Molecular Biology and Evolution

Comparative Testing of DNA Segmentation Algorithms Using Benchmark Simulations

Eran Elhaik,*[,1] Dan Graur,[1] and Krešimir Josić[2]

[1]Department of Biology & Biochemistry, University of Houston

[2]Department of Mathematics, University of Houston

***Corresponding author:** E-mail: eelhaik@gmail.com.

Associate editor: Hideki Innan

Abstract

Numerous segmentation methods for the detection of compositionally homogeneous domains within genomic sequences have been proposed. Unfortunately, these methods yield inconsistent results. Here, we present a benchmark consisting of two sets of simulated genomic sequences for testing the performances of segmentation algorithms. Sequences in the first set are composed of fixed-sized homogeneous domains, distinct in their between-domain guanine and cytosine (GC) content variability. The sequences in the second set are composed of a mosaic of many short domains and a few long ones, distinguished by sharp GC content boundaries between neighboring domains. We use these sets to test the performance of seven segmentation algorithms in the literature. Our results show that recursive segmentation algorithms based on the Jensen–Shannon divergence outperform all other algorithms. However, even these algorithms perform poorly in certain instances because of the arbitrary choice of a segmentation-stopping criterion.

Key words: isochores, GC content, segmentation algorithms, Jensen–Shannon divergence statistic, entropy, genome composition, benchmark simulations.

Research article

Introduction

In 1976, Bernardi and colleagues (Macaya et al. 1976) put forward a theory concerning the compositional makeup of vertebrate genomes. This description, later christened "the isochore theory" (Cuny et al. 1981), was based on buoyant density data of melted DNA fragments. Isochores were defined as genomic fragments longer than 300 kb that are "relatively homogeneous" in their guanine and cytosine (GC) composition (Macaya et al. 1976; Cuny et al. 1981; Bernardi et al. 1985; Bernardi 2000; Clay et al. 2001; Pavlicek et al. 2002). For example, the human genome was described as a mosaic of isochores belonging to five families: L1, L2, H1, H2, and H3, whose corresponding ranges of GC contents were said to be less than 38%, 38–42%, 42–47%, 47–52%, and more than 52%, respectively (Bernardi 2001).

With the advent of genome sequencing and the availability of whole-genome sequences, the isochore theory was challenged (Nekrutenko and Li 2000; Häring and Kypr 2001; International Human Genome Sequencing Consortium 2001; Cohen et al. 2005). Determining whether isochores exist requires an unambiguous definition of what constitutes an isochore. In the absence of an agreed-upon definition of "relative homogeneity," the concept of "isochore" has frequently been revised following conflicting results by different segmentation algorithms. Much of the controversy over the existence of isochores appears to be the result of the difficulties in identifying compositionally homogeneous domains within genome sequences using existing methods.

Many methods for isochore detection in genomic sequences have been proposed. These methods use the genomic sequence as sole input and partition the sequence into compositionally homogeneous domains according to predefined criteria. Segmentation methods include nonoverlapping, sliding window methods (Bernardi 2001; Clay et al. 2001; International Human Genome Sequencing Consortium 2001; Guéguen 2005; Costantini et al. 2006), walking Markov models (Fickett et al. 1992), hidden Markov models (Churchill 1989, 1992; Guéguen 2005), recursive segmentation methods (Bernaola-Galván et al. 1996; Oliver et al. 1999, 2004; Li 2001a, 2001b, 2002; Cohen et al. 2005), Bayesian methods (Husmeier and Wright 2002; Boys and Henderson 2004; Guéguen 2005), and least squares estimation methods (Haiminen and Mannila 2007). Other methods may be found in Braun and Müller (1998).

Unfortunately, these methods yielded inconsistent results (see fig. 1 in Schmidt and Frishman 2008). To illustrate this problem, we present the segmentation results obtained by applying seven commonly used segmentation algorithms on a 1-Mb sequence of human chromosome 1 (fig. 1). Some algorithms partitioned the DNA sequence into a few long domains, whereas others yielded many short ones. Similarly, some algorithms detected domains that are nearly equal in length, whereas the domains identified by others exhibited no characteristic length scale. Thus, at the present time, one cannot even state with any degree of certainty where an isochore starts and where it ends, let alone describe the isochoric structure of the genome.

As stated above, algorithm validation in the context of isochores is problematic because of the vagueness of the definition of isochores. It has, thus, become a common practice to "test" the effectiveness of a segmentation algorithm by its ability to identify previously "defined"

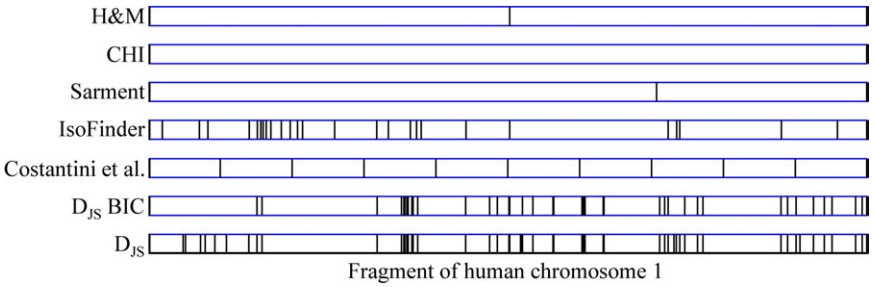


FIG. 1. Different compositional domains found on a 1-Mb fragment from human chromosome 1 by different segmentation algorithms (see Algorithms in Materials and Methods). To avoid clutter, only long domains (>10 kb) are shown.

isochores, such as those claimed to exist in the human major histocompatibility complex locus (Li 2001a, 2001b, 2002; Pavlicek et al. 2002; Li et al. 2003; Oliver et al. 2004; Guéguen 2005; Haiminen and Mannila 2007; Haiminen et al. 2007), where sharp GC content transitions have been reported (Fukagawa et al. 1995; Eyre-Walker and Hurst 2001). However, this approach is of limited use because a test performed in one region of the genome may not be applicable to other regions. Another common testing practice is to partition the human genome and show that isochores exist (Bernardi 2001; Wen and Zhang 2003; Oliver et al. 2004; Haiminen and Mannila 2007). This approach is circular, as isochores are postulated to be regions detected by an isochore-finding algorithm. Recent approaches (Haiminen et al. 2007; Schmidt and Frishman 2008) endeavor to find consensus isochores by integrating results of different segmentation methods. Although this approach may seem reasonable, we note that using questionable methods before testing them on benchmark simulations does not produce indisputable results.

In this study, we tried to avoid some of the semantic confusion regarding isochores and focused on quantifying the reliability of different segmentation algorithms. For this purpose, we generated genomic sequences containing a predetermined number of compositionally homogeneous domains, each separated from adjacent domains by sharp changes in GC content. Regardless of one's point of view, these simulated domains should be recognized as isochores. Hence, we set up a useful benchmark that enables us to test the ability of different segmentation algorithms to identify isochores. We did not attempt to capture all the statistical complexity of the genome. Our goal was merely to construct a very simple model of a genome consisting of compositionally homogeneous domains. Segmentation algorithms that do not perform well in this simple setting cannot be expected to perform well in real genomes. An additional goal was to identify intrinsic biases and systematic errors in the segmentation methods to help the development of improved segmentation algorithms.

Materials and Methods

Data Retrieval and Filtering

We downloaded human chromosome 1 (build 36.2) from the NCBI ftp Web site (ftp://ftp.ncbi.nlm.nih.gov/genomes/) and excluded Ns (unknown nucleotides or nulls).

Simulating Sequences with Fixed-Sized Homogeneous Domains

We tested the capability of different algorithms to detect compositionally homogeneous domains in two different simulation sets. In the first simulation set, we generated genomic sequences of different lengths, each composed of ten fixed-sized homogeneous domains distinct from one another in their GC composition. We manipulated two variables for every simulated genomic sequence: the size of the homogeneous domains and the between-domain variability σ_{sequence}, defined as the standard deviation (SD) of the mean GC content of all domains in a sequence.

For each sequence, domain lengths were selected from nine possible lengths: 10 kb, 50 kb, 100 kb, 200 kb, 300 kb, 500 kb, 1 Mb, 5 Mb, and 10 Mb. The between-domain variability (σ_{sequence}) was selected from five possible values: 0, 2.5, 5, 7.5, and 10, where 0 indicates a completely uniform sequence with indistinguishable domains and 10 indicates a high between-domain variation (fig. 2). The sequence mean GC content ($\mu_{\text{GCsequence}}$) was randomly drawn from a uniform distribution ranging from 10 to 90. For every domain in the sequence, we determined the mean GC content (μ_{GCdomain}) using the following equation:

$$\mu_{\text{GCdomain}} = \mu_{\text{GCsequence}} + (\sigma_{\text{sequence}} \times R_{\text{normal}}). \quad (1)$$

Here, R_{normal} is a random variable drawn from a normal distribution with a mean 0 and a SD 1. We used only μ_{GCdomain} within the range of 0 and 100. The nucleotides within every domain were uniformly distributed. Our data set consisted of 100 matrices composed of sequences with nine possible domain lengths and five possible between-domain variability values (a total of 4,500 sequences).

Simulating Chromosomal Sequences with Different-Sized Homogeneous Domains

In our second simulation set, we generated 200 chromosomal sequences consisting of few long homogeneous domains (>10 kb) and many short domains (1 kb). One hundred of the chromosomal sequences were composed of 100 long domains with lengths (L_{domain}) drawn from a normal distribution using

$$L_{\text{domain}} = \mu + (\sigma \times R_{\text{normal}}), \quad (2)$$

with mean (μ) of 10,000 and SD (σ) of 500,000. The other 100 chromosomal sequences were created similarly with long

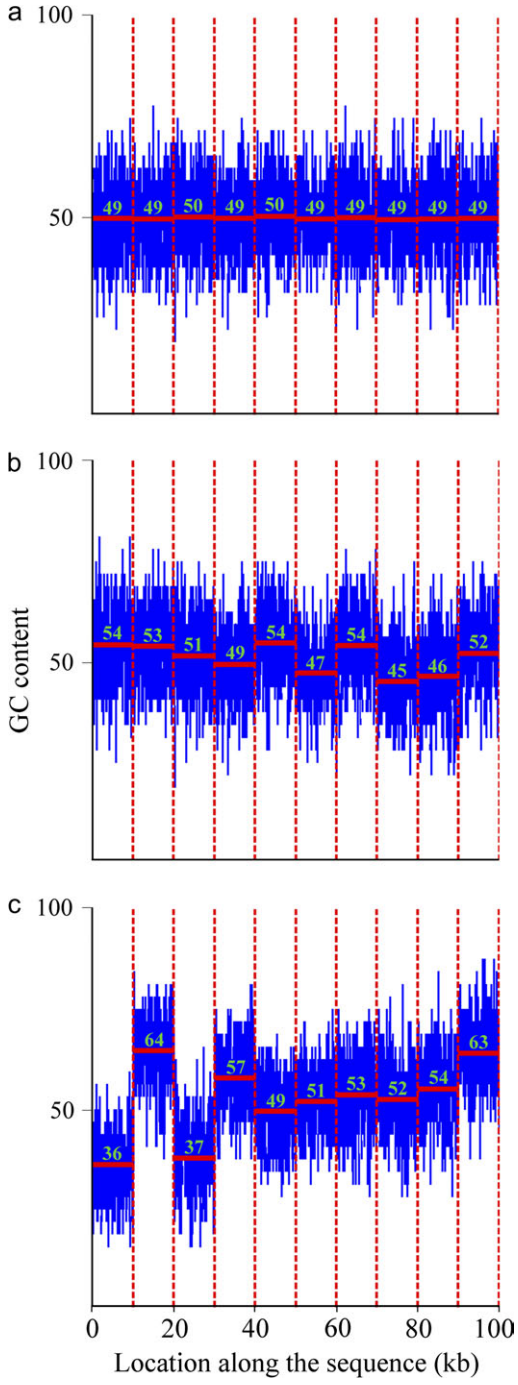


FIG. 2. Spatial distributions of GC content in 32-bp nonoverlapping windows for three simulated genomic sequences of size 100 kb. The sequences differ in their between-domain variability: (a) $\sigma_{sequence} = 0$, (b) $\sigma_{sequence} = 2.5$, and (c) $\sigma_{sequence} = 10$. Sequences contain ten equally sized homogeneous domains (vertical dotted lines). The mean GC content of each domain is shown above the horizontal red lines.

domain lengths drawn from a power-law probability distribution $p(x) = Cx^{-\alpha}$. Sequences were generated with $\alpha = 1.55$ and $x_{min} = 10$ kb using the Matlab script randht.m (version 1.0) (Clauset et al. 2009) provided at http://www.santafe.edu/~aaronc/powerlaws/. The exponent α defining the power-law distribution was determined by partitioning the human genome with the D_{JS} algorithm (see Algorithms) and plotting the homogeneous domains on a log-log scale histogram using logarithmic binning (Newman 2005). We note that power-law distributions are reported to characterize mammalian genomes, and the choice of exponent α falls within the range commonly used in the literature (e.g., Clay et al. 2001; Cohen et al. 2005). The total length of the short domains was arbitrarily set to 1% of the chromosomal sequence length. All domain positions were selected randomly. The mean GC content of the long domains was determined using equation (1) with $\mu_{GCsequence} = 40$ and $\sigma_{sequence} = 5$. The nucleotide distribution within each domain was uniform. The GC contents of short domain pairs were chosen randomly from uniform distributions ranging between 0% and 40% and between 60% and 100% to form clusters of alternating GC content. Supplementary table S1 presents some statistical information regarding the chromosomal sequences generated in the second simulated set. We limited the discussion to chromosomal sequences created with domain lengths drawn from a power-law distribution. Results were robust to the choice of distribution from which domain lengths were drawn.

Prior to employing the algorithms on the simulated genomic sequences and chromosomal sequences, we evaluated the differences in the mean GC content between all adjacent domains. Although domains were created independently from one another, occasionally two or more adjacent domains may have had very similar GC content by chance. Domains with a difference between their mean GC content smaller than a predetermined cutoff value of 1% were concatenated.

GC Content Calculation

Each sequence was divided into 32-bp nonoverlapping windows in length. We chose to work on short windows rather than on single nucleotides to save computation time without sacrificing accuracy (Cohen et al. 2005). In addition, we also note that using large window sizes (1 kb or longer) biases the segmentation, whereas using the small window size minimizes the segmentation bias (Dagan T, personal communication). The GC content for each window was calculated. All segmentation algorithms were applied on the resulting sequences of GC frequencies.

Algorithms

We tested seven segmentation algorithms from the literature. Below, we describe only the main ideas behind each algorithm. In most cases, the details and the implementation are presented in the original papers. We could not obtain the original code for one of the programs and reconstructed the program to the best of our ability on the basis of its description. For several algorithms, it is unclear which parameters were to be used under which circumstances. In such cases, we used the default parameters recommended by the authors. Unless obtained from the authors, algorithms were implemented in Matlab 7.5. These scripts are freely available from our Web site http://nsm.uh.edu/~dgraur/eran/simulation/main.htm.

D_{JS}. In this method (Cohen et al. 2005), sequences are recursively partitioned by maximizing the difference in GC content between adjacent subsequences, as measured by

the Jensen–Shannon divergence statistic (D_{JS}) (Lin 1991). The D_{JS} statistic is calculated over all possible partitioning points, and the sequence is partitioned at the position of maximum D_{JS}. The process of segmentation is terminated when the maximal D_{JS} value is smaller than a predetermined threshold (for full description, see Supplementary Material). We used a threshold of 5.8×10^{-5} (Dagan T, personal communication).

D_{JS} BIC. In this method (Li 2001a; Li et al. 2002), sequences are recursively partitioned by maximizing the difference in GC content between adjacent subsequences measured by the Jensen–Shannon divergence statistic (D_{JS}) (Lin 1991). The "segmentation strength" s is a predefined measure of stringency for the segmentation that considers the sequence length and the maximum D_{JS} value (for full description, see Supplementary Material). The process of segmentation continues as long as s is larger than a predetermined threshold s_0. We used $s_0 = 0$.

H&M. In Haiminen and Mannila's (2007) method, the sequence is partitioned into 100-kb nonoverlapping windows and their GC content is calculated. The algorithm divides the sequence into a predetermined number of domains (k) so that the sum of squares of the Euclidean distance between the GC content of the segments and the sequence GC content is minimized. A serious disadvantage of this algorithm is its quadratic computation time, which precluded the testing of its performance on long sequences ($\geq$100 kb). In addition, determining a realistic value for k, as far as real-life sequences are concerned, is not a straightforward task; however, in our simulation sets, it was simply the number of predetermined domains. We used the segmentation code provided by the authors at http://dx.doi.org/10.1016/j.gene.2007.01.028.

Sarment. This package implements an algorithm that partitions the sequence into k domains using maximal predictive partitioning (Guéguen 2005). As in the previous case, the final number of domains (k) has to be determined a priori. Again, in real-life situations, the value of k is unknown; however, in our simulation, the value of k is known. We used the segmentation code (version 4) provided by the authors at http://pbil.univ-lyon1.fr/software/sarment/ (file tar.gz).

Costantini et al. We used the sliding window algorithm as described by Costantini et al. (2006). In this method, the sequence is partitioned into 100-kb nonoverlapping windows. The GC variation within every window is calculated by the GC content SD. The windows are scanned for differences in the SD between every two adjacent windows. Adjacent windows with differences between their SD below a given threshold of 1% were considered concatenated domains (Costantini et al. 2006). We did not observe any cases that would justify using smaller thresholds.

Compositional Heterogeneity Index. This method (Nekrutenko and Li 2000) uses a divide-and-conquer approach (see Cormen et al. 1990). The sequence is partitioned to n windows of length l, and the compositional heterogeneity index (CHI) measure is calculated for each window. Sequences are then recursively partitioned by maximizing the difference in GC content between adjacent subsequences, measured by the CHI measure

$$\mathrm{CHI} = \frac{\frac{1}{n-1}\sum_{i=2}^{n}|\mathrm{GC}_i - \mathrm{GC}_{i-1}|}{\sqrt{\frac{P(1-P)}{L}}}, \qquad (3)$$

where the GC content of each nucleotide in the window is GC_i and the mean GC content of the window is P. The process of segmentation is terminated when the maximized CHI is smaller than a given threshold. In order to estimate the halting threshold parameter, 100,000 random sequences, each 1-Mb long, were drawn from a uniform distribution. Each of these sequences was partitioned into two at a random point, and the CHI value for each segment was calculated. We followed the procedure of Cohen et al. (2005) and chose a CHI value corresponding to the upper 5% of the cumulative CHI distribution. We tested a wide range of window sizes (l = 2, 3, 4, 5, and 10). The algorithm appeared to perform best with window size of 2.

IsoFinder. We were not successful in obtaining the code for this program from the corresponding author and reconstructed the IsoFinder algorithm from the original description in Oliver et al. (2004). (We note that a "canonical" IsoFinder program does not exist. The description of IsoFinder was published in 2004, but the Fortran program that was supposed to be available online was not! The program was revised and rerevised periodically for about 4 years; however, the changes were documented neither in the literature nor online. Currently, a new "stand-alone" IsoFinder program is posted online. This program yields results that are essentially indistinguishable from D_{JS}, with the exception that short segments are arbitrarily concatenated into longer domains, thereby artificially increasing the sizes of the "isochores." Unfortunately, no detailed description is available for this "new improved" IsoFinder, except for a short unreviewed note in arxiv [http://arxiv.org/abs/0806.1292]. For all intents and purposes, the 2004 IsoFinder no longer exists. Under these circumstances, the best we could do is to reconstruct the algorithm based on the descriptions in the original paper. We compared our code with that of IsoFinder [as of January 2007] and found that our version of the algorithm detected nearly all the borders detected by this IsoFinder but also other borders not reported by IsoFinder. Because we wanted to include this algorithm in our study, while at the same time, we were reluctant to keep up with a "moving target," we chose to use our own code. It is available at http://nsm.uh.edu/~dgraur/eran/simulation/main.htm.)

Our IsoFinder algorithm uses a sliding pointer that moves from left (5′) to right (3′) of the sequence. At each point, the mean GC contents to the left and the right of the pointer are compared using the t-statistic. At the maximum t-statistic point (t_{filt}), the algorithm compares the two subsequences to the left and to the right of the point as follows: both subsequences are divided into windows of

size l_0 and their window GC contents are compared using Student's t-test to obtain t_{filt}. The distribution of t_{filt} values, $P(\tau = t_{filt})$, was obtained using Monte Carlo simulations (Bernaola-Galván et al. 2001). The statistical significance, $P(\tau)$, of the possible partitioning point with $t_{filt} = \tau$ is defined as the probability of obtaining the value τ or lower values within a random sequence. If $P(\tau)$ exceeds a predetermined threshold P_0, the sequence is partitioned into two subsequences and the procedure is repeated recursively. We followed Oliver et al. (2004) and used $l_0 = 3{,}000$ bp and $P_0 = 95\%$.

Comparing Different Segmentational Results with Real Genomic Data

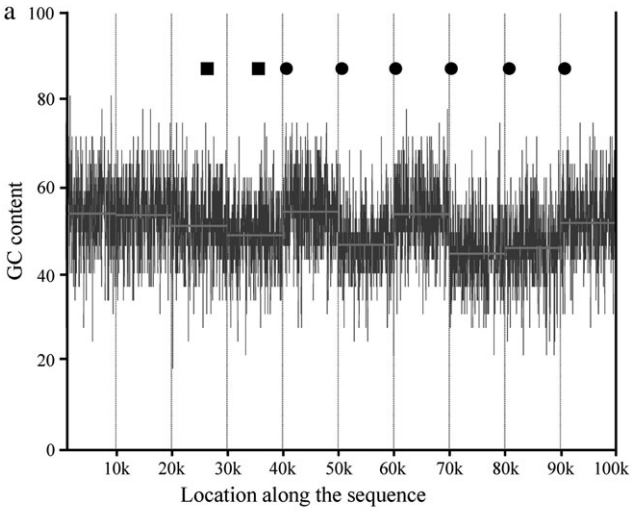
We used the seven segmentation algorithms to partition a 10-Mb fragment of the human chromosome 1, starting with position 1. It was sufficient to partition one chromosome to reveal the differences between the different algorithms. For the Sarment and H&M algorithms, the maximum k value for which results were obtained was $k = 10$. All other algorithms were used with default parameters.

We analyzed the results in two ways: First, for every algorithm, we calculated the proportion of domain borders that were similar to those inferred by other algorithms. Two borders inferred by different algorithms were considered similar if the distance between their chromosomal locations was smaller than 5 kb. Next, for every algorithm, we calculated the proportion of homogeneous domains. In accordance with the homogeneity test (see Homogeneity Test), we analyzed only domains larger than 10 kb.

Data Analysis

There are two possible ways to assess true/false positives/negatives. One can consider either the borders of the domains or the domains themselves. Because the algorithms determine partition points, it seems natural to use the borders as the variable. However, defining borders as true/false positive/negative would force us to reward partial domain detection, which is not really meaningful. For example, consider a sequence of ten equally sized domains with nine internal borders and an algorithm that detects every other border (two, four, six, and eight). The algorithm sensitivity would be 0 in our method because it did not detect any homogeneous domain. In fact, the algorithm only detected heterogeneous domains—the opposite of what we are looking for. By using borders, the algorithm sensitivity would have been nearly 50%, that is, the algorithm would wrongly not be considered a complete failure.

A true positive (TP) border is a border that was identified at a maximum distance of 5 kb or 5% of its domain size from a predetermined border, whichever is smaller. A domain that has both its borders as TP inferences is considered a correctly inferred domain (fig. 3*a*). To evaluate the segmentation results, we used two statistics: sensitivity and precision rate (fig. 3*b*). Sensitivity is the proportion of correctly inferred domains out of all predetermined domains. The precision rate statistic quantifies the probability of the positive prediction of the algorithm, that is, the proportion of correctly inferred domains out of all domains reported by the algorithm. To test whether the differences between the algorithms are significant, we use the one-tailed Wilcoxon rank-sum test with $P < 0.05$ (Sokal and Rohlf 1995, p. 427–431) and the Bonferroni correction for multiple tests (Sokal and Rohlf 1995, p. 240, 702–703).



b

		Predetermined border exists in proximity: Present (predetermined border exists)	Predetermined border exists in proximity: Absent (predetermined border does not exist)	
Algorithm segmentation outcome	**Positive** (reporting inferred border)	**TP** (Inferred border matches a predetermined border)	**FP** (Inferred border reported where a predetermined border does not exist)	**Precision rate** TP/(TP+FP)
	Negative (not reporting inferred border)	**FN** (Inferred border was not reported where a predetermined border exist)	**TN** (Inferred border was not reported where a predetermined border does not exist)	**Negative predictive value** TN/(TN+FN)
		Sensitivity TP/(TP+FN)	**Specificity** TN/(FP+TN)	

Fig. 3. (*a*) The spatial distribution of GC content for a simulated genomic sequence originally made of ten homogeneous domains, each 10 kb in length (vertical bars). The first two domains have been concatenated because, by chance, the difference in their mean GC content was lower than the cutoff value. Correct inferences (circles) and incorrect inferences (squares) are marked. Here, six out of nine domains were detected accurately. (*b*) The contingency table used to calculate sensitivity and precision rate. For the example in (a), the table values are TP = 6, FP = 3, and FN = 3. The sensitivity and the precision rate are 66%.

In figure 3*a*, we illustrate partitioning of a 100-kb simulated genomic sequence composed of ten equally sized homogeneous domains. The contingency table in figure 3*b* further elaborates the terminology used in the analyses. In this particular example, the difference between the mean GC content of the first two domains was below 1%; therefore, they are treated as a single domain. Thus, only nine domains and eight internal borders are considered. Consider a hypothetical algorithm applied to the above sample sequence. Suppose this algorithm identified

eight partitioning points, that is, borders within the sequence borders, dividing the sequence into nine domains. Compared with the built-in domains, the first three were not detected (false negative, FN = 3) and the last six were detected accurately (TP = 6). In addition, three domains were detected erroneously because at least one of their borders was not in the proximity (<5 kb) of any other predetermined border (false positive, FP = 3). In cases in which the domains were longer, true negative (TN) became much larger effectively dwarfing TP, FN, and FP. Thus, we did not use TN nor compute any statistic, for example, specificity, which used TN. For our hypothetical example, the sensitivity TP/(TP + FN) and precision rate TP/(TP + FP) are therefore calculated to be 66%.

For the first simulation set of genomic sequences, we analyzed the algorithm results in two dimensions: in sequences with similar between-domain variability and different domain lengths in one dimension and in sequences with different between-domain variability and similar domain lengths in the other. We calculated the sensitivity and precision rate statistics for each sequence in each dimension and plotted the average sensitivity versus the average precision rate. We chose this presentation form because it decouples the algorithm performances and the error costs by depicting the trade-offs between sensitivity and precision rate. Each graph space distinguishes among several outcomes. For example, the point (0, 0) represents absolute failure to detect anything and the point (100, 100) represents perfect domain detection where all inferred domains match all predetermined domains.

To evaluate the results for the chromosomal sequences in the second set, we used three statistics: short domain sensitivity, long homogeneous domain sensitivity, and precision rate. Here, the two sensitivity statistics quantify how well the algorithm identifies short and long homogeneous domain borders. The maximum distance between a predetermined border of a short domain and a correctly inferred short domain border was conservatively set to 96 bp.

Homogeneity Test

To compare the within-domain relative homogeneity of a domain with that of the sequence on which it resides, we applied the *F*-test to the variance in GC content of the two (Zar 1999). We followed the procedure proposed by Cohen et al. (2005) and divided every inferred domain into 2,048-bp nonoverlapping windows. The choice of window size was supposed to ensure robust results with short domains of length 10 kb and above. The GC content for each window was calculated for the inferred domain in question and for the entire sequence within which the domain resides. Because the *F*-test assumes the data are normally distributed, we applied the arcsine-root transformation to the GC content values of the windows within each domain (and sequence) before calculating the variance (Sokal and Rohlf 1995). Finally, we applied the false discovery rate correction (Benjamini and Hochberg 1995) for multiple comparisons.

Table 1. Percentage of Inferred Domain Borders That Are in Close Proximity (≤5 kb) to Borders Found by Other Segmentation Algorithms (see text).

		D_{JS}			Costantini		
Algorithms	**D_{JS}**	**BIC**	**H&M**	**Sarment**	**et al.**	**CHI**	**IsoFinder**
D_{JS}	—	84	2	3	13	0	35
D_{JS} BIC		—	3	3	13	0	35
H&M			—	56	1	0	3
Sarment				—	0	0	3
Costantini et al.					—	0	11
CHI						—	0
IsoFinder							—

A one-tailed *F*-test with a null hypothesis H_0: $\sigma^2_{segment} \geq \sigma^2_{sequence}$ and an alternative hypothesis H_1: $\sigma^2_{segment} < \sigma^2_{sequence}$ was applied with $n_1 - 1$ and $n_2 - 1$ degrees of freedom, where n_1 and n_2 are the numbers of windows in the domain and in the corresponding sequence, respectively. If the variance over a domain was found to be significantly smaller ($P < 0.01$) than that of the corresponding sequence, then the domain was considered more homogeneous than the sequence.

Computation Time

We calculated the algorithm mean computation times for sequences of low between-domain variability (5) and three sizes: 100 kb, 1 Mb, and 10 Mb.

Results

Comparison of Segmentation Results Obtained by Different Algorithms

We partitioned a 10-Mb fragment of human chromosome 1 to illustrate the difference in the inferred domains found by the different algorithms (fig. 1). First, for every algorithm, we measured the proportion of inferred domains that were similar to those inferred by other algorithms (table 1), that is, the extent to which the results of various algorithms agree. The D_{JS} and D_{JS} BIC algorithms produced the most similar results (84%), followed by the Sarment and H&M algorithms (60%). The similarity between the results of all other algorithms was low (<50%).

The number of homogeneous domains (>10 kb) inferred by the different algorithms varied greatly (0–192), although their relative proportion out of all inferred domains was mostly high (table 2). However, we note that for sufficiently long domains (>50 kb), the homogeneity test loses significance (Cohen et al. 2005). Therefore, segmentation algorithms that infer only a few long domains may be erroneously considered precise.

Comparison of Algorithm Performances on the First Simulated Data Set with Fixed-Sized Homogeneous Domains

To gauge domain relative homogeneity in our simulated data set, we compared the variance in GC content within each domain with that of the sequence on which it resides. As expected, sequences generated with between-domain

Table 2. Partitioning Results for a 10-Mb Fragment of Human Chromosome 1.

Algorithms	Inferred Domains	Homogeneous Domains (percentage out of inferred domains)
D_{JS}	216	192 (89)
D_{JS} BIC	189	168 (89)
H&M	10	10 (100)
Sarment	10	10 (100)
Costantini et al.	98	81 (83)
CHI	1	0 (0)
IsoFinder	102	79 (78)

variability of 0 had almost no homogeneous domains, which made them nearly indistinguishable. Only 12–38% of the domains of lengths 10 kb were found to be homogeneous (see Homogeneity Test). In other words, segmentation algorithms were not expected to detect more than 40% domains in the shortest sequences. The domains in all other sequences were homogeneous, and segmentation algorithms were expected to detect them.

Figure 4*a* shows the mean sensitivity versus precision rate statistics for algorithms applied to sequences of the first simulation set with similar between-domain variability and different domain lengths. To simplify the presentation, we show only three algorithms with distinct results: D_{JS} BIC, Costantini et al., and IsoFinder (for full results, see supplementary figs. S1 and S2). On average, the D_{JS} and D_{JS} BIC algorithms had the highest sensitivity (70–95%) and precision rate (80–87%, shown in supplementary figs. S1 and S2). The results of these two algorithms were nearly identical for all between-domain variability values and significantly better than those of other algorithms (Wilcoxon rank-sum, $P < 0.05$). The sensitivity of the IsoFinder algorithm increased (27–37%) for higher between-domain variability values; however, its precision rate decreased (38–22%). That is, the algorithm inferences improve with the distinguishability of domains; however, the proportion of correctly inferred domains out of all inferred domains decreases. The Costantini et al. algorithm had an average sensitivity and precision rate of 57% and 37%, respectively. Its performances were mostly unaffected by the change in the between-domain variability. The sensitivity and precision rate of Sarment and CHI algorithms were the lowest among all algorithms and were unaffected by the change in between-domain variability.

Figure 4*b* shows the mean sensitivity versus precision rate statistics for D_{JS} BIC, Costantini et al., and IsoFinder applied to sequences of the first simulation set with different between-domain variability and similar domain lengths (for full results, see supplementary figs. S3 and S4). As before, D_{JS} and D_{JS} BIC performed significantly better than all other algorithms with sensitivity values exceeding 95%, with the exception of 100-kb domains (Wilcoxon rank-sum, $P < 0.05$) (supplementary figs. S3 and S4). The Costantini et al. algorithm outperformed all other algorithms only in detecting predetermined domains of size 100 kb. Its performances are otherwise characterized by low precision

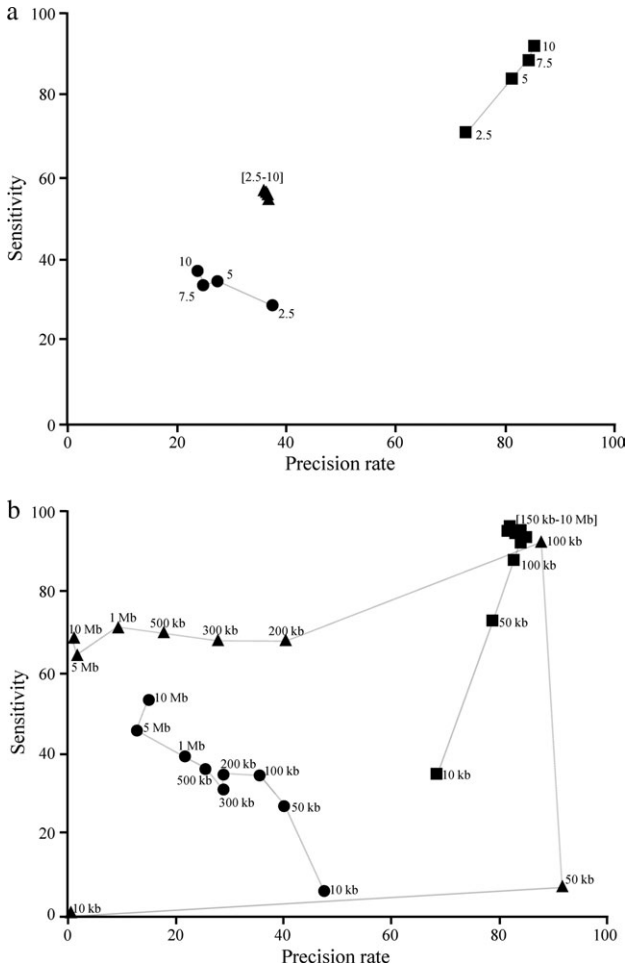


Fig. 4. Mean sensitivity versus precision rate for three algorithms: D_{JS} BIC (squares), Costantini et al. (triangles), and IsoFinder (circles) applied to sequences of the first simulation set. Results are presented for sequences consisting of (*a*) similar between-domain variability and different domain lengths and (*b*) different between-domain variability and similar domain lengths.

rates (0–40%) for nearly all other domain lengths (<50 kb and >100 kb). In other words, less than 40% of the algorithm inferences were correct for sequences consisting of homogeneous domains of various lengths. IsoFinder sensitivity increased almost linearly with the increase in domain sizes (5–52%); however, its precision rate decreased from 47% (10 kb) to 15% (10 Mb). The H&M algorithm had the largest variability in its results, which suggests that the algorithm may be too sensitive to variation in nucleotide composition.

Comparison of Algorithm Performances on the Second Simulated Data Set with Simulated Chromosomal Sequences

In our first simulation set, we considered genomic sequences consisting of ten fixed-sized homogeneous domains. We now test the algorithms on the second simulation set, which is a more realistic data set of chromosomal sequences consisting of varying domain lengths. An example of a simulated chromosomal sequence is shown in figure 5.

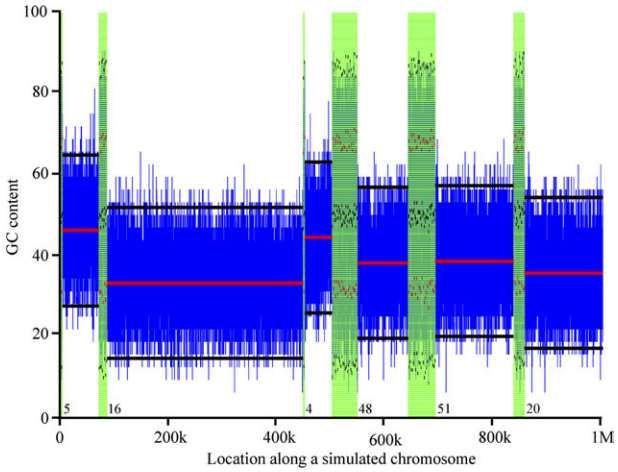


Fig. 5. The spatial distribution of GC content of 32-bp nonoverlapping windows along a fragment of 1 Mb in length from a simulated chromosomal sequence. The green lines represent the domain borders. The mean GC content and the GC content SDs of the domains are marked with red and black horizontal lines, respectively. The number of short domains is noted for each cluster.

The figure presents a 1-Mb-long chromosomal sequence with 139 short domains (1 kb) and 6 long homogeneous domains (>10 kb). The short domains are clumped together because of their large number (~4,000) compared with the long homogeneous domains (supplementary table S1).

In table 3, we summarized the three statistics obtained for each algorithm: short domain sensitivity, long homogeneous domain sensitivity, and precision rate. In agreement with our previous results, the D_{JS} and D_{JS} BIC algorithms significantly outperformed all other algorithms, although the differences between the two remained statistically insignificant (Wilcoxon rank-sum, $P < 0.05$). The Costantini et al., IsoFinder, and Sarment algorithms could not detect any domain border accurately, and nearly all their inferences were in error. The H&M and CHI algorithms were excluded from this analysis due to their large computation time requirement.

When tested for homogeneity, approximately 77% of the long domains drawn from a power-law distribution were found to be homogeneous. Although the test could not be applied to the short domains because of their size, they were distinguishable from one another because of their GC content.

Table 3. Relative Performances of Seven Segmentation Algorithms.

Algorithms	Short Domain Sensitivity (%)	Long Homogeneous Domain Sensitivity (%)	Precision Rate (%)
D_{JS}	96	87	98
D_{JS} BIC	95	66	98
H&M[a]	—	—	—
Sarment	0	2	3
Costantini et al.	0	~0	~0
CHI[a]	—	—	—
IsoFinder	0	5	3

[a] The algorithm's large computation time prevented us from testing it.

Table 4. Computation Times (in hours) for Different Segmentation Algorithms as a Function of the Sequence Length. Results Obtained from the Algorithm Mean Computation Times for Sequences of Low Between-Domain Variability (5) and Three Sizes: 100 kb, 1 Mb, and 10 Mb.

Algorithm	100 kb	1 Mb	10 Mb
D_{JS}	0.01	0.07	0.67
D_{JS} BIC	0.01	0.07	0.67
H&M	16.67	—[a]	—[a]
Sarment	0.01	0.06	0.44
Costantini et al.	<0.01	0.01	0.05
CHI	0.09	10.33	66.67
IsoFinder	0.01	0.66	10

[a] The extremely long computation time required for this algorithm prevented us from testing it.

Computation Times

Using sequences from the first simulation set, we estimated the algorithm computation times on sequences of varied lengths and between-domain variability value of 5. The results show that the fastest algorithms are Costantini et al. and Sarment, followed closely by D_{JS} and D_{JS} BIC (table 4). IsoFinder, CHI, and H&M algorithms required the largest computation time. It was impossible to estimate the computation time of H&M algorithm on the complete data set because of the long computation time; however, partitioning a 10-Mb fragment of human chromosome 1 took approximately 26 h.

Discussion

We used a series of benchmark sequences to evaluate which of the sequence segmentation algorithms can detect predetermined domain borders between compositionally homogeneous domains. The results show that the D_{JS} and D_{JS} BIC significantly outperformed all other algorithms. A serious disadvantage of the H&L and Sarment algorithms was the need to predetermine the number of domains to be found. Often, the choice of a predetermination criterion is based on the number of domains that the authors favor (e.g., Haiminen and Mannila 2007). Predetermining the results is subjective and reduces the role of the segmentation algorithm to positioning the domain borders. A different form of domain predetermination is exercised in the Costantini et al. algorithm that best detected domains that matched its sliding window size.

In the simulations, we attempted to capture some of the statistical features of the genome that characterize its compositional homogeneity. Although we did not capture the whole compositional complexity of the genome, we believe that the framework established here is an important step in addressing the isochore question. Generating sequences with predetermined homogeneous domains gave us an advantage that cannot be gained by using real genomic data. These simulated data allowed us to study the sensitivity, precision rate, and computation time of each algorithm.

Our intent was not to demonstrate that one algorithm finds more isochores than another but rather to emphasize the importance of using a reliable method to study isochores and to demonstrate the differences of distinct method results.

It is important to note that every algorithm requires at least one fixed parameter as input. Using different parameter values often alters the segmentation results significantly. Because it is impossible to consider all parameter values for all algorithms, we used the parameter values recommended by the authors. It is therefore possible that, with different parameters, some algorithms will perform better than presented. In practice, however, one set of parameters would not maximize the algorithm performances for all sequence. For example, D_{JS}-based algorithms, which performed relatively well, failed to infer many domains with low between-domain variability or short domains because of the particular choice of the thresholds. Had the threshold been lower, the D_{JS} algorithm would have improved its performances with low between-domain variability and short domains, but its precision rate with long domains or domains with high between-domain variability would have declined sharply (results not shown). Deciding which parameters to use is a common difficulty to all algorithms and can only be resolved by a parameter-free algorithm. Such an algorithm is currently under development. In the meantime, we recommend using D_{JS} with its default parameters.

Our simulated domains have clear boundaries, and it is possible (but not probable) that with less clearly defined boundaries, other methods will perform better than those based on the Jensen–Shannon divergence. Our data clearly indicate that all the algorithms perform worse when the between-domain variability is reduced, as would be the case if boundaries were less clearly defined.

We hope that benchmarks will be useful in testing new segmentation algorithms and that poorly performing algorithms will not be used further without modification. We also hope that our framework is a first step in an attempt to explore the compositional structures of genomes.

Supplementary Material

Supplementary table S1 and figures S1–S4 are available at *Molecular Biology and Evolution* online (http://www.mbe.oxfordjournals.org/).

Acknowledgments

D.G. and E.E. were supported in part by National Science Foundation (NSF) grant DBI-0543342. K.J. was supported in part by a Texas Advanced Research Program/Advanced Technology Program grant and NSF grants DMS-0604429 and DMS-0817649. We thank Tal Dagan for kindly providing the Matlab scripts for the D_{JS} algorithm. We thank Laurent Guéguen for his help in using the Sarment algorithm.

References

Benjamini Y, Hochberg Y. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J R Stat Soc Ser B Met*. 57:289–300.

Bernaola-Galván P, Ivanov PC, Nunes Amaral LA, Stanley HE. 2001. Scale invariance in the nonstationarity of human heart rate. *Phys Rev Lett*. 87:168105.

Bernaola-Galván P, Román-Roldán R, Oliver JL. 1996. Compositional segmentation and long-range fractal correlations in DNA sequences. *Phys Rev E*. 53:5181–5189.

Bernardi G. 2000. Isochores and the evolutionary genomics of vertebrates. *Gene* 241:3–17.

Bernardi G. 2001. Misunderstandings about isochores. Part 1. *Gene* 276:3–13.

Bernardi G, Olofsson B, Filipski J, Zerial M, Salinas J, Cuny G, Meunier-Rotival M, Rodier F. 1985. The mosaic genome of warm-blooded vertebrates. *Science* 228:953–958.

Boys RJ, Henderson DA. 2004. A Bayesian approach to DNA sequence segmentation. *Biometrics* 60:573–581.

Braun JV, Müller HG. 1998. Statistical methods for DNA sequence segmentation. *Statist Sci*. 13:142–162.

Churchill GA. 1989. Stochastic models for heterogeneous DNA sequences. *Bull Math Biol*. 51:79–94.

Churchill GA. 1992. Hidden Markov chains and the analysis of genome structure. *Comput Chem*. 16:107–115.

Clauset A, Shalizi CR, Newman MEJ. 2009. Power-law distributions in empirical data. *SIAM Review*. 51:661–703.

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.

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

Cormen TH, Leiserson CE, Rivest RL. 1990. Introduction to algorithms. Cambridge (MA): MIT Press.

Costantini M, Clay O, Auletta F, Bernardi G. 2006. An isochore map of human chromosomes. *Genome Res*. 16:536–541.

Cuny G, Soriano P, Macaya G, Bernardi G. 1981. The major components of the mouse and human genomes: preparation, basic properties and compositional heterogeneity. *Eur J Biochem*. 115:227–233.

Eyre-Walker A, Hurst LD. 2001. The evolution of isochores. *Nat Rev Genet*. 2:549–555.

Fickett JW, Torney DC, Wolf DR. 1992. Base compositional structure of genomes. *Genomics* 13:1056–1064.

Fukagawa T, Sugaya K, Matsumoto K, Okumura K, Ando A, Inoko H, Ikemura T. 1995. A boundary of long-range G + C% mosaic domains in the human MHC locus: pseudoautosomal boundary-like sequence exists near the boundary. *Genomics* 25:184–191.

Guéguen L. 2005. Sarment: python modules for HMM analysis and partitioning of sequences. *Bioinformatics* 21:3427–3428.

Haiminen N, Mannila H. 2007. Discovering isochores by least-squares optimal segmentation. *Gene* 394:53–60.

Haiminen N, Mannila H, Terzi E. 2007. Comparing segmentations by applying randomization techniques. *BMC Bioinformatics* 8:171.

Häring D, Kypr J. 2001. Mosaic structure of the DNA molecules of the human chromosomes 21 and 22. *Mol Biol Rep*. 28:9–17.

Husmeier D, Wright F. 2002. A Bayesian approach to discriminate between alternative DNA sequence segmentations. *Bioinformatics* 18:226–234.

International Human Genome Sequencing Consortium. 2001. Initial sequencing and analysis of the human genome. *Nature* 409:860–921.

Li W. 2001a. Delineating relative homogeneous G+C domains in DNA sequences. *Gene* 276:57–72.

Li W. 2001b. New stopping criteria for segmenting DNA sequences. *Phys Rev Lett*. 86:5815–5818.
Li W. 2002. Are isochore sequences homogeneous? *Gene* 300:129–139.
Li W, Bernaola-Galván P, Carpena P, Oliver JL. 2003. Isochores merit the prefix 'iso'. *Comput Biol Chem*. 27:5–10.
Li W, Bernaola-Galván P, Haghighi F, Grosse I. 2002. Applications of recursive segmentation to the analysis of DNA sequences. *Comput Chem*. 26:491–510.
Lin J. 1991. Divergence measures based on the Shannon entropy. *IEEE Trans Inf Theory*. 37:145–151.
Macaya G, Thiery JP, Bernardi G. 1976. An approach to the organization of eukaryotic genomes at a macromolecular level. *J Mol Biol*. 108:237–254.
Nekrutenko A, Li WH. 2000. Assessment of compositional heterogeneity within and between eukaryotic genomes. *Genome Res*. 10:1986–1995.
Newman MEJ. 2005. Power laws, Pareto distributions and Zipf's law. *Contemp Phys*. 46:323–351.
Oliver JL, Carpena P, Hackenberg M, Bernaola-Galván P. 2004. IsoFinder: computational prediction of isochores in genome sequences. *Nucleic Acids Res*. 32:W287–W292.
Oliver JL, Román-Roldán R, Perez J, Bernaola-Galván P. 1999. SEGMENT: identifying compositional domains in DNA sequences. *Bioinformatics* 15:974–979.
Pavlicek A, Clay O, Jabbari K, Paces J, Bernardi G. 2002. Isochore conservation between MHC regions on human chromosome 6 and mouse chromosome 17. *FEBS Lett*. 511:175–177.
Schmidt T, Frishman D. 2008. Assignment of isochores for all completely sequenced vertebrate genomes using a consensus. *Genome Biol*. 9:R104.
Sokal RR, Rohlf FJ. 1995. Biometry. 3rd ed. New York: W.H. Freeman and Company.
Wen SY, Zhang CT. 2003. Identification of isochore boundaries in the human genome using the technique of wavelet multiresolution analysis. *Biochem Biophys Res Commun*. 311:215–222.
Zar JH. 1999. Biostatistical analysis. Upper Saddle River (NJ): Prentice-Hall.