Stat Biosci (2010) 2: 18–40
DOI 10.1007/s12561-010-9019-9

Non-parametric Decoding on Discrete Time Series and Its Applications in Bioinformatics

Hsieh Fushing · Shu-Chun Chen · Chii-Ruey Hwang

Received: 8 December 2009 / Accepted: 8 April 2010 / Published online: 28 April 2010

Abstract We address the question: How do we non-parametrically decode the unknown state-space vector underlying a lengthy discrete time series? The time series of concern is governed by one non-autonomous dynamics with only two internal states. This question pertinently reflects the dilemma of computing infeasibility against inferential bias found in many scientific areas. This dilemma becomes an issue when considering whether to have, or not to have likely very unrealistic structural assumptions on the state-space dynamics in most of real-world applications. To resolve this dilemma, the decoding problem is transformed into an event-intensity change-point problem without prior knowledge of the number of change-points involved. A new decoding algorithm, called Hierarchical Factor Segmentation (HFS), is proposed to achieve computability and robustness. Performance of the HFS algorithm in terms of total decoding error is compared to the decoding benchmark Viterbi algorithm through computer experiments. Under Hidden Markov Model (HMM) settings with true parameter values, our HFS algorithm is competitive against the Viterbi algorithm. Interestingly, when the Viterbi algorithm operates with maximum likelihood estimated (MLE) parameter values, our HFS algorithm performs significantly better. Similar favorable results are found when the Markov assumption is violated. We further demonstrate one very important application of our HFS algorithm in bioinformatics as a promising computational solution for finding CpG islands—DNA segments with aggregated CpG dinucleotides—on a genome sequence. A real illustration on a subsequence of human chromosome $^{\#}22$ is carried out and compared with one popular search algorithm.

H. Fushing (✉)
Department of Statistics, Mathematical Sciences Building, University of California at Davis, Davis, CA 95616, USA
e-mail: fushing@wald.ucdavis.edu

S.-C. Chen · C.-R. Hwang
Math. Inst. Academia Sinica, Taipei, Taiwan

Keywords Aggregating pattern · CpG island · Kullback distance · Intensity change-points analysis · Maximum entropy distribution · Schwarz' information criterion (BIC)

1 Introduction

Decoding is most often referred to in information theory as an algorithm for recovering a sequence of code words or messages, given a sequence of output signals in communication through a noise channel [12]. It is a task corresponding to attempts to understand non-autonomous dynamics in mathematics and physics [21]. In general, a non-autonomous dynamics is described via a system of difference, or differential, equations as:

$$\begin{aligned} \Delta X(t+1) &= G\big(X(t), S(t), \theta\big); \\ \Delta S(t+1) &= F\big(\bar{S}(t), \vartheta\big). \end{aligned} \tag{1}$$

Here changes in an observable response variable X are regulated by the unobservable internal state variable $S(t)$ through $G(\cdot : \theta)$, while $S(t)$, also called the driving force, is self-regulated by its own past history $\bar{S}(t)$ through $F(\cdot : \vartheta)$. Thus the information about the state-space dynamics represented by $F(\cdot : \vartheta)$ is the key to understanding the whole dynamics.

In statistics, especially in time series literature, the idea of state-space models can be traced back to a great extent to the non-autonomous dynamics given in (1) [6, 17, 30]. The functional forms of $G(\cdot : \theta)$ and $F(\cdot : \vartheta)$ are usually assumed known up to the unknown parameters, as in mathematics, but the time series of $\mathcal{S}_n = \{S(t) : t = 1, 2, \ldots, n\}$, generally called the state-space vector, is missing, as is often found in physics. However, work on state-space models commonly focuses on making inferences about the parameters θ and ϑ under structural assumptions on $G(\cdot)$ and $F(\cdot)$ based on the observed time series $\mathcal{X}_n = \{X(t) : t = 1, 2, \ldots, n\}$. Most often, if not always, the state-space vector $\mathcal{S}_n$ is not the primary interest. It has become routine that $\mathcal{S}_n$ is treated merely as missing data to be integrated out from the complete data likelihood function of θ and ϑ in the EM-algorithm.

Here the specific task of recovering the unknown $\mathcal{S}_n$ based on $\mathcal{X}_n$ is called decoding. When $G(\cdot : \theta)$ and $F(\cdot : \vartheta)$ are assumed known up to θ and ϑ, this task is called parametric decoding. Parametric decoding is popular in many real-world applications, especially related to the Hidden Markov Model (HMM) with finite states. For example, parametric decoding is used to compute the sequence of words given a sequence of acoustic signals in speech recognition [23]. It is also used to compute the sequence of "business cycles," or regime changes, given a time series of the unemployment rate in quantitative finance and econometrics [13]. It is used to compute a sequence of exons and introns given a genome sequence in computational genetics [5]. And as mentioned, it is used in computing a sequence of messages given an output signal sequence in information theory [12].

From a computational perspective in parametric decoding, when the state variable $S(t)$ has a finite number $|S|$ of states, decoding generally involves finding the most "probable" n-vector of $S(\cdot)$ among $|S|^n$ possible configurations. An exhaustive search for this most probable one is surely infeasible when n is large. Surprisingly, under certain parametric structural assumptions, the decoding computation can be reduced to a much more feasible level, such as one of order $O(n|S|^k)$ for a fixed number k. Usually this significant reduction in computation is achieved by a dynamic programming technique. Thus most popular decoding algorithms have a dynamic programming backbone. For example, the most widely used decoding technique is the Viterbi algorithm [29]. Its computation of the most likely configuration can be achieved within $O(n|S|^2)$ operations [12]. Now several of its variant algorithms are also popular for different purposes, such as the posterior-Viterbi algorithm.

To put the achievements in parametric decoding in perspective, in order to understand non-autonomous dynamics as a whole, it should be noted that these parametric decoding algorithms all intrinsically rest on the parametric structural assumptions. Therefore we can expect that these decoding results will be sensitive to any assumption violation. In other words, parametric decoding can be rather non-robust.

In this paper, decoding the unknown state-space vector $\mathcal{S}_n$, given no structural assumptions imposed on the functionals $G(\cdot)$ and $F(\cdot)$, is our primary interest. This task is termed "non-parametric decoding." Thus the chief merits of non-parametric decoding are freedom from the sensitivity of parametric assumptions, and avoidance of biased statistical inference when extracting information of $F(\cdot)$.

Our non-parametric decoding algorithm is called the Hierarchical Factor Segmentation (HFS) algorithm. It was originally developed by Fushing et al. [7] for compressing the recording of animal behavior. In addition to the study of animal behavior, this technique has been shown to successfully provide very promising results in the study of circadian rhythms, psychological and emotional coherence, and stock dynamics in several separate reports by the authors.

From the computational perspective under a non-parametric setting, there is no platform for dynamic programming techniques, neither is an exhaustive search possible. Since the state variable $S(t)$ only takes two values, ideally the non-parametric decoding can be thought of as a task in change-point analysis, but without prior knowledge as to how many change-points are involved. This is still a problem with computing infeasibility.

To overcome this difficulty, the theoretical aspect of the HFS algorithm is briefly depicted as follows. Instead of directly analyzing the measurement of $\mathcal{X}_n$, we transform this time series into a 0 or 1 record of a recurrent characteristic event. Our HFS algorithm is performed on this 0–1 time series to partition the whole time series into segments, such that between adjacent segments intensities of the chosen characteristic event are significantly different. Then a model selection problem in deduced, and the Schwarz information criterion (BIC) [27] is applied to select the "best" partitions. The computational aspect of the HFS algorithm, as will be see in Sect. 3, is indeed very practical.

Regarding the decoding performance of the HFS algorithm, a series of computer experiments are carried out, and the total number of decoding errors is used as the performance index. Surprisingly, our HFS algorithm with its threshold parameters selected via the Schwarz information criterion (BIC) achieves a very competitive performance when compared to the benchmark Viterbi algorithm with the true parameter values under the HMM setting. Very importantly, again under the same HMM settings, the HFS algorithm is demonstrated to exhibit a performance much superior to that of the Viterbi algorithm with maximum likelihood estimates (MLE) for the parameter values. Furthermore, we also show that the performance of the Viterbi algorithm is not as desirable as that of the HFS when the Markov structural assumption is misspecified.

We then apply the HFS algorithm to identify CpG islands on a human genome subsequence 1.4×10^5 base pairs long. We further demonstrate that our HFS algorithm is a promising computational solution for finding CpG islands, which is one very important issue in bioinformatics. The significance of finding CpG islands is that this genome segment, with a high content of CpG dinucleotides (i.e., a cytosine directly followed by a guanine), is found in most expressed genes (see [2, 3, 26]). Recently its epigenome and functional properties in normal and diseased cells or tissues have been under very intensive investigation in biological and biomedical research. However, so far in bioinformatics and computational genetics literature the identification of CpG islands is still carried out through more or less ad hoc algorithms, or by the Viterbi algorithm and its variants, where the Markov assumption is known to be invalid. Our HFS algorithm provides very efficient computations for identifying CpG islands based on its original epigenomic property. Indeed, this is the problem that motivated our study.

This paper is organized as follows. The setting of a non-autonomous dynamic system and computational measurements of evidence for an event aggregating pattern is discussed in Sect. 2. Our HFS algorithm is introduced in Sect. 3 and its optimal threshold values are derived in Sect. 4. In Sect. 5, we report the results of computer experiments under HMM and beyond HMM settings. Real data analysis of CpG islands is reported in Sect. 6. The final discussion section collects several decoding-related issues.

2 Heuristic Ideas, Event Aggregating Patterns and Maximum Entropy

In this section, we lay out the setting for decoding algorithms. To prepare the development of the HFS algorithm, we discuss some related heuristic ideas, how to perceive event aggregating patterns, and why the maximum entropy principle is needed.

Throughout this paper we work on a discrete time series $\mathcal{X}_n$ of length n, with $X(t)$ taking values from a finite set of symbols $\mathcal{A} = \{A_1, \ldots, A_K\}$, for $t = 1, 2, \ldots, n$. While the state variable $S(t)$ takes only 0 or 1 value, $X(t)$ is distributed according to a discrete probability $\mathcal{P}_{S(t)}$ defined on $\mathcal{A} = \{A_1, \ldots, A_K\}$. Here we suppose that for all symbols $A_k \in \mathcal{A}$, $0 < \frac{\mathcal{P}_0(A_k)}{\mathcal{P}_1(A_k)} < \infty$, so that no event can identify a state on $S(t)$ with certainty. As for the HMM setting, the Markov property of order one is used for simplicity. This corresponds to a very simple linear graphic structure sequentially linking $S(1)$ through $S(n)$. See Geman and Kochanek [12] for an example of a HMM represented by a more complex graph.

2.1 Heuristic Ideas

The idea behind our development in this paper goes heuristically as follows: first, we empirically choose an observable event that has the potential of separating $\mathcal{P}_0$ from

$\mathcal{P}_1$; second, we then find evidence whether there exists an aggregating pattern of such an event; third, the HFS algorithm is used to construct a segmentation on $\mathcal{X}_n$ as a quasi-structure on $\mathcal{S}_n$; fourth, by applying the maximum entropy principle to each segment of the segmentation, a likelihood of two intensity parameters is derived; and finally, the Schwarz information criterion is used to select the "best" segmentation of $\mathcal{X}_n$.

Therefore we do not construct a likelihood function for $\mathcal{P}_0$ and $\mathcal{P}_1$ and their transition probabilities as in the HMM setting. Once a decoded $\mathcal{S}_n$ is available from the HFS algorithm, statistical inference about $\mathcal{P}_0$ and $\mathcal{P}_1$ and the governing dynamics of $\mathcal{S}_n$ become possible. Discussion of these related issues is deferred to a separate study because they involve inferences that are rather distinct from the decoding issues, which are the focus here, and would certainly lengthen this paper considerably.

More specifically, a segmentation of $\mathcal{X}_n$ can be perceived as a spatial mapping for heterogeneity of recurrence times of the chosen event. In Sect. 2.2 we try to evaluate the evidence of whether this particular event indeed reveals uneven aggregating patterns on $\mathcal{X}_n$. In Sect. 2.3, we look at an aspect of evidence of aggregating patterns by comparing the empirical one to its maximum entropy background, which supposedly contains no aggregating patterns at all.

2.2 Null Recurrence Time Distribution via Maximum Entropy Principle

At first we consider the hypothetical setting that $\mathcal{X}_n$ is generated under one constant state-space trajectory. For a chosen event, its average recurrence time is empirically computable from $\mathcal{X}_n$. We take this average as the only available information to be employed as a constraint in derivation of the recurrence time distribution under the null hypothesis. Via the maximum entropy principle [15, 16], this null recurrence time distribution is found in form of geometric distribution with intensity parameter specified by the average recurrence time. Though this principle is not commonly applied outside of Bayesian statistics, it is popularly used in physics for constructing a distribution that rightly fits the information conveyed by the known constraints. This distribution ideally and conceptually should not contain any feature beyond what is specified by the constants. In the next subsection, this null recurrence time distribution would be compared with its empirical counterpart for detecting whether $\mathcal{X}_n$ is indeed generated under one constant state-space trajectory or not. If this null hypothesis is rejected, then we would take the collection of observed recurrence times as being generated from two different geometric distributions corresponding to the two different states.

Consider a generic event of interest A taken from a σ-field constructed based on $\mathcal{A}^k$ with small constant integer k ($\ll n$). With event A, the observed time series $\mathcal{X}_n$ is transformed into a 0–1 digital string, denoted by $\mathcal{C}^0 = (C^0(k), \dots, C^0(n-k+1))$, according to the following coding scheme: $C^0(t) = 1$ if event A is observed at time t on $\mathcal{X}_n$; otherwise, $C^0(t) = 0$. A schematic coding on a genome subsequence and an illustrative real example of $\mathcal{C}^0$ with $k = 2$ are given respectively in panels (a) and (b) of Fig. 1.

Further, we use the notations T_A and $c = \hat{E}[T_A] = \hat{E}[T_A|\mathcal{C}^0]$ to respectively denote the recurrence time random variable and the average recurrence time of code

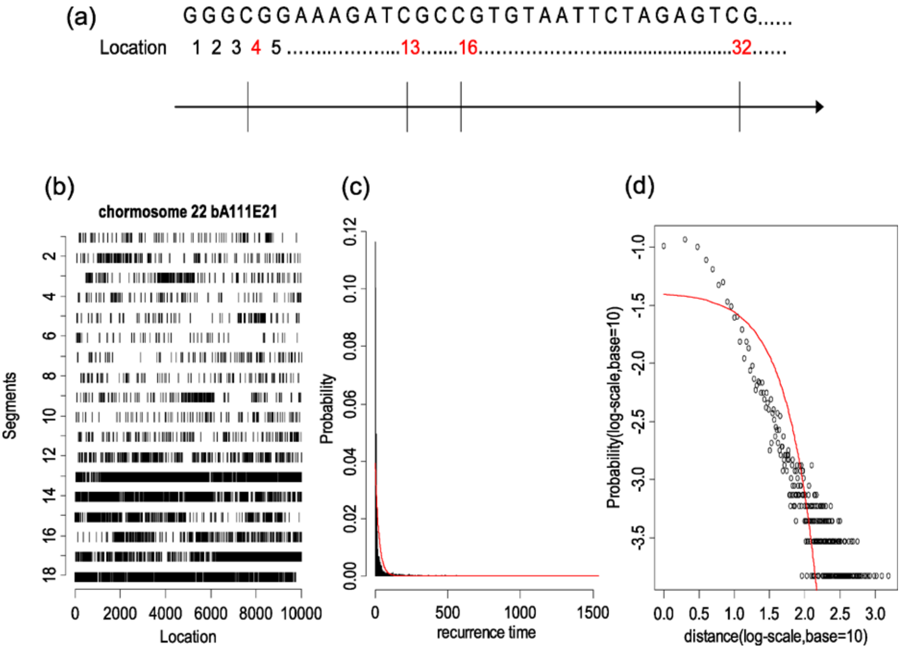


Fig. 1 Event and event-time series, recurrence time histogram and the maximum entropy distribution. (**a**) The CG (one C followed by G) as the chosen event, and a schematic transformation of genome subsequence into an event-time series; (**b**) the event-time series; (**c**) the histogram of recurrence time ($\mathcal{H}^*$) and its corresponding maximum entropy geometric distribution ($G^{\mathrm{ME}}(\hat{\lambda})$); (**d**) $\mathcal{H}^*$ vs. $G^{\mathrm{ME}}(\hat{\lambda})$ on log-probability and log-time scale

word 1 on $\mathcal{C}^0$, which is equivalent to the recurrence time random variable and its average of event A along $\mathcal{X}_n$. We take the observable average recurrence time $c = \hat{E}[T_A]$ as the only available piece of information regarding the distribution of T_A. We then derive such a distribution via maximum entropy principle as follows. Let the probability distribution of T_a be denoted by $p_i = \Pr[T_A = i]$, $i = 0, 1, \ldots$ and the infinity vector $\tilde{p} = (p_0, p_1, \ldots)$. Consider a target function subject to the two constraints: $E[T] = c = \sum_{i=1}^{\infty} i p_i$ and $1 = \sum_{i=0}^{\infty} p_i$:

$$Q(\tilde{p}, \lambda) = \sum_{i=0}^{\infty} p_i \log p_i + \eta_1 \left(\sum_{i=1}^{\infty} i p_i - c \right) + \eta_2 \left(\sum_{i=1}^{\infty} p_i - 1 \right).$$

By applying the Lagrange Multiplier approach, the maximum entropy distribution of T_A as a maximizer of $Q(\tilde{p}, \lambda)$ is specified by:

$$p_1 = 1/c;$$
$$p_i = 1/ce^{\lambda(i-1)};$$
$$\lambda = \log\left\{ \frac{c}{c-1} \right\}.$$

We denote this geometric distribution by $G^{\mathrm{ME}}(\lambda)$ with characteristic intensity $\lambda = \log\{\frac{c}{c-1}\}$. Then we take $G^{\mathrm{ME}}(\lambda)$ as the null distribution. We test the null hypothesis of constant state-space trajectory by comparing $G^{\mathrm{ME}}(\lambda)$ with the empirical one summarized from $\mathcal{C}^0$ in the next subsection. An illustrative comparison of the two distributions is given in panels (c) and (d) of Fig. 1.

2.3 Evidence of Event Aggregating Patterns

We construct the empirical recurrence time distribution, or histogram, of T_A from $\mathcal{C}^0$ and denote it as $\mathcal{H}^*$. In order to perceive whether $\mathcal{H}^*$ contains information about an aggregating pattern of event A, we compare $\mathcal{H}^*$ with the maximum entropy geometric distribution $G^{\mathrm{ME}}(\hat{\lambda})$ with $\hat{\lambda} = \log\{\frac{c}{c-1}\}$ and average recurrence time $E[\hat{T}_A] = c$. Heuristically, when there exist aggregating patterns along $\mathcal{C}^0$, $\mathcal{H}^*$ should manifest one vivid distinction from $G^{\mathrm{ME}}(\hat{\lambda})$: $\mathcal{H}^*$ has higher probabilities on relatively extremely small as well as relatively extremely large recurrence times than that of $G^{\mathrm{ME}}(\hat{\lambda})$, while the pattern is reversed on the middle region of recurrence time. This distinct feature is illustrated in Fig. 1 based on a genome subsequence from human chromosome #22. Visually, Fig. 1(b) shows a rather clear alternating pattern of aggregation and sparsity of CG dinucleotides. These visual patterns are reflected in the aforementioned distinction between $G^{\mathrm{ME}}(\hat{\lambda})$ and $\hat{R}_A$ as shown in panel (c), and even more vividly in panel (d), of the figure.

Though Fig. 1 reveals pictorial evidence of aggregating patterns, how to effectively summarize such evidence quantitatively may not be obvious. For instance, consider the Kullback distance between $G^{\mathrm{ME}}(\hat{\lambda})$ and $\mathcal{H}^*$ defined as

$$\mathcal{K}\big(\mathcal{H}^*, G^{\mathrm{ME}}(\hat{\lambda})\big) = \sum_{i=1}^{\infty} \mathcal{H}^*(i) \log \frac{\mathcal{H}^*(i)}{G^{\mathrm{ME}}(i:\hat{\lambda})}.$$

From Fig. 1(d), the log-probability ratios $\log \frac{\mathcal{H}^*(i)}{G^{\mathrm{ME}}(i:\hat{\lambda})}$ are positive for the two tail regions, but negative on the middle region. Although the three regions are all coherent with the aggregating pattern, the positive and negative cancelation would make the Kullback distance a less effective summarizing statistic for evaluating the characteristic feature. In order to resolve this shortcoming of Kullback distance, one must determine how to mark the three regions first before evaluating a proposed summarizing statistic. This task could involve smoothing techniques and possibly some ad hoc choices of thresholds. Hence it is not at all a simple task that can be done without potential disagreement.

Given the foundation that the recurrence time of CG dinucleotides is not likely being generated by one single geometric distribution, there must exist several state-switching embeddings underlying the segment of genome sequence. Under the binary state-space assumption, this fact implies the existence of aggregating and sparse patterns of event of interest. In the next section, from the segmentation perspective, instead of comparing $G^{\mathrm{ME}}(\hat{\lambda})$ and $\mathcal{H}^*$, we propose a more effective methodology for evaluating the event aggregation vs. sparsity patterns on $\mathcal{X}_n$. Thus we are free from the above difficulty pertaining to Kullback distance.

3 Hierarchical Factor Segmentation (HFS) Algorithm

The Hierarchical Factor Segmentation (HFS) algorithm is constructed based on the event-recurrent time distribution and multi-level coding schemes in this section. Functionally it is designed to find where the aggregating patterns begin and end along the temporal span of the time series of interest. Consequently the time series is partitioned into alternating sequences of high and low event-intensity segments.

Hierarchical factor segmentation (HFS) algorithm: With a chosen observable event A on the time series $\mathcal{X}_n$, the construction procedure of the HFS algorithm is depicted via the following steps:

HFS-1. By the event definition, $\mathcal{X}_n$ is transformed into 0–1 digital string of base 2 via the following coding scheme: code 1 for $X(t)$ when the event is observed at time t, otherwise code 0 for $X(t)$. This first-level 0–1 digital string is denoted by the code sequence $\mathcal{C}^0$.

HFS-2. Construct a histogram, say $\mathcal{H}^*$, of the event-recurrent time distribution from $\mathcal{C}^0$, and denote the sequence of recurrence time (inter-event spacing) by $\mathcal{R}^*$.

HFS-3. Choose the first threshold value as an upper p_* percentile on $\mathcal{H}^*$, say Cs, to transform the $\mathcal{R}^*$ sequence into a 0^*–1^* digital string via the second-level coding scheme: (1) a recurrent time less than Cs is coded by a 0^*; (2) a recurrent time greater than Cs is coded by a 1^*. The resultant digital string of base 2 is denoted by the code sequence $\mathcal{C}^*$.

HFS-4. Upon code sequence $\mathcal{C}^*$, we take code word 1^* as another "new" event and construct its corresponding recurrent time histogram $\mathcal{H}^@$ and the sequence of inter-1^*-event spacing as $\mathcal{R}^@$.

HFS-5. Choose the second threshold value as an upper $p_@$ percentile from $\mathcal{H}^@$, say Cs^*, to transform $\mathcal{R}^@$ into another $0^@$–$1^@$ digital string via the top level coding scheme: (1) an inter-1^*-event spacing less than Cs^* is coded by $0^@$; (2) an inter-1^*-event spacing greater than Cs^* is coded by a $1^@$. The third digital string of base 2 is denoted by $\mathcal{C}^@$.

HFS-6. The resultant code sequence $\mathcal{C}^@$ is mapped back onto the time series $\mathcal{C}^0$ or $\mathcal{X}_n$ as a partition of $|\mathcal{R}^@|(=m)$ segments on the time span $[1, n]$. Denote this partition as $\mathcal{N}(\mathcal{X}_n) = \{[N_{Li}, N_{Ri})\}_{i=1}^m$.

This partition or segmentation $\mathcal{N}(\mathcal{X}_n)$ will be seen to achieve an aggregating pattern by separating the high-event-intensity segments against the low-event-intensity ones. A segment, say $[N_{Li}, N_{Ri})$, corresponding to $1^@$-code, is a period of time points falling in-between two widely separated 1^*-codes. The wide separation of two successive 1^*-codes implies that there are many 1-codes on the particular segment of code sequence $\mathcal{C}^0$, or equivalently, many events are observed on the segment of $\mathcal{X}_n$. Thus this is a segment of high event-intensity. In contrast, a segment, say $[N_{Li'}, N_{Ri'})$, corresponding to $0^@$-code (including segments corresponding to the two 1^*-codes on both of its ends) would present an extended period of having many 0-codes, but very sparse 1-codes. Thus it is a segment of low event-intensity.

The computational essence of the HFS algorithm is that, under no parametric assumptions, the number of partitions $\mathcal{N}(\mathcal{X}_n)$ pertaining to all the possible integer threshold values (Cs, Cs^*) with regard to a chosen event is much less than 2^n. It is this small set of candidate partitions on $\mathcal{X}_n$ that will facilitate the computing feasibility for model selection techniques, as discussed in the next section.

4 Intensity Change-point Analysis and Schwarz Information Criterion (BIC)

After application of the HFS algorithm with threshold parameters (Cs, Cs^*), $\mathcal{X}_n$ is partitioned into two classes of segments corresponding to code words $0^@$ and $1^@$ defining the segmentation $\mathcal{N}(\mathcal{X}_n)$. Furthermore, we consider that the recurrence time of event A within all segments corresponding to code word $0^@$ is stationary and geometrically distributed as $\mathrm{G}(\cdot : \lambda_0)$. In contrast, the recurrence time within all $1^@$ segments is stationary and $\mathrm{G}(\cdot : \lambda_1)$ distributed. Therefore any transition between $0^@$ and $1^@$ segments is taken as an intensity change-point. Denote the number of change-points by $H^0(\tilde{\tau})$ with $\tilde{\tau} = (Cs, Cs^*)$ for notational simplicity.

Let $\{k_{0j}\}_{j=1}^{m_0}$ and $\{k_{1j'}\}_{j'=1}^{m_1}$ denote the observed recurrent time in the $0^@$ and $1^@$ segments respectively. Next we compute the likelihood function of (λ_0, λ_1) based on the observed recurrence times within the two classes of segments, assuming they are independent copies, as

$$L(\lambda_0, \lambda_1) = P_0^{m_0}(\lambda_0) \prod e^{-\lambda_0(k_{0j}-1)} P_0^{m_1}(\lambda_1) \prod e^{-\lambda_1(k_{1j'}-1)} \tag{2}$$

where $P_0(\lambda) = (1 - e^{-\lambda})$, and m_0 and m_1 are the total numbers of recurrence times on $0^@$ segments and $1^@$ segments, respectively.

The maximum likelihood estimates of (λ_0, λ_1) are derived as follows:

$$\hat{\lambda}_0 = \log\left\{\frac{\bar{c}_0}{\bar{c}_0 - 1}\right\},$$

$$\hat{\lambda}_1 = \log\left\{\frac{\bar{c}_1}{\bar{c}_1 - 1}\right\},$$

where $\bar{c}_0 = \frac{\sum k_{0j}}{m_0}$ and $\bar{c}_1 = \frac{\sum k_{1j'}}{m_1}$ are the average recurrence times on $0^@$ segments and $1^@$ segments, respectively.

As in the null case when no segmentation is imposed on $\mathcal{X}_n$ with the maximum entropy applicable on its whole time span, the event-recurrent time distribution is again geometrically distributed. Indeed, its maximum likelihood estimate coincides with the maximum entropy distribution. This is due to the fact that the number of events on $\mathcal{X}_n$ is a fixed constant. Denote this geometric distribution $\mathrm{G}(\cdot : \hat{\lambda})$ with $\hat{\lambda} = \log \frac{1+\bar{c}}{\bar{c}}$, $\bar{c} = n/m$ and $m = m_0 + m_1$. The maximum likelihood affiliated with $\mathrm{G}(\cdot : \hat{\lambda})$ is calculated as $L(\hat{\lambda}|\sigma) = P_0^m(\hat{\lambda})e^{-\hat{\lambda}(n-m)}$.

Thus the log-likelihood ratio is computed as

$$\begin{aligned}\Delta^0(\tilde{\tau}) &= \log L(\lambda_0, \lambda_1|\sigma) - \log L(\hat{\lambda}|\sigma) \\ &= \sum_{k=0}^{1}\left[m_k \log\left\{\frac{1}{\bar{c}_k - 1}\right\} - l_k \log\left\{\frac{\bar{c}_k}{\bar{c}_k - 1}\right\}\right] \\ &\quad - \left[m \log\left\{\frac{1}{\bar{c} - 1}\right\} - n \log\left\{\frac{\bar{c}}{\bar{c} - 1}\right\}\right]\end{aligned}$$

where $l_0 + l_1 = n$, and $l_0 = \sum k_{0j}$ and $l_1 = \sum k_{1j'}$ are the total lengths of the $0^@$ and $1^@$ segments, respectively.

It is clear that $\Delta^0(\tilde{\tau})$ can be a reasonable and effective candidate for a test statistic of the hypothesis of whether there exist aggregating patterns within $\mathcal{X}_n$, as briefly discussed in the previous section. It is worth emphasizing again here that the number of change-points of event-intensity is unknown. Hence the lack of this prior information raises an important question: Is $\Delta^0(\tilde{\tau})$ approximately chi-squared distributed, and, if so, with how many degrees of freedom under the null and the alternative hypotheses? This question and the related regime change issues will be addressed in detail in a separate report. Here we focus on the decoding issue.

To make use of $\Delta^0(\tilde{\tau})$ for selecting an optimal threshold parameter $\tilde{\tau}$ in the HFS algorithm, we apply the Schwarz' information criterion, also called the BIC criterion. That is, we perform the following optimization:

$$\begin{aligned}\Gamma(\tilde{\tau}) &= \Delta^0(\tilde{\tau}) - \frac{\log m}{2}\big(H^0(\tilde{\tau}) + 1\big), \\ \tilde{\tau}^{\text{opt}} &= \arg\max_{\tilde{\tau}}\big[\Gamma(\tilde{\tau})\big].\end{aligned}$$

From here on, the HFS algorithm is meant to be performed according to the choice of optimal threshold parameter $\tilde{\tau}^{\text{opt}}$.

Another closely related alternative model selection procedure is based on the minimum description length (MDL) of Rissanen [24, 25] (see also [19] and [20], for review and tutorial). The MDL criterion applied in our setting is the following:

$$\begin{aligned}\Gamma^*(\tilde{\tau}) &= \Delta^0(\tilde{\tau}) - \log n\big(H^0(\tilde{\tau}) + 1\big) - \frac{1}{2}[\log l_0 + \log l_1], \\ \tilde{\tau}^{\text{opt}*} &= \arg\max_{\tilde{\tau}}\big[\Gamma^*(\tilde{\tau})\big].\end{aligned}$$

Based on experience from our computer experiments, the penalty for increasing the number of change-points is too high. In fact, it might not be surprising, since this criterion was derived when the order of the set of candidate partitions is 2^n. In sharp contrast, our set of candidate partitions via HFS is indeed many orders smaller.

5 Computer Experiments Under Hidden Markov Models and Beyond

In this section we conduct computer experiments to numerically evaluate the decoding performance of our HFS algorithm with the BIC-selected threshold parameter

$\tilde{\tau}^{\mathrm{opt}}$. The performance index is measured through the average decoding error rate. In the experiments, we also compute fully parametric decoding procedures via the Viterbi and posterior-Viterbi algorithms under Hidden Markov Models (HMM) and non-HMM models with the parameter values being either completely known or estimated via the maximum likelihood estimation (MLE) approach. Their performances are meant to serve as benchmarks for the decoding efficiency of the HFS algorithm.

5.1 HMM Setting

Here we make use of the Hidden Markov Model setting with two dice: one is fair; the other is loaded, an example considered in [5, Chap. 3]. The alphabet is $\mathcal{A} = \{1, 2, \dots, 6\}$, and the state $\mathcal{P}_0(i) = 1/6$ for the fair die is designated as the state $S(t) = 0$, while the state $\mathcal{P}_1(i) = 1/10$ for $i = 1, 2, \dots, 5$, and $\mathcal{P}_1(6) = 1/2$ for the loaded die are designated as the state $S(t) = 1$. The Markov dynamics governed by the transition probability matrices of the two states 0 and 1 is specified by $p_{01} = \Pr[S(t+1) = 1 \mid S(t) = 0]$ and $p_{10} = \Pr[S(t+1) = 0 \mid S(t) = 1]$. Three cases of (p_{01}, p_{10}) are considered: (I) $(p_{01}^{(1)}, p_{10}^{(1)}) = (0.05, 0.1)$; (II) $(p_{01}^{(2)}, p_{10}^{(2)}) = (0.1, 0.2)$; (III) $(p_{01}^{(3)}, p_{10}^{(3)}) = (0.4, 0.5)$.

The MLEs for all involved parameters are computed via the Baum–Welch algorithm [1], which is equivalent to the EM algorithm [4]. The Viterbi algorithm, available in R-code, computes the most likely configuration as:

$$\mathcal{S}_n^V = \arg\max \Pr[\mathcal{S}_n \mid \mathcal{X}_n, \mathcal{P}_1, \mathcal{P}_0, p_{10}, p_{01}]$$

with either the true parameter values of $\mathcal{P}_1, \mathcal{P}_0, p_{10}, p_{01}$ or their MLE estimates.

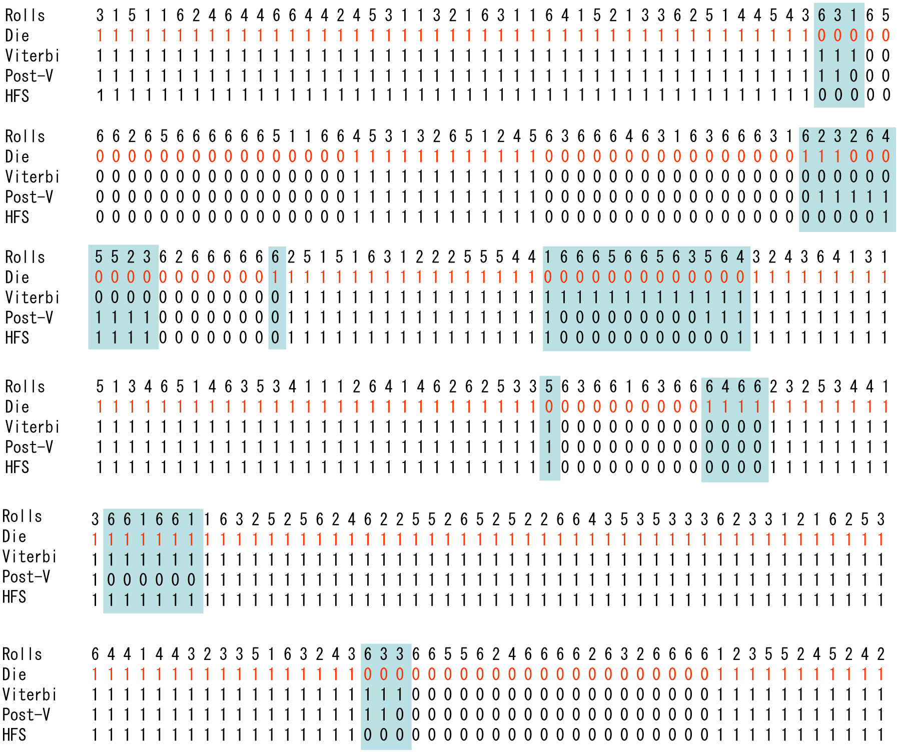
In contrast, the posterior-Viterbi algorithm computes the most likely $S(t)$ for each t individually as:

$$S^{\mathrm{PV}}(t) = \arg\max \Pr\big[S(t) \bigm| \mathcal{X}_n, \mathcal{P}_1, \mathcal{P}_0, p_{10}, p_{01}\big], \quad t = 1, \dots, n,$$

also with either the true parameter values of $\mathcal{P}_1, \mathcal{P}_0, p_{10}, p_{01}$ or their MLE estimates.

In case I the state-space vector $\mathcal{S}_n$ is expected to have $11\dots11$ segments (each of expected length 10) being well-separated by $000\dots00$ segments (each of expected length 20). This kind of aggregating pattern in $\mathcal{S}_n$ is slightly blurred in case II due to larger values of (p_{01}, p_{10}) and shorter expected lengths of repeating states. In case III, it is very difficult to decode $\mathcal{S}_n$, since the aggregating pattern almost disappears into a fashion very close to that of a random fair coin toss.

As an illustration of our computational decoding results with the side-6 as the chosen event, the Occasionally Dishonest Casino example of Chap. 3 in [5] is reanalyzed and reported in Fig. 2. Different choices of event could be used, but the results are expected to be less efficient in distinguishing the two dices. Based on the same data, the concave shape of Γ functions in the BIC procedure over a region of threshold values is reported in Fig. 3. It is very interestingly demonstrated as well that the modes of the concave curves are shown to correspond to very low error rates, which is comparable to the benchmarks set by the Viterbi and posterior-Viterbi algorithms with true parameter values. This corresponding relationship indicates that the HFS algorithm with BIC-selected threshold parameters captures the aggregating



```
Rolls    3 1 5 1 1 6 2 4 6 4 4 6 6 4 4 2 4 5 3 1 1 3 2 1 6 3 1 1 6 4 1 5 2 1 3 3 6 2 5 1 4 4 5 4 3 6 3 1 6 5
Die      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
Viterbi  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
Post-V   1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0
HFS      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0

Rolls    6 6 2 6 5 6 6 6 6 6 6 5 1 1 6 6 4 5 3 1 3 2 6 5 1 2 4 5 6 3 6 6 6 4 6 3 1 6 3 6 6 6 3 1 6 2 3 2 6 4
Die      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0
Viterbi  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Post-V   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1
HFS      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

Rolls    5 5 2 3 6 2 6 6 6 6 6 6 2 5 1 5 1 6 3 1 2 2 2 5 5 5 4 4 1 6 6 6 5 6 6 5 6 3 5 6 4 3 2 4 3 6 4 1 3 1
Die      0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1
Viterbi  0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Post-V   1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
HFS      1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1

Rolls    5 1 3 4 6 5 1 4 6 3 5 3 4 1 1 1 2 6 4 1 4 6 2 6 2 5 3 3 5 6 3 6 6 1 6 3 6 6 6 4 6 6 2 3 2 5 3 4 4 1
Die      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
Viterbi  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
Post-V   1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
HFS      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1

Rolls    3 6 6 1 6 6 1 1 6 3 2 5 2 5 6 2 4 6 2 2 5 5 2 6 5 2 5 2 2 6 6 4 3 5 3 5 3 3 3 6 2 3 3 1 2 1 6 2 5 3
Die      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Viterbi  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Post-V   1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
HFS      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Rolls    6 4 4 1 4 4 3 2 3 3 5 1 6 3 2 4 3 6 3 3 6 6 5 5 6 2 4 6 6 6 6 2 6 3 2 6 6 6 6 1 2 3 5 5 2 4 5 2 4 2
Die      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
Viterbi  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
Post-V   1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
HFS      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
```

Fig. 2 Decoding on Occasionally Dishonest Casino example via HFS (16 errors), Viterbi (28 errors) and posterior-Viterbi (28 errors) algorithms. Discrepancies in decoding are marked and total decoding errors are reported in parentheses

pattern on $\mathcal{S}_n$ fairly well in this data set. In fact, this case turns out to be rather typical under the case I setting, as indicated by results from a simulation study reported in Fig. 4.

Figure 4 presents the performance of the HFS algorithm with the optimal threshold parameter $\tilde{\tau}^{\text{opt}}$ in comparison to the Viterbi and posterior-Viterbi algorithms with both true and MLE-estimated parameter values on 100 simulated time series data in case I. In panel (a), the HFS algorithm outperforms the Viterbi and posterior-Viterbi algorithms with true parameter values in around 30% of the simulated time series. In contrast, it is better in more than 60% of the replications when the two algorithms are carried out with MLE parameter values. The correlation between the error rate and the Kullback distance between the maximum entropy geometric distribution and the empirical recurrence time distribution are rather significant among the 100 replications of the Viterbi and posterior-Viterbi algorithms with MLE-estimated parameter values, but less so among those with true parameter values. This implies that the Viterbi and posterior-Viterbi algorithms are not robust with MLE-estimated parameter values, but also that this problem will become worse when the decoding task gets more difficult.

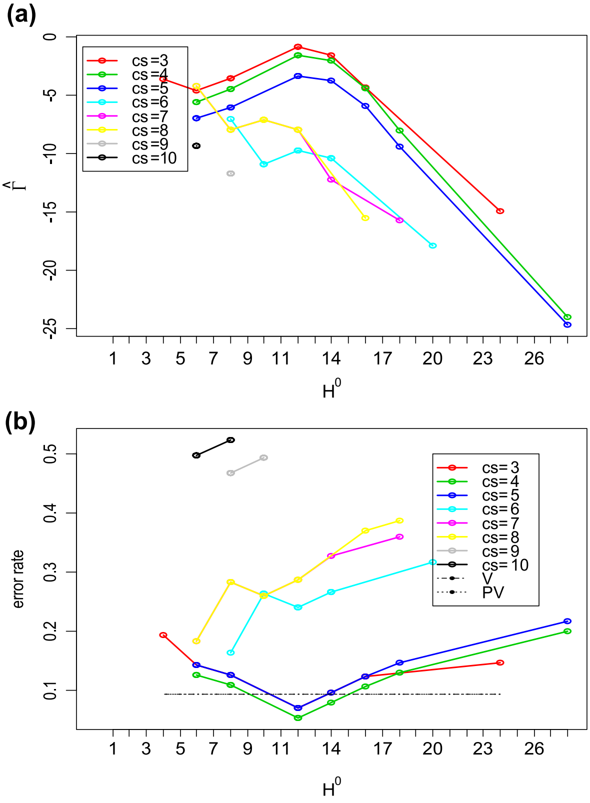


Fig. 3 BIC criterion and corresponding error rates based on Occasionally Dishonest Casino example. (**a**) The Γ functional curves pertaining to various threshold values of $\mathcal{H}^*$ with total number of change-points H_0 as the X-axis; (**b**) Total decoding error rate corresponding to each curve

In panel (b) via (smoothed) density of error rates, overall the Viterbi and posterior-Viterbi algorithms with true parameter values perform better than the HFS algorithm. But this overall pattern is reversed when the two algorithms are implemented with MLE-estimated parameter values. In panel (c), the performance comparisons are summarized in terms of operation characteristic (ROC) curves having the error rate histogram of the HFS algorithm as the baseline [14]. Here we use the following version of ROC curve for comparing a distribution function $G(\cdot)$ with a baseline distribution $F(\cdot)$: $G(F^{-1}(t))$, $0 \le t \le 1$, with $F^{-1}(t)$ the quantile function of $F()$. If $G(\cdot) \equiv F(\cdot)$, then the ROC curve is the diagonal segment of the unite square. If $G(\cdot)$ is located on the left of $F(\cdot)$, or equivalently the ransom variable corresponding to $G(\cdot)$ is stochastically smaller than the one corresponding to $F(\cdot)$, then the ROC curve would be above the diagonal. The farther away from the diagonal the ROC curve is, the stronger evidence of $G(\cdot)$ being on left of $F(\cdot)$ is. On the other hand, if the ROC curve is below the diagonal, then $G(\cdot)$ is located on the right of $F(\cdot)$. That is, the relationship of stochastic ordering of the two corresponding random variables is reverse. Therefore these ROC curves in panel (c) reconfirm the pattern in the sense that the densities of error rates of the Viterbi and posterior-Viterbi algorithms with true parameter values are located on the left-hand side of the HFSs, while the two densities of the Viterbi and posterior-Viterbi algorithms with MLE-estimated parameter values are located on the right-hand side of the HFSs.

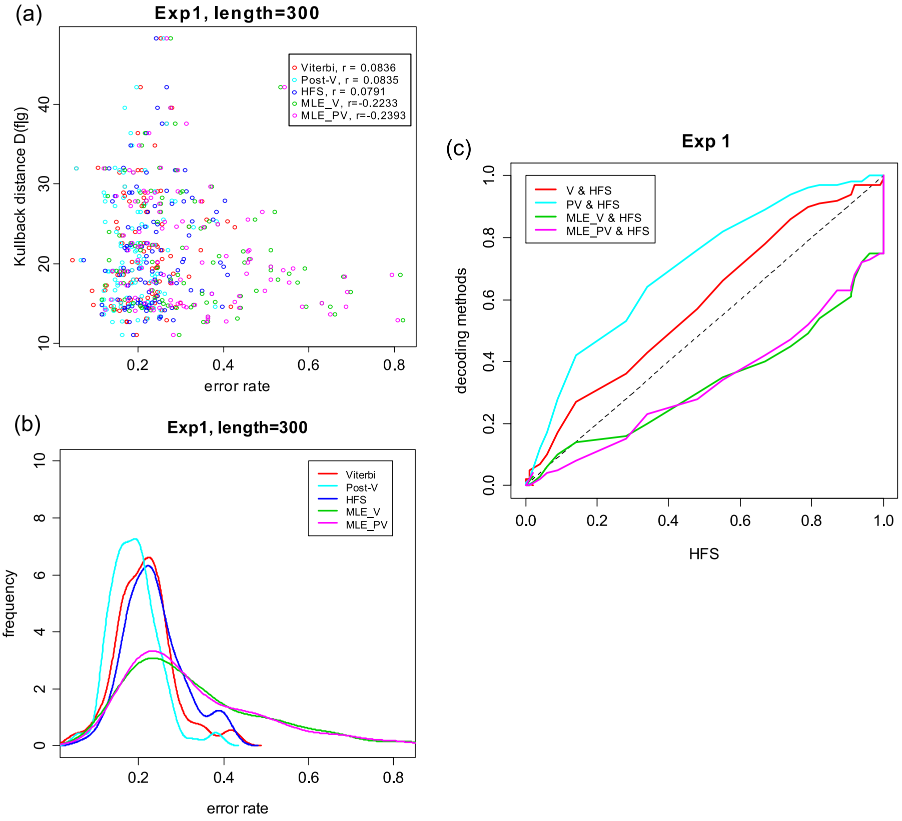


Fig. 4 HMM case I: total error rate for HFS vs. Viterbi and posterior-Viterbi (with true and MLE-estimated parameter values) algorithms. (**a**) Scatter plot of error rates w.r.t. Kullback distances; (**b**) densities of total decoding errors of the five decoding algorithms; (**c**) ROC curves with HFS histogram as the baseline

The decoding task in case II becomes more difficult than that in case I as shown in Fig. 5. In panel (a), the HFS algorithm only outperforms the Viterbi and posterior-Viterbi algorithms with true parameter values in around 20% of the simulated time series, while it remains at more than 60% when the two algorithms are carried out with MLE parameter values. Though panels (b) and (c) show decoding performance results similar to those in Fig. 4, panel (c) especially reveals a very important phenomenon: that the gap in performance between the Viterbi algorithm with true and MLE parameter values becomes larger; likewise for the posterior-Viterbi algorithm.

In case III with the dynamics of $\mathcal{S}_n$ being nearly equivalent to that of fair coin tossing, as shown in Fig. 6, the HFS algorithm is not comparable in decoding performance to both the Viterbi and posterior-Viterbi algorithms with true parameter values. This is mainly due to the fact that there is almost no aggregating pattern to be found on the simulated time series. However, the HFS algorithm surprisingly still performs better than the Viterbi and posterior-Viterbi algorithms with MLE-estimated parameter values. In panel (c), once again the performance gap of the Viterbi, or posterior-Viterbi,

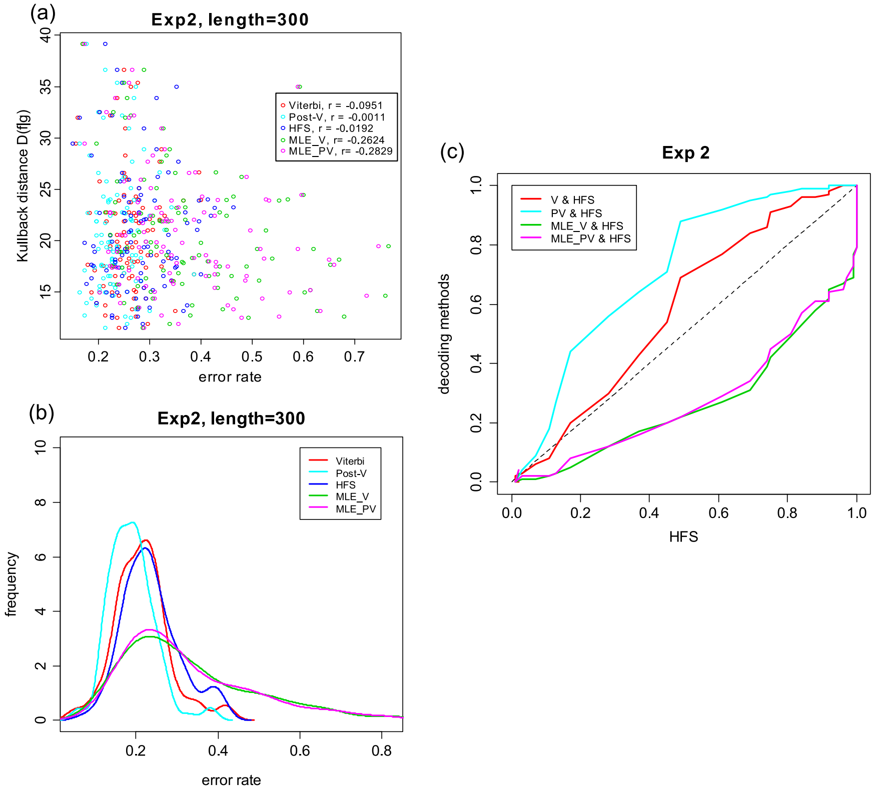


Fig. 5 HMM case II: total error rate for HFS vs. Viterbi and posterior-Viterbi (with true and MLE-estimated parameter values) algorithms. (**a**) Scatter plot of error rates w.r.t. Kullback distances; (**b**) densities of total decoding errors of the five decoding algorithms; (**c**) ROC curves with HFS histogram as the baseline

algorithm with both true and MLE parameter values relative to the HFSs is enlarged even more.

By summarizing the results reported in Figs. 3 through 6, we can conclude with confidence that the HFS algorithm is an effective and robust non-parametric decoding methodology under HMM setting. This computer experiment also pertinently points out that the robustness issues of the two benchmark decoding algorithms, Viterbi and posterior-Viterbi, need more research attention, and have to be used with caution.

5.2 Beyond HMM Setting

In another computer experiment, we consider a setting whose $\mathcal{S}_n$ dynamics has a regular pattern of being in 0 and 1 states for equal and fixed lengths of duration. This regularity violates the Markov property. By keeping P_0 and P_1 as in case I in the previous subsection, time series of length 300 is generated with $\mathcal{S}_n$ alternating its states every 10, 15 or 20 units of time. The Viterbi algorithm is carried out by using MLE estimations for the parameters of P_0 and P_1 and (p_{01}, p_{10}). From the

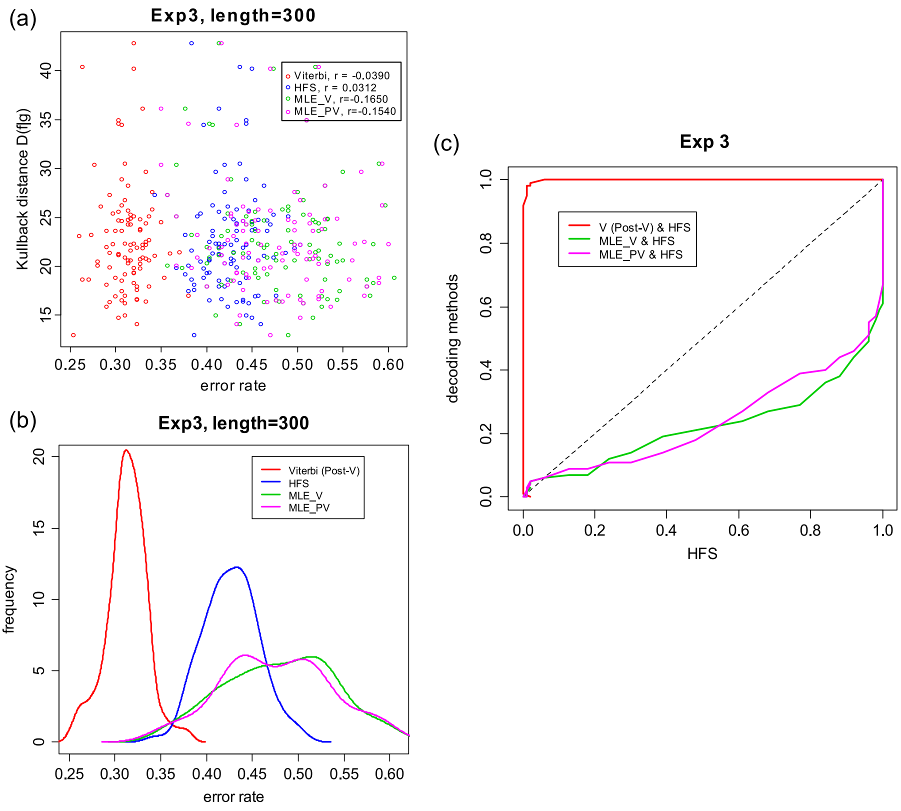


Fig. 6 HMM case III: total error rate for HFS vs. Viterbi and posterior-Viterbi (with true and MLE-estimated parameter values) algorithms. (**a**) Scatter plot of error rates w.r.t. Kullback distances; (**b**) densities of total decoding errors of the five decoding algorithms; (**c**) ROC curves with HFS histogram as the baseline

experimental results reported in Fig. 7, we can see that the Viterbi algorithm is also sensitive to the violation of the Markov assumption.

Finally, we emphasize again here that the demonstrated non-robustness of the Viterbi and posterior-Viterbi algorithms with respect to MLE estimates and the Markov assumption deserve proper caution in their real-world applications. Our HFS algorithm is surely more than an alternative to these two benchmarks for decoding purposes.

At the end of this section it is worth making further remarks regarding the comparison of HFS and Viterbi algorithms beyond the setting of binary latent state that we have discussed so far. In a case of the time series involving only one state, without state-change, both algorithms could overfit the state trajectory. Nevertheless, HFS algorithm is equipped with a safety check via geometric mixture analysis. As in cases with more the two latent states, Viterbi is mathematically more convenient than HFS algorithm, since multiple applications of HFS algorithm on the same time series are needed for marking multiple states. However, the non-parametric nature of HFS algorithm might always deserve consideration in most of real-world applications.

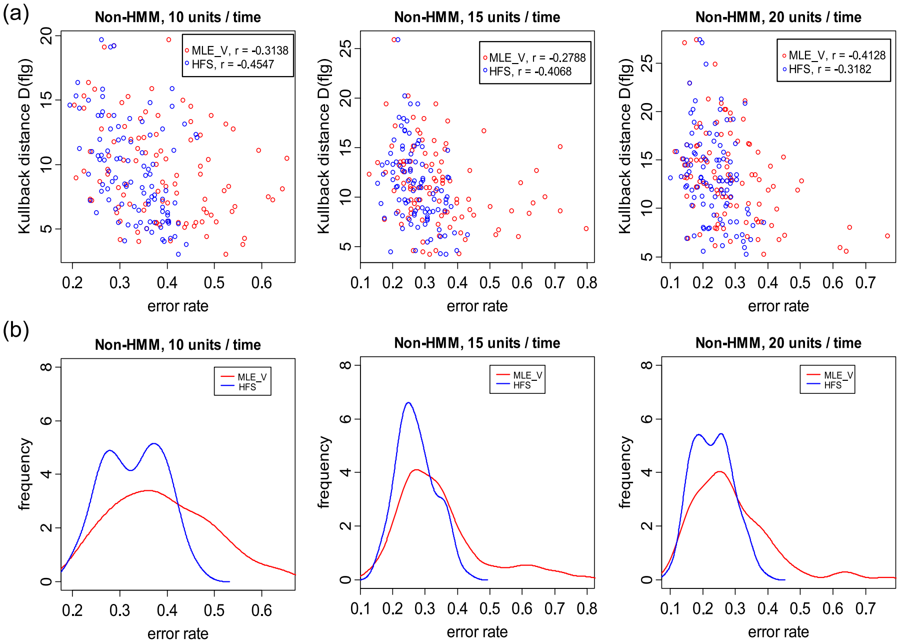


Fig. 7 Non-HMM case: total error rate for HFS vs. Viterbi (with MLE-estimated parameter values) algorithms. (**a**) Scatter plot of error rates w.r.t. Kullback distances; (**b**) densities of total decoding errors of the two decoding algorithms

6 CpG-island Data Analysis

The CpG dinucleotides (i.e., a cytosine directly followed by a guanine) were found to be relatively rare in vertebrates [2, 11]. It occurs that approximately three quarters (≈75%) as often as would be expected via the product of G (guanine) and C (cytosine) proportions in DNA of whole human genomes under an independence assumption. The striking feature is that CpG is very unevenly distributed throughout all 23 human chromosomes. Clusters of CpG—so-called CpG islands—are found to be associated with house-keeping genes and are typically near the transcription start sites. It is also highly associated with promoters, especially with coding exons, but not with non-coding exons or intons [26]. One explanation put forth is that DNA methylation in the human genome is largely confined to CpG dinucleotides, and methylated CpGs restrict transcription, while unmethylated CpGs in the vicinity of a gene allow the gene to be expressed. This gene expression property has led to intensive investigation and development for linking CpG islands with many disease states [18]. Another theory as to why many expressed genes have a very high CpG content is through evolution via natural selection.

While the CpG islands and their association are very actively studied in genetics, their identification is typically based on the algorithm proposed in Gardiner-Garden and Frommer [11] and its variant [28]. For convenience, the following criteria are listed below (see also the website http://genome.ucsc.edu/cgi-bin/hgTrack):

CpG filtering criteria: CpG islands were predicted by searching the sequence one base at a time, scoring each dinucleotide ($+17$ for CG and -1 for others) and identifying maximal scoring segments. Each segment was then evaluated for the following criteria:

- GC content of 50% or higher
- length greater than 200 bp
- ratio greater than 0.6 of observed number of CpG dinucleotides to the expected number on the basis of the number of Gs and Cs in the segment.

This set of criteria has been widely used in many CpG island-finder algorithms in genetics, many of them available on the Web (http://www.ebi.ac.uk/emboss/cpgplot).

On the one hand, the above set of criteria does not embrace epigenetic and functional properties, with which CpG islands were originally identified [3]. On the other hand, this set of criteria is not mathematically complete so as to be taken as the CpG island "definition" on a genome sequence. This set of threshold-based criteria is somehow ad hoc, because no objective standards are available for defining CpG islands. Therefore, from a biological perspective, any research attempt to analyze CpG islands related to properties of promoters encounters the dual difficulty of defining what constitutes a CpG island and what constitutes a CpG island-promoter association [26].

Very importantly, from a computational perspective, this incompleteness would render that finding all the CpG islands on a long enough genome is nearly infeasible when no mathematical structural assumptions are imposed on the CpG dinucleotides occurrence mechanism. Hence we can expect that many ad hoc CpG finder algorithms run into the danger of missing many segments of a genome satisfying the above set of criteria. We illustrate this in the following real example.

Here we apply our HFS algorithm to the bA111E21 subsequence of human chromosome 22 with computed optimal threshold parameter $\tilde{\tau}^{\text{opt}} = (48, 6)$. The genome information of the genome subsequence bA111E21 is summarized in Fig. 8, while Fig. 9 shows the HFS-decoded CpG islands marked on the genome sequence, and Table 1 reports CpG-island filtering information for each computed CpG island.

From Fig. 8 we see that the proportion of CpG dinucleotides on this genome subsequence bA111E21 is less than one quarter of the expected proportion $(1/16)$. Its recurrence-time histogram and log-probability plot together indicate rather vivid aggregating patterns. By visualizing Fig. 9, the HFS algorithm seems to capture all potential aggregating segments on the genome subsequence. The performance of HFS is confirmed in Table 1. All computed CpG islands are very close to the three CpG filtering criteria, except some are slightly shorter than 200bp. In comparison, two islands located at $(30312\ldots 30696)$ and $(113102\ldots 113307)$ are identified through the EMBOSS program, available online at the website (http://www.ebi.ac.uk/emboss/cpgplot). These two islands are almost completely covered by two HFS-decoded CpG islands located at $(30338, 30844)$ and $(113128, 113452)$. Apparently, the EMBOSS program has missed many segments that satisfy the CpG filtering criteria.

The next important questions are: How many CpG islands are missed by the HFS algorithm? Are all HFS-decoded CpG islands real? These two questions will be ad-

(a)

A	C	G	T
40645	27484	29698	39891

$$\hat{P}_{CG} = \frac{1655}{137717} = 0.0120174$$

(b)

	A	C	G	T
A	13536	7050	9666	10393
C	9526	7043	1655	9259
G	8811	5552	7939	7396
T	8772	7839	10437	12843

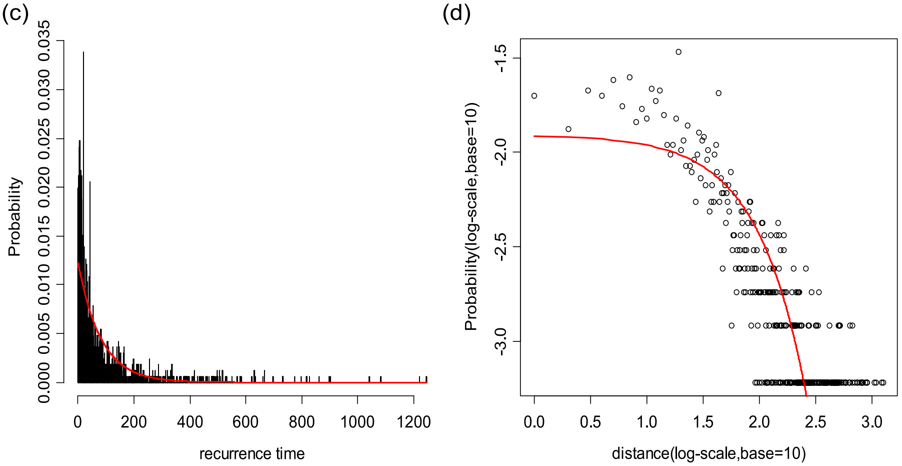


Fig. 8 Summarized information of the genome subsequence bA111E21. (**a**) The A, G, T, and C nucleotides proportions; (**b**) 16 dinucleotides contents; (**c**) the recurrence time of CpG dinucleotides and the maximum entropy geometric distribution (*the smooth curve*) based on average recurrence time $1/\hat{P}_{GC}$ and the *red line* for maximum-entropy geometric distribution; (**d**) the same comparison as of panel (**c**) on log-probability vs. log-length scales

dressed thoroughly in a companion report. The latter question is addressed by matching HFS-decoded CpG islands with existing databases on promoters as proposed in [26] and other experimentally confirmed databases. Finding CpG islands is such an active field nowadays. Potentially the HFS algorithm together with some learning rules sustained by experimental databases will allow scientists to compute and recognize CpG islands with high reliability in the near future.

7 Discussion

The HFS algorithm is proposed in this paper as a brand-new method of nonparametric decoding for lengthy discrete time series. We reveal that HFS indeed has high potential for decoding a binary state-space vector through the evidence derived from computer experiments and real genetic data analysis on CpG islands. Since it requires no structural assumptions, its performance naturally is very robust and free of systematic bias. For the same reason, its spectrum of applicability will be rather wide. Here we mention a few successful applications of variant versions of our nonparametric decoding computations. For instance, the HFS algorithm is used to dissect

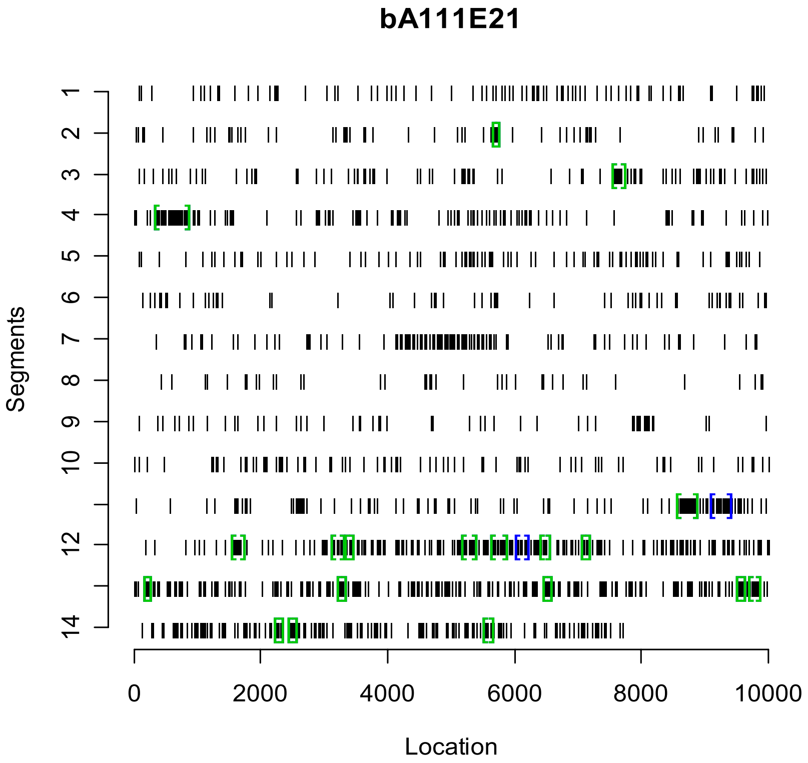


Fig. 9 HFS-decoded CpG islands marked with braces on the genome subsequence bA111E21

the circadian rhythmic cycles of cockroach [8], and to identify the patterns of coherent vs. non-coherent dyadic interaction of a human couple [9]. In fact, our development can also be applied to discrete-time series of continuous measurement, such as stock price in financial market [10]. The potentially wide applicability of HFS algorithm rests on its very minimum requirement. Hopefully this simple computational approach is found useful in many studies of dynamic systems with various data types.

Rigorous and theoretical developments for the HFS algorithm, beyond the intensity change-point perspective considered here, will be further studied from the perspectives of coding and data compression and Rissanen's statistical complexity. The fact that its construction rests on a hierarchy of recurrent time sequences and distributions seems solid enough to allow it to be able to bring out dynamic patterns of the observed time series, especially the aggregating pattern of certain chosen events. This objective segmentation algorithm can be further extended to decoding discrete time series generated from non-autonomous systems with internal state variables taking values for more than two states. The computer experimental results reported here also raise important theoretical and practical issues regarding the robustness of the Viterbi algorithm as the most popular decoding technique. More future research effort and attention is necessary to adequately address these issues.

We further mention the modeling difficulties that the Viterbi and its variant algorithms would face in the CpG-islands example. Originally any genome subsequence,

Table 1 Locations of CpG-islands identified by HFS algorithm with three filtering information and colored segments missed by EMBOSS program

Lstar	9721	15624	27553	29719	30338	45157	64127	88033	102484	108564
Lstop	9867	15736	27770	30019	30844	45394	65682	88179	102664	108900
length > 200	147	113	218	301	507	238	1556	147	181	337
GC content > 50%	0.544218	0.557522	0.555046	0.538206	0.65286	0.579832	0.522494	0.469388	0.436464	0.715134
O/E ratio > 0.6	0.572549	0.941176	0.675057	0.567108	0.700123	0.505115	0.637363	1.055104	0.833333	0.724138
Lstar	109011	111560	112960	113128	113517	115022	115186	115490	115648	116017
Lstop	109652	111759	113017	113452	113630	115126	115371	115565	115956	116236
length > 200	642	200	58	325	114	105	186	76	309	220
GC content > 50%	0.680685	0.47	0.706897	0.652308	0.640351	0.666667	0.612903	0.710526	0.631068	0.568182
O/E ratio > 0.6	0.492036	0.8284	0.87468	0.728155	0.523552	0.700337	0.699874	0.652174	0.655808	0.570313
Lstar	116296	117071	117280	118553	119964	120173	123222	123441	124851	126262
Lstop	116547	117215	117416	118840	120043	120310	123366	123567	125138	126341
length > 200	252	145	137	288	80	138	145	127	288	80
GC content > 50%	0.607143	0.655172	0.591241	0.576389	0.575	0.630435	0.648276	0.598425	0.583333	0.575
O/E ratio > 0.6	0.568467	0.609023	0.56044	0.613248	1.781955	0.88961	0.555985	0.593407	0.602941	1.781955
Lstar	126472	129505	129714	130897	132224	132456	134473	135525	135734	137207
Lstop	126611	129649	129850	131130	132333	132604	134592	135669	135852	137440
length > 200	140	145	137	234	110	149	120	145	119	234
GC content > 50%	0.621429	0.655172	0.59854	0.589744	0.572727	0.624161	0.4	0.662069	0.630252	0.589744
O/E ratio > 0.6	0.982065	0.614509	0.665734	0.62973	0.963384	0.979669	1.342105	0.671016	0.59	0.62973

such as the one labeled by bA111E21, is a symbolic string of A, G, T, and C (nucleotides). So there are 16 possible pairs of dinucleotides, and one of them, the CG ordered pair, is taken as an event of interest. The application of HMM will face two deterministic constraints: (1) given the previous symbol, there are only 4 possible dinucleotides to be seen, not 16; (2) after observing a CpG, the next pair is certainly not to be a CpG. To a great extent, these constraints will complicate the Markovian structure underlying the Viterbi and its variant algorithms as natural candidates for decoding. These complications will change the graphic representation for the Markov chain and consequently cause computational complexity. It is worth noting that these constraints have no impact at all on our HFS algorithm. We simply represent the genome sequence as a 0–1 digital string, as in Fig. 9. The HFS algorithm takes the 0–1 string as its data and performs the decoding computations. This is the great advantage of our HFS algorithm.

In fact, the CpG-islands example is not particular; all dynamic programming based decoding techniques are prone to sensitivity to the structural assumptions on the state variable $S(t)$, especially when analyzing lengthy dynamic time series. Hence a nonparametric decoding algorithm would be not only desirable, but also crucial in scientific endeavors.

A popular statistical technique called scan statistics [22] is often applied to mark or detect defective regions without assuming the state-space structure in quality control literature. It employs the sliding window idea as EMBOSS program does. And its main developments are focused on the distributions of the maximum and minimum numbers of data points, or events, contained in the window. To perform the quality control, these distributions are used as the bases for judging where the events of interest might be too aggregating, and where the events might be too sparse. Again, the efficiency of scan statistics critically depends on the window width. Hence it is expected to also share the same drawback of EMBOSS program on CpG-islands example. Thus this technique likely misses a large percentage of target regions that are lengthier than the window width, but have heterogeneous intensity of event.

References

1. Baum LE, Petrie T, Soules G, Weiss N (1970) A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. Ann Math Stat 41:164–171
2. Bird AP (1986) CpG islands and the function of DNA methylation. Nature 321:209–213
3. Bock C, Walter J, Paulsen M, Lengauer T (2006) CpG island mapping by epigenome prediction. PLoS Comput Biol 6:1055–1069
4. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Ser B 39(1):1–38
5. Durbin R, Eddy S, Krogh A, Mitchison G (1998) Biological sequence analysis. Cambridge University Press, Cambridge
6. Durbin J, Koopman SJ (2001) Time series analysis by state space methods. Oxford University Press, New York
7. Fushing H, Hwang CR, Lee HC, Lan YC, Horng SB (2006) Testing and mapping non-stationarity in animal behavioural processes: a case study on an individual female bean weevil. J Theor Biol 238:805–816

8. Fushing H, Chen SC, Lee HJ (2010) Statistical computations on biological rhythms I: Dissecting variable cycles and computing phase shifts in activity event time series. J Comput Graph Stat (in press)
9. Fushing H, Ferrer E, Chen SC, Chow SM (2010) Dynamics of dyadic interaction I: Exploring nonstationarity of intra- and inter-individual affective processes via hierarchical segmentation and stochastic small-world networks. Psychometrika (in press)
10. Fushing H, Chen SC, Hwang C-R (2010) Discovering stock dynamics through multidimensional volatility phases. Quant Finance (in press)
11. Gardiner-Garden M, Frommer M (1987) CpG islands in vertebrate genomes. J Mol Biol 196:261–282
12. Geman S, Kochanek K (2001) Dynamic programming and the graphic representation of error-correcting codes. IEEE Trans Inf Theory 47:549–568
13. Hamilton JD (2005) What's real about the business cycles? Fed Reserve Bank St Louis Rev 87(4):435–452
14. Hsieh F, Turnbull B (1996) Non- and semi-parametric estimation of the receiver operating characteristics (ROC) curve. Ann Stat 24:25–40
15. Jaynes ET (1957) Information theory and statistical mechanics. Phys Rev 106:620–630
16. Jaynes ET (1957) Information theory and statistical mechanics II. Phys Rev 108:171–190
17. Kalman RE (1960) A new approach of linear filtering and prediction problems. J Basic Eng Trans ASMA Ser D 82:35–45
18. Laird PW (2005) Cancer epigenetics. Hum Mol Genet 14:R65–R76
19. Lanterman AD (2001) Schwarz, Wallace, and Rissanen: Intertwining themes in theories of model order estimation. Int Stat Rev 69:185–212
20. Lee TCM (2001) An introduction to coding theory and the two-part minimum description length principle. Int Stat Rev 69:169–183
21. Manuca R, Savit R (1996) Stationarity and nonstationarity in time series analysis. Physica D 99:134–161
22. Naus J (1982) Approximations for distributions of scan statistics. J Am Stat Assoc 77:177–182
23. Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77:257–286
24. Rissanen J (1996) Fisher information and stochastic complexity. IEEE Trans Inf Theory 42:40–47
25. Rissanen J (1997) Stochastic complexity in learning. J Comput Syst Sci 55:89–95
26. Saxonov S, Berg P, Brutlag D (2006) A genome-wide analysis of CpG dinucleotides in the human genome distinguishes two distinct classes of promoters. Proc Natl Acad Sci USA 103:1412–1417
27. Schwarz G (1978) Estimating the dimension of a model. Ann Stat 6:461–464
28. Takai D, Jones PA (2002) Comprehensive analysis of CpG islands in human chromosomes 21 and 22. Proc Natl Acad Sci USA 99:3740–3745
29. Viterbi AJ (1967) Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. IEEE Trans Inf Theory 13:260–269
30. West M, Harrison J (1997) Bayesian forecasting and dynamic models, 2nd edn. Springer, New York