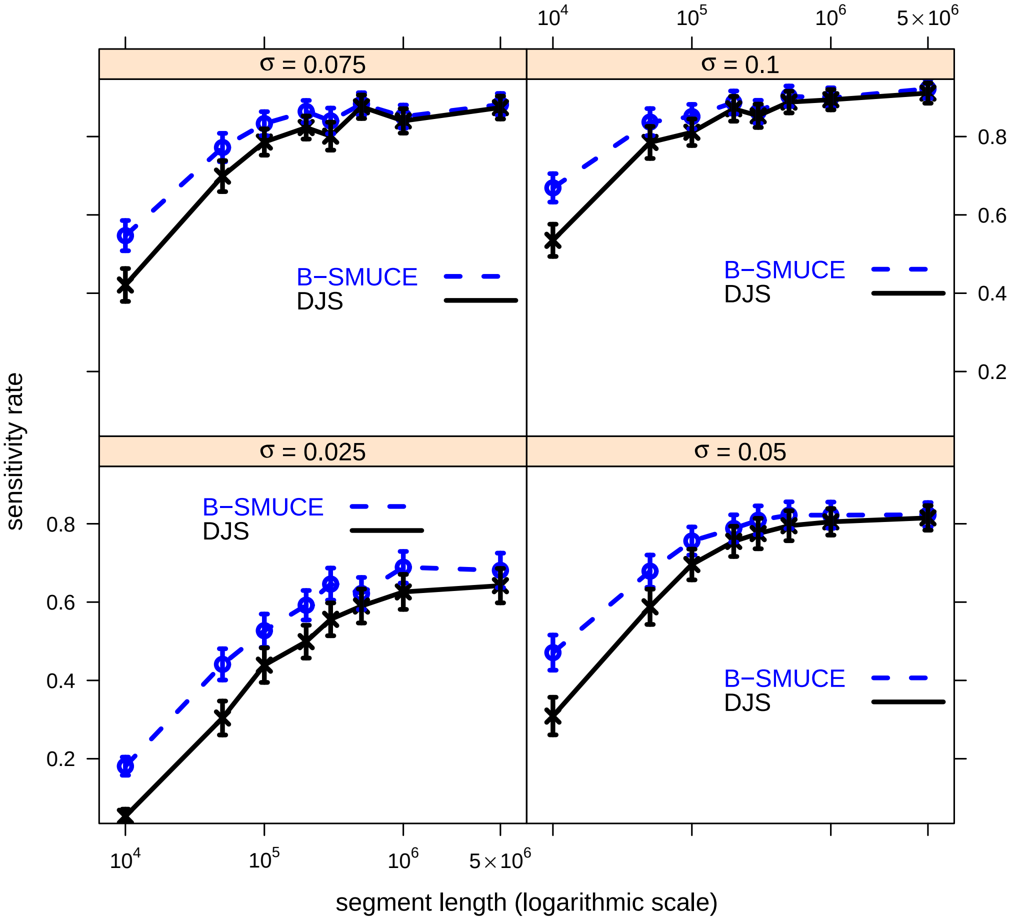

FIGURE 1. Average sensitivity rate as defined in (11) for D_{JS} (○) and B-SMUCE (×). Results are based on simulations under scenario I (segments of equal length) for several values of σ. The sensitivity obtained with the multiresolution criterion tends to be higher. At the simulated segment lengths, 95% confidence intervals are given as error bars.

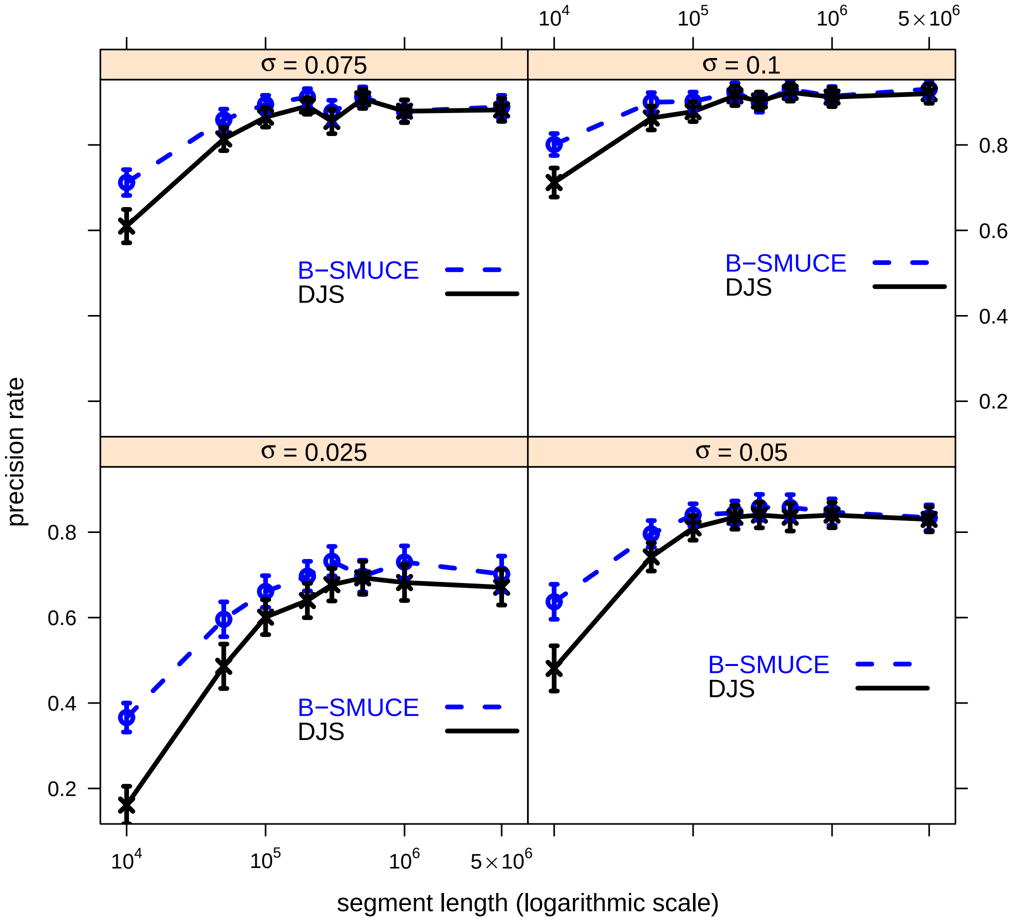

FIGURE 2. Average precision rate, as defined in (12) for D_{JS} ($\circ$) and B-SMUCE ($\times$). Results are based on simulations under scenario I (segments of equal length) for several values of σ. The precision rate obtained with the multiresolution criterion tends to be higher. At the simulated segment lengths, 95% confidence intervals are given as error bars.

Figures 3 and 4 show the average FNSLE (14) and FPS localization errors (15) that measure the accuracy of the segment detection. For these quantitative criteria, we only consider sequences that are non-homogeneous ($\sigma > 0$). Again, B-SMUCE shows better performance, leading to smaller errors on average. With increasing heterogeneity σ there will be typically larger differences between neighboring segments, and thus smaller errors. The graphs also seem to indicate that both FPSLE and FNSLE tend to get larger with increasing sequence lengths. A closer inspection reveals that this is mostly caused by outliers, as the median accuracy of detection stays nearly the same for all segment lengths. These outliers occur when a segment is either missed or detected incorrectly, and such events lead to larger errors when the segments are longer.

Table 1. Sensitivity rate and precision rate under scenario 2 for the Jensen-Shannon divergence method (D_{JS}) and B-SMUCE. Long and short segments were generated from a power law distribution and the total sequence length was $n = 10^6$. We provide averages (and in parentheses standard errors) over 100 simulation runs.

	D_{JS}	B-SMUCE
avg. true number	27.51 (1.90)	27.51 (1.90)
avg. number of true pos.	23.26 (1.69)	24.68 (1.80)
avg. number of false pos.	2.51 (0.20)	1.60 (0.14)
avg. number of false neg.	4.25 (0.37)	2.83 (0.24)
avg. sensitivity	0.83 (0.016)	0.87 (0.015)
avg. precision	0.88 (0.013)	0.92 (0.013)

3.2.2. **Segments According to Power Law**

In our second simulation setup, we generated 100 sequences consisting of segments with different lengths. As in [18], we chose the segment lengths according to a power law distribution:

$$p(x) = Cx^{-a}\mathbb{1}[x > x_0], \tag{16}$$

with $a = 1.55$ and $x_0 = 10000$. The parameter a was chosen to mimic segment lengths for mammalian, in particular human, DNA sequences; see [18], [12], [13]. Notice that the minimal segment length $x_0 = 10000$ was introduced to avoid very short segments that are difficult to detect. The total sequence length was taken to be $n = 10^6$. For even numbered segments, we selected the response probability p_j according to a uniform distribution on $[0.6, 1]$, whereas for odd segments we took p_j uniformly from $p_j \in [0, 0.4]$.

Qualitatively, it turns out that B-SMUCE performs better than the Jensen-Shannon entropy criterion D_{JS} both in terms of sensitivity and precision rate, see Table 1. B-SMUCE detected 90% of all true segments within the desired margin of error. Furthermore 94% of all detected segments were correct, again, within the desired level of accuracy. Since we used B-SMUCE with a type I error probability of 5% for including too many segments, this implies that almost all of the detected jumps were detected within the required level of accuracy. Furthermore when incorrect, our method usually detected not more than one spurious segment boundary. We also tried IsoPlotter on the simulated sequences and got 85.23 detected segments on average. Given an average number of 27.51 true segments (see Table 1), more than three times the true number of segments has been detected on average. More detailed results on ISOPLOTTER can be found in the supplementary material.

B-SMUCE also leads to good results in terms of the FPSLE and FNSLE, see Table 2. Thus on average, detected segments and true segments match more closely with B-SMUCE than with the D_{JS} criterion.

Table 2. False negative sensitive localization errors (FNSLE) and false positive sensitive localization errors (FPSLE) under scenario 2 for the Jensen-Shannon divergence method (D_{JS}) and B-SMUCE. Long and short segments were generated under a power law distribution and the total sequence length was $n = \times 10^6$. The results are averages (and standard errors) over 100 simulation runs. The error rates were standardized according to the average segment length.

	FNSLE	FPSLE
D_{JS}	0.27 (0.436)	0.06 (0.103)
B-SMUCE	0.11 (0.164)	0.01 (0.014)

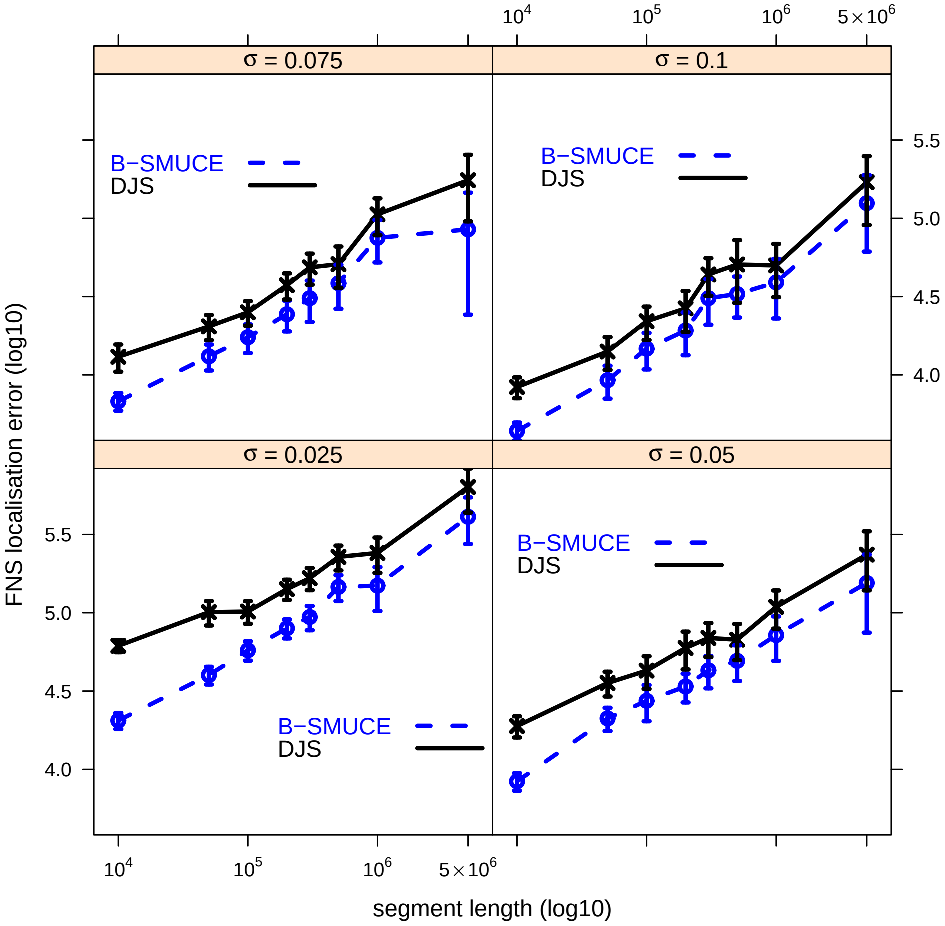

Figure 3. Logarithm (base 10) of average FNSLE (false negative sensitive localization error), as defined in (14) for D_{JS} (○) and B-SMUCE (×). Results are based on simulations under scenario I (segments of equal length) for several values of σ. The errors encountered with the multiresolution criterion tend to be smaller. At the simulated segment lengths, 95% confidence intervals are given as error bars.

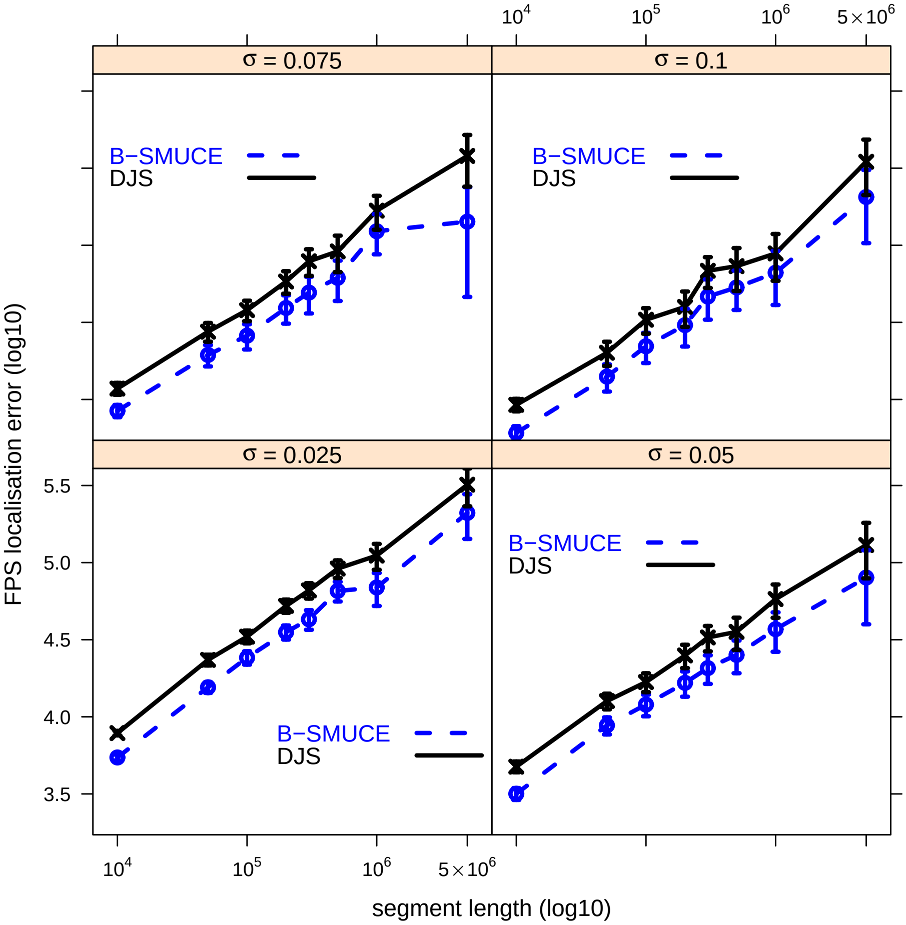

Figure 4. Logarithm (base 10) of average FPSLE, as defined in (15) for D_{JS} ($\circ$) and B-SMUCE ($\times$). Results are based on simulations under scenario I (segments of equal length) for several values of σ. The errors encountered with the multiresolution criterion tend to be smaller. At the simulated segment lengths, 95% confidence intervals are given as error bars.

3.3. **Real Data**

We applied our segmentation algorithm to three data sets. The first two examples, λ phage and human MHC complex, have previously been studied in the context of segmentation algorithms. As a further example, a 10Mb sequence chunk (hg19, chr1:50000002-60000000) has been arbitrarily chosen from human chromosome 1.

The genome of bacteriophage λ consists of 48502 bases and was one of the first completely sequenced genomes. Our segmentation led to 6 segments with boundaries *1, 22501, 27829, 33186, 39172, 46367, 48502.* Notice that the same number of segments although with a bit different boundaries has been reported as the outcome of a segmentation using Hidden Markov Models in Chapter 4 of [9].

We next investigate human genome data from chromosome 6p21.3 and 6p22.1 (hg19, chr6:29,677,952-33,289,874). This segment harbors the much studied human MHC complex.

We found a number of segments even larger than that in [19], contradicting once again the concept of homogeneous isochores. from the UCSC browser for this example.) We recoded G,C as “1” and A,T as zero, and applied B-SMUCE, and both D_{JS} and IsoPlotter to this data. With D_{JS}, we found 182 segments. With B-SMUCE and a type I error probability of $\alpha = 0.05$, we identified 640 segments. (With $\alpha = 0.01$, 528 segments were obtained, and choosing $\alpha = 0.1$ led to 716 segments.)

A natural question is whether the number of 640 or 182 segments is more appropriate. To address this issue, notice that the statistical error control associated with the multiresolution criterion suggests that there are (except for a small error probability) at least 640 segments. To check whether this finding is also compatible with the D_{JS} segmentation, we simulated the segmentation with 640 segments obtained with B-SMUCE as our null model. We simulated 100 data sets from this null model, and for 80% of all data sets D_{JS} led to a segmentation with the number of segments at most 182. With the number of segments taken as test statistic, this amounts to an estimated p-value of 0.80. Thus the segmentation based on the Jensen-Shannon (D_{JS}) criterion does not contradict the assumption of 640 segments, whereas the hypothesis of 182 segments is rejected by the multi-scale criterion underlying B-SMUCE as not being compatible with the data.

We also applied IsoPlotter 2.4 to the data. With its adaptive detection threshold, 227 segments were identified. The homogeneity test (one-sided F-test) provided with the IsoPlotter software confirmed for 180 of these 227 segments that they are significantly more homogeneous than the entire considered DNA sequence. Although this observation does not give us the number of segments actually present, it seems interesting that the number of sufficiently homogeneous segments found by IsoPlottor is almost the same as the number of segments identified with the D_{JS} criterion.

Table 3. Run times (in seconds) of the B-SMUCE algorithm when applied to sequences of different length taken from the human chromosome 1. The computations were been carried out on a single cluster core with 2.6 Ghz and 8 GB RAM. The results are reported for $\alpha = 0.05$ and for bins of lengths 32 and 10. For $\alpha = 0.01$, very similar run times have been obtained.

sequence length	10^5	$2{\cdot}10^5$	$5 \cdot 10^5$	10^6	$5 \cdot 10^6$	10^7
run time bin size 32	0.37	1.0	3.3	9.4	51.9	102.9
run time bin size 10	1.7	4.0	12.3	30.7	169.7	384.0

Finally we considered the region between 50Mb and 60Mb on the human chromosome 1. Here, we tried bins both of size 10 and 32. It turned out that with the finer partition slightly more segments were detected than with the larger bins of size 32, although the difference (1096 versus 1041) was not very large. It seems plausible that fine scale variation can be detected more easily with shorter bins.

To illustrate the run time behavior of the B-SMUCE algorithm with our default significance threshold $\alpha = 0.05$, we considered several shorter sequences taken from the above mentioned 10Mb DNA sequence. Table 3 gives the run times of our algorithm (in seconds) in dependence of the sequence length.

To give an idea about the run-times of D_{JS} and IsoPlotter, we applied them on the same sequence pieces. With the standard options (bin size: 32, shortest detectable domain size: 3008) the run times for the longest sequence (10^7 bases) were 6.2 sec. (D_{JS}) and 9.6 sec. (IsoPlotter). Notice however that B-SMUCE is designed to detect segments of any length, and the shortest segment detected by B-SMUCE in the context of our run time analysis was 80 bases long. We therefore also recorded the run times for D_{JS} and IsoPlotter with the minimum segment length changed from 3008 to 80. For a bin width of 32 and a sequence length of 10^7, the run time for D_{JS} remained unchanged, whereas the run time for IsoPlotter increased to 329.1 sec.

For the human genome data considered here, a cross check with the genome annotation revealed that several segments have an interpretation in terms of genes/exons, repetitive elements or CpG islands. As the GC content may depend on several functional and evolutionary factors, we do not expect simple explanations for many of the identified segments. Nevertheless, we explore the overlap of the identified segments with available annotation in the supplement.

4. **Conclusion**

We introduced a new method (B-SMUCE) for the segmentation of biological sequences. The segmentation is with respect to a binary response; here, we have considered GC content but it might be interesting to apply the method to other applications involving binary responses (such as ancestral/derived state of alleles in population genetic applications). Our approach provides precise statistical error control, and will produce a parsimonious segmentation that does not contain more segments than there actually are with a user specified preassigned probability of $1-\alpha$. A comparison under the benchmark scenarios taken from [18] suggests that the proposed method B-SMUCE is more accurate than previously proposed approaches.

Interestingly, the difference to the popular Jensen-Shannon criterion in terms of the number of detected segments has been particularly large for the human data.

Acknowledgement

We are grateful to Dr. Marlies Dolezal for her helpful comments. The research of A.M and H.S. is supported by DFG FOR 916, SFB 803 and the Volkswagen Foundation. The work by A.F. is supported by the FWF (Vienna Graduate School of Population Genetics).

References

[1] Amit, M., Donyo, M., Hollander, D., Goren, A., Kim, E., Gelfman, S., Lev-Maor, G., Burstein, D., Schwartz, S., Postolsky, B., Pupko, T., Ast, G. (2012), Differential GC content between exons and introns establishes distinct strategies of splice-site recognition, *Cell Rep.* **1(5)**, 543–556.

[2] Benjamini, Y., Speed, T.P. (2012), Summarizing and correcting the GC content bias in high-throughput sequencing, *Nucleic Acids Res.* **40**, e72.

[3] Bernardi, G. (1989) The isochore organization of the human genome, *Annu. Rev. Genet.* **23**, 637–661.

[4] Bernardi, G. (1995) The human genome: organization and evolutionary history, *Annu. Rev. Genet.* **29**, 445–476.

[5] Bernardi, G. (2001) Misunderstandings about isochores. Part I, *Gene* **276**, 3–13.

[6] Boysen, L., Kempe, A., Liebscher, V., Munk, A., Wittich, O. (2009) Consistencies and rates of convergence of jump-penalized least squares estimators, *Ann. Statist.* **37**, 157–183.

[7] Braun, J.V., Müller, H.G., (1998), Statistical methods for DNA segmentation, *Stat. Sci.* **13**, 142–162.

[8] Braun, J.V., Braun, R.K., Müller, H.G. (2000) Multiple change-point fitting via quasi-likelihood, with application to DNA sequence segmentation, *Biometrika* **87**, 301–314.

[9] Cristianini, N. and Hahn, M.W. (2007) Computational Genomics, *Cambridge University Press, Cambridge, UK.*

[10] Churchill, G.A. (1989) Stochastic models for heterogeneous DNA sequences, *B. Math. Biol.* **51**, 79–94.

[11] Churchill, G.A. (1992) Hidden Markov chains and the analysis of genome structure, *Comput. Chem.* **16**, 107–115.

[12] Clay, O., Carels, N., Douady, C., Macaya, G., Bernardi,G. (2001) Compositional heterogeneity within and among isochores in mammalian genomes. I. CsCl and sequence analyses, *Gene* **276**, 15–24.

[13] Cohen, N., Dagan, T., Stone, L., Graur, D. (2005) GC composition of the human genome: in search for isochores, *Mol. Biol. Evol.* **22**, 1260–1272.

[14] Cuny, G., Soriano, P., Macaya, G., Bernardi, G. (1981) The major components of the mouse and human genomes. 1. preparation, basic properties and compositional heterogeneity, *Eur. J. Biochem. / FEBS J.* **115**, 227–233.

[15] Davies, L., Höhenrieder, C. and Krämer, W. (2012) Recursive computation of piecewise constant volatilities, *Comput. Stat. Data An.* **11**, 3623–3631.

[16] Dümbgen, L., Spokoiny, V.G. (2001) Multiscale testing of qualitative hypotheses, *Ann. Statist.* **29**, 124–152.

[17] Dümbgen, L., Walther, G. (2008) Multiscale inference about a density, *Ann. Statist.* **36**, 1758–1785.

[18] Elhaik, E., Graur, D., Josić, K. (2010a) Comparative testing of DNA segmentation algorithms using benchmark simulations, *Mol. Biol. Evol.* **27**, 1015–1024.

[19] Elhaik, E., Graur, D., Josić, K. (2010b) Identifying compositionally homogeneous and nonhomogeneous domains within the human genome using a novel segmentation algorithm, *Nucleic Acids Res.* **38**, e158.

[20] Fickett, J.W., Torney, D.C., Wolf, D.R. (1992) Base compositional structure of genomes, *Genomics* **13**, 1056–1064.

[21] Frick, K., Munk, A., Sieling, H. (2013) Multiscale Change-Point Inference, *ArXiv* e-prints, 1301.7212, discussion paper with rejoinder by the authors. *J. Roy. Stat. Soc. B*, to appear.

[22] Friedrich, F., Kempe, A., Liebscher, V., Winkler, G. (2008) Complexity Penalized M- Estimation: Fast Computation, *J. Comput. Graph. Stat.* **17**, 201–224.

[23] Freudenberg, J., Wang, M., Yang, Y., Li, W. (2009) Partial correlation analysis indicates causal relationships between GC-content, exon density and recombination rate in the human genome, *BMC Bioinformatics* **10**, (Suppl 1) 66.

[24] Fullerton, S.M., Carvalho, A.B., and Clark, A.G. (2001) Local Rates of Recombination Are Positively Correlated with GC Content in the Human Genome, *Mol. Biol. Evol.* **18**, 1139–1142.

[25] Galtier, N., Piganeau, G., Mouchiroud, D., and Duret, L. (2001) GC-content evolution in mammalian genomes: the biased gene conversion hypothesis, *Genetics* **159**, 907–911.

[26] Keith, J.M. (2006) Segmenting Eukaryotic Genomes with the Generalized Gibbs Sampler, *J. Comput. Biol.* **13**, 1369–1383.

[27] Killick, R., Fearnhead, P., Eckley, I.A. (2012) Optimal detection of changepoints with a linear computational cost, *J. Am. Stat. Assoc.* **107**, 1590–1598.

[28] Oliver, J.L., Román-Roldán, R., Pérez, J., Bernaola-Galván, P. (1999) SEGMENT: identifying compositional domains in DNA sequences, *Bioinformatics* **15**, 974–979.

[29] Risso, D., Schwartz, K., Sherlock, G., Dudoit, S. (2011) GC-Content Normalization for RNA-Seq Data, *BMC Bioinformatics* **12**, 480.

[30] Sueoka, N. (1962) On the genetic basis of variation and heterogeneity of DNA base composition, *Proc. Natl. Acad. Sci. USA* **48**, 582–592.

[31] Walther, G. (2010) Optimal and fast detection of spatial clusters with scan statistics, *Ann. Statist.* **38**, 1010–1033.

[32] Winkler, G., Liebscher, V. (2002) Smoothers for discontinuous signals, *J. Nonparametr. Stat.* **14**, 203–222.

[33] Yao, Yi-Ching (1988) Estimating the number of change-points via Schwarz' criterion, *Statist. Probabil. Lett.* **6**, 181–189.