JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 17, Number 12, 2010

Pp. 1697–1709
DOI: 10.1089/cmb.2008.0233

Importance Sampling of Word Patterns in DNA and Protein Sequences

*HOCK PENG CHAN,[1] *NANCY RUONAN ZHANG,[3] and LOUIS H.Y. CHEN[2]

ABSTRACT

Monte Carlo methods can provide accurate *p*-value estimates of word counting test statistics and are easy to implement. They are especially attractive when an asymptotic theory is absent or when either the search sequence or the word pattern is too short for the application of asymptotic formulae. Naive direct Monte Carlo is undesirable for the estimation of small probabilities because the associated rare events of interest are seldom generated. We propose instead efficient importance sampling algorithms that use controlled insertion of the desired word patterns on randomly generated sequences. The implementation is illustrated on word patterns of biological interest: palindromes and inverted repeats, patterns arising from position-specific weight matrices (PSWMs), and co-occurrences of pairs of motifs.

Key words: importance sampling, Monte Carlo, motifs, palindromes, position-specific weight matrices.

1. INTRODUCTION

SEARCHING FOR MATCHES TO A WORD PATTERN, also called a motif, is an important task in computational biology. The word pattern usually represents a functional site, such as a transcription factor binding site (TFBS) in a promoter region of a DNA sequence or a ligand docking site in a protein sequence. Statistical significance of over-representation of these word patterns provides valuable clues to biologists. Consequently, much work has been done on the use of asymptotic limiting distributions to approximate these *p*-values (Prum et al., 1995; Reinert et al., 2000; Régnier, 2000; Robin et al., 2002; Huang et al., 2004; Leung et al., 2005; Mitrophanov and Borodovsky, 2006; Pape et al., 2008). However, the approximations may not be accurate for short words or for words consisting of repeats and most theoretical approximations work only in specific settings. String-based recursive methods can provide exact *p*-values (Gusfield, 1997), but they can be computationally expensive when the number of words in the word pattern is large.

Direct Monte Carlo algorithms for estimating *p*-values of word patterns are easy to implement but are inefficient for the estimation of very small *p*-values, because in such cases, almost all the simulated sequences do not contain the required number of word patterns. We propose in this article importance sampling algorithms that insert the desired word patterns, either randomly or controlled by a hidden Markov

[1]Department of Statistics and Applied Probability, and [2]Institute for Mathematical Sciences, National University of Singapore, Singapore, Republic of Singapore.
[3]Department of Statistics, Stanford University, Stanford, California.
*These two authors are joint first authors.

model, on the simulated sequences. The algorithms are described in Section 2 and are illustrated on several word patterns of biological interest: palindromes and inverted repeats in Section 3, high-scoring words with respect to position-specific weight matrices (PSWMs) in Section 4, and co-occurrences of motifs in Section 5. Numerical results show that variance reduction of several orders of magnitude are achieved when applying the proposed importance sampling algorithms on small p-values. The technical details are consolidated in Appendices A–D and include a proof of the asymptotic optimality of the importance sampling algorithms (see Appendix D).

2. IMPORTANCE SAMPLING OF WORD PATTERNS

2.1. Word counting

Let $|B|$ denote the number of elements in a set B. By selecting randomly from a finite set B, we shall mean that each $b \in B$ has probability $|B|^{-1}$ of being selected. For any two sequences $\mathbf{v} = v_1 \cdots v_m$ and $\mathbf{u} = u_1 \cdots u_r$, the notation $\mathbf{vu}$ shall denote the concatenated sequence $v_1 \cdots v_m u_1 \cdots u_r$. We also denote the length of $\mathbf{v}$ by $\ell(\mathbf{v})(=m)$. Although we assume implicitly an alphabet $\chi = \{a, c, g, t\}$, representing the four nucleotide bases of DNA sequences, the algorithms can be applied on any countable alphabet, for example the alphabet of 20 amino acids in protein sequences.

We will represent the word pattern of interest by a set of words $\mathcal{V}$ and assume that $|\mathcal{V}| < \infty$. Let $\mathbf{s} = s_1 \cdots s_n$ denote a sequence of DNA bases under investigation and let N_m be the maximum number of non-overlapping words from $\mathcal{V}$ in $\mathbf{s}_m = s_1 \cdots s_m$. We say that there exists a word in $\mathcal{V}$ at the end of $\mathbf{s}_m$ if $s_{m-j+1} \cdots s_m \in \mathcal{V}$ for some $j > 0$. Moreover, the smallest such j is the length of the shortest word at the end of $\mathbf{s}_m$. We have the recursive relations, for $m \geq 1$,

$$N_m = \begin{cases} N_{m-1} & \text{if there is no word in } \mathcal{V} \text{ at the end of } \mathbf{s}_m, \\ N_{m-j} + 1 & \text{if the shortest word in } \mathcal{V} \text{ at the end of } \mathbf{s}_m \text{ is of length } j, \end{cases} \tag{2.1}$$

with the initialization $N_0 = 0$. We denote N_n simply by N. It is also possible to modify (2.1) to handle the counting of possibly overlapping words.

2.2. Monte Carlo evaluation of statistical significance

We begin by describing direct Monte Carlo. To evaluate the signifiance of observing c word patterns in an observed sequence $\mathbf{s}$, we generate independent copies of the sequence from a Markov chain with transition probabilities estimated either from $\mathbf{s}$ or from a local neighborhood of $\mathbf{s}$. The proportion of times $\{N \geq c\}$ occurs among the independent copies of $\mathbf{s}$ is then the direct Monte Carlo estimate of the p-value $p_c := P\{N \geq c\}$.

It is quite common for many sequences to be analyzed simultaneously. Hence to correct for the effect of multiple comparisons, a very small p-value is required for any one sequence before statistical significance can be concluded. Direct Monte Carlo is well-known to be very inefficient for estimating small probabilities in general and many importance sampling schemes have been proposed to overcome this drawback, for example, in sequential analysis (Siegmund, 1976), communication systems (Cottrell et al., 1983), bootstrapping (Johns, 1988; Do and Hall, 1992), signal detection (Lai and Shan, 1999), moderate deviations (Fuh and Hu, 2004), and scan statistics (Chan and Zhang, 2007). In this article, we provide change of measures that are effective for the importance sampling of word patterns.

For ease of exposition, assume that the background sequence of bases follows a first-order Markov chain with positive transition probabilities

$$\sigma(xy) := P\{s_{i+1} = y | s_i = x\}, \quad x, y \in \chi. \tag{2.2}$$

Let π be the stationary distribution, and let $\sigma(v_1 \cdots v_i) = \prod_{j=1}^{i-1} \sigma(v_j v_{j+1})$. Before executing the importance sampling algorithms, we first create a word bank of the desired word pattern, with each word in the word bank taking the value $\mathbf{v} \in \mathcal{V}$ with probability $q(\mathbf{v}) > 0$. The procedure for the selection of q and construction of the word banks will be elaborated in Sections 3–5. For completeness, we define $q(\mathbf{v}) = 0$ when $\mathbf{v} \notin \mathcal{V}$. Let $\beta(\mathbf{v}) = q(\mathbf{v})/\sigma(\mathbf{v})$. For ease of computation, we shall generate a dummy variable s_0 before generating s and denote $s_0 \cdots s_n$ by $\mathbf{s}_0$. The first importance sampling algorithm, for the estimation of p_1 only, is as follows.

Algorithm A (for $c=1$):

1. Select a word $\mathbf{v}$ randomly from the word bank. Hence the word takes the value $\mathbf{v} \in \mathcal{V}$ with probability $q(\mathbf{v})$.
2. Select i_0 randomly from $\{1, \ldots, n-\ell(\mathbf{v})+1\}$.
3. Generate s_0 from the stationary distribution and $s_1, \ldots, s_{i_0-1}$ sequentially from the underlying Markov chain. Let $s_{i_0} \cdots s_{i_0+\ell(\mathbf{v})-1} = \mathbf{v}$ and generate $s_{i_0+\ell(\mathbf{v})}, \ldots, s_n$ sequentially from the underlying Markov chain.

Let $\ell_{\min} = \min_{\mathbf{v}\in\mathcal{V}} \ell(\mathbf{v})$ and $\ell_{\max} = \max_{\mathbf{v}\in\mathcal{V}} \ell(\mathbf{v})$. Recall that $\beta(\mathbf{v})=0$ for $\mathbf{v} \notin \mathcal{V}$. Then

$$L(\mathbf{s}_0) := \sum_{\ell=\ell_{\min}}^{\ell_{\max}} (n-\ell+1)^{-1} \sum_{i=1}^{n-\ell+1} \beta(s_i \cdots s_{i+\ell-1})/\sigma(s_{i-1}s_i) \tag{2.3}$$

is the likelihood ratio of generating $\mathbf{s}_0$ from Algorithm A and from the underlying Markov chain (with no insertion of word patterns). If Algorithm A is run independently K times, with the kth copy of $\mathbf{s}_0$ generated denoted by $\mathbf{s}_0^{(k)}$, then

$$\widehat{p}_{\mathrm{I}} := K^{-1} \sum_{k=1}^{K} L^{-1}(\mathbf{s}_0^{(k)}) \mathbf{1}_{\{N^{(k)}\geq c\}} \tag{2.4}$$

is unbiased for p_c. The term $\mathbf{1}_{\{N^{(k)}\geq c\}}$ is superfluous when using Algorithm A since at least one word pattern from $\mathcal{V}$ is generated in every copy of $\mathbf{s}_0$.

We restrict Algorithm A to $c=1$ because the random insertion of more than one word patterns into the simulated sequence can result in a hard to compute likelihood ratio. To handle more general c, we use a hidden Markov model device in Algorithm B below, with hidden states X_i taking either value 0 (do not insert word pattern) or 1 (insert word pattern), so that the likelihood ratio can be computed recursively. Let

$$\rho_i = P\{X_i = 1 | s_0 \cdots s_i\} \tag{2.5}$$

be the word insertion probability at position $i+1$ along the DNA sequence. For example, the user can simply select $\rho_i = c/n$ for all i so that approximately c word patterns are inserted in each generated sequence $\mathbf{s}_0$. Each copy of $\mathbf{s}_0$ is generated in the following manner.

Algorithm B (for $c \geq 1$):

1. Let $i=0$, generate s_0 from the stationary distribution and X_0 satisfying (2.5).
2. (a) If $X_i = 1$, select a word $\mathbf{v}$ randomly from the word bank. If $\ell(\mathbf{v}) \leq n-i$, that is, if the word $\mathbf{v}$ can fit into the remaining sequence, let $s_{i+1} \cdots s_{i+\ell(\mathbf{v})} = \mathbf{v}$, generate $X_{i+\ell(\mathbf{v})}$ according to (2.5), increment i by $\ell(\mathbf{v})$ and go to step 3.
   (b) If the word selected in 2(a) cannot fit into the remaining sequence or if $X_i = 0$, generate s_{i+1} from the underlying Markov chain and X_{i+1} satisfying (2.5). Increment i by 1 and go to step 3.
3. If $i < n$, repeat step 2. Otherwise, end the recursion.

Let $L_i = L_i(s_0 \cdots s_i)$ be the likelihood ratio of generating $s_0 \cdots s_i$ from Algorithm B and from the underlying Markov chain. Let $\gamma_j = \sum_{\mathbf{v}\in\mathcal{V}:\ell(\mathbf{v})\leq j} q(\mathbf{v})$ be the probability that a randomly chosen word from the word bank has length not exceeding j. Then

$$L_i = (1-\rho_{i-1}\gamma_{n-i+1})L_{i-1} + \sum_{\ell=\ell_{\min}}^{\ell_{\max}} \rho_{i-\ell} L_{i-\ell} \beta(s_{i-\ell+1} \cdots s_i)/\sigma(s_{i-\ell}s_{i-\ell+1}) \text{ if } i \geq 1, \tag{2.6}$$

with $L_i = 0$ for $i \leq 0$.

The estimator (2.4), with $L = L_n$, is unbiased if and only if all configurations of $\mathbf{s}_0$ satisfying $N \geq c$ can be generated via Algorithm B. To ensure this, it suffices for us to impose the constraint

$$\rho_i < 1 \text{ for all } i < n - \ell_{\min}(c - N_i), \tag{2.7}$$

so that we do not force the insertion of too many word patterns.

3. PALINDROMIC PATTERNS AND INVERTED REPEATS

Masse et al. (1992) reported clusters of palindromic patterns near origins of replication of viruses. There has been much work done to estimate their significance, for example, using Poisson and compound Poisson approximations (Leung et al., 1994, 2005). The four nucleotides can be divided into two complementary base pairs with a and t forming a pair and c and g forming the second pair. We denote this relation by writing $a^c = t$, $t^c = a$, $c^c = g$ and $g^c = c$. For a word $\mathbf{u}_m = u_1 \cdots u_m$, we define its complement $\mathbf{u}_m^c = u_m^c \cdots u_1^c$. A palindromic pattern of length $\ell = 2m$ is a DNA sequence that can be expressed in the form $\mathbf{u}_m \mathbf{u}_m^c$. For example, $\mathbf{v} = acgcgt$ is a palindromic pattern. Note that the complement of $\mathbf{v}$, that is the word obtained by replacing each letter of $\mathbf{v}$ by its complement, is $tgcgca$, which is just $\mathbf{v}$ read backwards. This interesting property explains the terminology "palindromic pattern."

Inverted repeats can be derived from palindromic patterns by inserting a DNA sequence of length d in the exact middle of the pattern. The class of word patterns for inverted repeats can be expressed in the form

$$\mathcal{V} = \{\mathbf{u}_m \mathbf{z} \mathbf{u}_m^c : d_1 \le \ell(\mathbf{z}) \le d_2\}, \tag{3.1}$$

with $0 \le d_1 \le d_2$. When $d_1 = d_2 = 0$, then (3.1) is the class of all palindromic patterns of length $2m$.

The construction of word banks for palindromic patterns is straightforward. It all boils down to generating $\mathbf{u}_m$ in some suitable manner. We advocate generating $\mathbf{u}_m$ with probability proportional to $\pi(u_1)\sigma(\mathbf{u}_m)\sigma(\mathbf{u}_m^c)$ or $\pi(u_1)\sigma(\mathbf{u}_m \mathbf{u}_m^c)$ and show how this can be done in Appendix A.

Having a word bank for palindromic patterns allows us to create a word bank for inverted repeats easily. The procedure is as follows.

1. Select $\mathbf{u}_m \mathbf{u}_m^c$ randomly from a word bank of palindromic patterns and d randomly from $\{d_1, \ldots, d_2\}$.
2. Let $z_0 = u_m$ and generate $z_1, \ldots, z_d$ sequentially from the underlying Markov chain.
3. Store the word $\mathbf{u}_m \mathbf{z}_d \mathbf{u}_m^c$ into the word bank for inverted repeats.

This procedure allows γ_j, see (2.6), to be computed easily. In particular, $\gamma_j = (j - d_1 + 1)/(d_2 - d_1 + 1)$ for $d_1 \le j \le d_2$, $\gamma_j = 0$ for $j < d_1$ and $\gamma_j = 1$ for $j > d_2$.

4. POSITION-SPECIFIC WEIGHT MATRIX

PSWMs are commonly used to derive fixed-length word patterns or motifs that transcription factors bind onto and usually range from four to twenty bases long. Databases such as TRANSFAC, JASPAR, and SCPD curate PSWMs for families of transcription factors. For example, the PSWM for the SWI5 transcription factor in the yeast genome (Zhu and Zhang, 1999) is

$$\begin{array}{c} \\ a \\ c \\ g \\ t \end{array} \begin{pmatrix} 4 & 0 & 4 & 1 & 1 & 4 & 0 & 0 & 0 & 0 & 0 & 2 \\ 1 & 2 & 1 & 1 & 3 & 2 & 0 & 0 & 7 & 0 & 0 & 0 \\ 2 & 2 & 0 & 2 & 1 & 0 & 2 & 7 & 0 & 0 & 7 & 5 \\ 0 & 3 & 2 & 3 & 2 & 1 & 5 & 0 & 0 & 7 & 0 & 0 \end{pmatrix}. \tag{4.1}$$

Let $w_i(v)$ denote the entry in a PSWM that corresponds to base v at column i and let m be the number of columns in the PSWM. For any word $\mathbf{v}_m$ (of length m), a score

$$S(\mathbf{v}_m) := \sum_{i=1}^{m} w_i(v_i)$$

is computed and words with high scores are of interest. We let $\mathcal{V}$ be the set of all $\mathbf{v}_m$ with score not less than a pre-specified threshold level t. In other words,

$$\mathcal{V} = \{\mathbf{v}_m : S(\mathbf{v}_m) \ge t\} \tag{4.2}$$

is a set of motifs for the PSWM associated with a given transcription factor. The matrix is derived from the frequencies of the four bases at various positions of known instances of the TFBS, which are usually confirmed by biological experiments. Huang et al. (2004) provide a good review of the construction of PSWMs.

In principle, we can construct a word bank for $\mathcal{V}$ by simply generating words of length m from the underlying Markov chain and discarding words that do not belong to the motif. However, for t large, such a procedure involves discarding a large proportion of the generated words. It is more efficient to generate the words with a bias towards larger scores. In Appendix B, we show how, for any given $\theta > 0$, a tilted Markov chain can be constructed to generate words $\mathbf{v}$ with probability mass function

$$q_\theta(\mathbf{v}) = e^{\theta S(\mathbf{v})}\pi(v_1)\sigma(\mathbf{v})/\Lambda(\theta), \tag{4.3}$$

where $\Lambda(\theta)$ is a computable normalizing constant. If words with scores less than t are discarded, then the probability mass function of non-discarded words is

$$q(\mathbf{v}) = \xi e^{\theta S(\mathbf{v})}\pi(v_1)\sigma(\mathbf{v})/\Lambda(\theta) \quad \text{for } \mathbf{v} \in \mathcal{V}, \tag{4.4}$$

where ξ is an unknown normalizing constant that can be estimated by the reciprocal of the fraction of non-discarded words. There are two conflicting demands placed on the choice of θ. As θ increases, the expected score of words generated under $q_\theta(\mathbf{v})$ increases. We would thus like θ to be large so that the fraction of discarded words is small. However at the same time, we would also like θ to be small, so that the variation of $\beta(\mathbf{v}) = q(\mathbf{v})/\sigma(\mathbf{v})$ over $\mathbf{v} \in \mathcal{V}$ is small. Since

$$E_{q_\theta}[S(\mathbf{v})] = \frac{d}{d\theta}[\log \Lambda(\theta)], \tag{4.5}$$

we suggest choosing the root of the equation $\frac{d}{d\theta}[\log \Lambda(\theta)] = t$. See Appendix B for more detail on the the computation of $\Lambda(\theta)$ and the numerical search of the root.

4.1. Example 1

We illustrate here the need for alternatives to analytical p-value approximations by applying Algorithm A on some special word patterns. Let P_π denotes probability with v_1 following stationary distribution π. Huang et al. (2004) suggested an approximation, which for $c = 1$ reduces to

$$P\{N \geq 1\} \doteq 1 - (1 - P_\pi\{S(\mathbf{v}_m) \geq t\})^{n-m+1}. \tag{4.6}$$

Consider $s_1, \ldots, s_n$ independent and identically distributed random variables taking values a, c, g and t with equal probabilities. Let

$$W_{\text{rep}} = \begin{array}{c} a \\ c \\ g \\ t \end{array}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{4.7}$$

$$W_{\text{norep}} = \begin{array}{c} a \\ c \\ g \\ t \end{array}\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.8}$$

and consider counting of words with score at least t for $t = 9$, 10 and 11. The approximation (4.6) is the same for both (4.7) and (4.8) but we know that the p-value when the PSWM is (4.7) should be smaller due to the tendency of the word patterns to clump together. Of course, declumping corrections can be applied to this special case but this is not so straightforward for general PSWMs. Table 1 compares the analytical, direct Monte Carlo and importance sampling approximations of $P\{N \geq 1\}$ for (4.7) and (4.8) with $n = 200$. The simulations reveal substantial over-estimation of p-values for W_{rep} when using (4.6). Algorithm A is able to maintain its accuracy over the range of t considered whereas direct Monte Carlo has acceptable accuracy only for $t = 9$.

TABLE 1. COMPARISONS OF ANALYTICAL, DIRECT MONTE CARLO, AND IMPORTANCE SAMPLING APPROXIMATIONS FOR $P\{N \geq 1\}$ WITH $n = 200$ IN EXAMPLE 1

t	*9*	*10*	*11*
Analytical	7.1×10^{-2}	7.1×10^{-3}	4.2×10^{-4}
W_{rep}			
Direct MC	$(3.6 \pm 0.6)\times10^{-2}$	$(5 \pm 2)\times10^{-3}$	0
Algorithm A	$(3.0 \pm 0.1)\times10^{-2}$	$(4.0 \pm 0.2)\times10^{-3}$	$(2.7 \pm 0.1)\times10^{-4}$
W_{norep}			
Direct MC	$(6.7 \pm 0.8)\times10^{-2}$	$(9 \pm 3)\times10^{-3}$	$(1 \pm 1)\times10^{-3}$
Algorithm A	$(7.5 \pm 0.2)\times10^{-2}$	$(6.9 \pm 0.2)\times10^{-3}$	$(4.1 \pm 0.1)\times10^{-4}$

Each Monte Carlo entry is obtained using 1000 simulation runs and are expressed in the form $\widehat{p} \pm$ standard error.

4.2. *Example 2*

We implement Algorithm B here with

$$\rho_i = \min\left\{1, \left(\frac{c - N_i}{n - i - (c - N_i)(m-1)}\right)^+\right\}, \tag{4.9}$$

where $x^+ = \max\{0, x\}$. We choose ρ_i in this manner to encourage word insertion when there are few bases left to be generated and the desired number of word patterns has not yet been observed. The motif consists of all words of length 12 having score at least 50 with respect to the PSWM (4.1). The transition matrix for generating the DNA sequence is

$$\begin{array}{c} a \\ c \\ g \\ t \end{array}\begin{pmatrix} .3577 & .1752 & .1853 & .2818 \\ .3256 & .2056 & .1590 & .3096 \\ .2992 & .2180 & .2039 & .2789 \\ .2381 & .1943 & .1905 & .3771 \end{pmatrix}, \tag{4.10}$$

and the length of the sequence investigated is $n = 700$. We see from Table 2 variance reduction of 10–100 times in the simulation of probabilities of order 10^{-1} to 10^{-3}. For smaller probabilities, direct Monte Carlo does not provide an estimate whereas estimates from the importance sampling algorithm retain their accuracy. Although importance sampling takes about two times the computing time of direct Monte Carlo for each simulation run, the savings in computing time to achieve the same level of accuracy are quite substantial.

5. CO-OCCURRENCES OF MOTIFS

For a more detailed sequence analysis of promoter regions, one can search for cis-regulatory modules (CRM) instead of single motifs. We define CRM to be a collection of fixed length motifs that are located in a fixed order in proximity to each other. They are signals for co-operative binding of transcription factors, and are important in the study of combinatorial regulation of genes. CRMs have been used successfully to gain a deeper understanding of gene regulation (Chiang et al., 2003; Zhou and Wong, 2004; Zhang et al., 2007). We focus here on the simplest type of CRM: A co-occurring pair of high scoring words separated by

TABLE 2. $\widehat{p} \pm$ STANDARD ERROR FOR EXAMPLE 2 WITH 1000 COPIES OF s_0 GENERATED FOR BOTH DIRECT MONTE CARLO AND IMPORTANCE SAMPLING USING ALGORITHM B

c	*Direct MC*	*Algorithm B*
1	$(9.6 \pm 0.9)\times10^{-2}$	$(9.1 \pm 0.3)\times10^{-2}$
2	$(3 \pm 2)\times10^{-3}$	$(4.2 \pm 0.2)\times10^{-3}$
3	0	$(1.3 \pm 0.1)\times10^{-4}$
4	0	$(2.6 \pm 0.3)\times10^{-6}$

a gap sequence of variable length. Let $S_1(\cdot)$ be the score of a word of length m calculated with respect to a PSWM W_1, and $S_2(\cdot)$ the score of a word of length r calculated with respect to a PSWM W_2. Let $0 \le d_1 < d_2 < \infty$ be the prescribed limits of the length of the gap and t_1, t_2 threshold levels for W_1 and W_2, respectively. The family of words for the co-occurring motifs is

$$\mathcal{V} = \{\mathbf{v}_m \mathbf{z} \mathbf{u}_r : S_1(\mathbf{v}_m) \ge t_1, S_2(\mathbf{u}_r) \ge t_2, d_1 \le \ell(\mathbf{z}) \le d_2\}. \tag{5.1}$$

In Section 4, we showed how word banks for the motifs $\mathcal{V}_1 := \{\mathbf{v}_m : S_1(\mathbf{v}_m) \ge t_1\}$ and $\mathcal{V}_2 := \{\mathbf{u}_r : S_2(\mathbf{u}_r) \ge t_2\}$ are created. Let q_i be the probability mass function for $\mathcal{V}_i$. A word bank for $\mathcal{V}$ can then be created by repeating the following steps.

1. Select $\mathbf{v}_m$ and $\mathbf{u}_r$ independently from their respective word banks.
2. Select d randomly from $\{d_1, \ldots, d_2\}$. Generate $z_1, \ldots, z_d$ sequentially from the underlying Markov chain, initialized at $z_0 = v_m$.
3. Store $\mathbf{w} = \mathbf{v}_m \mathbf{z}_d \mathbf{u}_r$ into the word bank.

Let q be the probability mass function of the stored words. Then

$$q(\mathbf{w}) = (d_2 - d_1 + 1)^{-1} q_1(\mathbf{v}_m) \sigma(v_m \mathbf{z}_d) q_2(\mathbf{u}_r) \tag{5.2}$$

and hence $\beta(\mathbf{w}) = q(\mathbf{w})/\sigma(\mathbf{w}) = (d_2 - d_1 + 1)^{-1} \beta_1(\mathbf{v}_m) \beta_2(\mathbf{u}_r)/\sigma(z_d u_1)$.

5.1. *Example 3*

The transcription factors SFF (with PSWM W_1) and MCM1 (with PSWM W_2) are regulators of the cell cycle in yeast, and are known to co-operate at close distance in the promoter regions of the genes they regulate (Spellman et al., 1998). Their PSWMs can be obtained from the database SCPD. Define $\mathcal{V}$ by (5.1) with $t_1 = 48$, $t_2 = 110$, $d_1 = 0$ and $d_2 = 100$. We would like to estimate the probability that the motif $\mathcal{V}$ appears at least once within a promoter sequence of length $n = 700$. The estimated probability using Algorithm A is 3.4×10^{-3} with a standard error of 3×10^{-4}. The corresponding standard error for 1000 direct Monte Carlo runs would have been about 2×10^{-3}, which is large relative to the underlying probability.

5.2. *Structured motifs*

These co-occurring motifs considered in Robin et al. (2002) consist essentially of fixed word patterns $\mathbf{x}_m$ and $\mathbf{y}_r$ separated by a variable length gap, with an allowance for the mutation of up to one base in $\mathbf{x}_m \mathbf{y}_r$. The motif can be expressed as

$$\mathcal{V} = \{\mathbf{v}_m \mathbf{z} \mathbf{u}_r : d_1 \le \ell(\mathbf{z}) \le d_2, |\{i : v_i \ne x_i\}| + |\{i : u_i \ne y_i\}| \le 1\}. \tag{5.3}$$

We create a word for the word bank of $\mathcal{V}$ in the following manner.

1. Select k randomly from $\{0, \ldots, m+r\}$. If $k = 0$, then there is no mutation and we let $\mathbf{v}_m \mathbf{u}_r = \mathbf{x}_m \mathbf{y}_r$. Otherwise, change the kth base of $\mathbf{x}_m \mathbf{y}_r$ equally likely into one of the three other bases and denote the mutated sequence as $\mathbf{v}_m \mathbf{u}_r$.
2. Select d randomly from $\{d_1, \ldots, d_2\}$ and generate the bases of $\mathbf{z} = z_1 \cdots z_d$ sequentially from the underlying Markov chain, initialized at $z_0 = v_m$.

We perform a simulation study on eight structural motifs selected for their high frequency of occurrences in part of the *Bacillus subtilis* DNA dataset. We consider $(d_1, d_2) = (16, 18)$ and $(5, 50)$, with length of DNA sequence $n = 100$, and a Markov chain with transition matrix

$$\begin{array}{c} \\ a \\ c \\ g \\ t \end{array} \begin{pmatrix} 0.35 & 0.16 & 0.18 & 0.31 \\ 0.33 & 0.20 & 0.15 & 0.32 \\ 0.32 & 0.22 & 0.19 & 0.27 \\ 0.25 & 0.20 & 0.19 & 0.35 \end{pmatrix}.$$

In Table 3, we compare importance sampling estimates of $P\{N \ge 1\}$ using Algorithm A with analytical p-value estimates from Robin et al. (2002) and direct Monte Carlo p-value estimates. The analytical p-value

TABLE 3. COMPARISON OF DIRECT MONTE CARLO, IMPORTANCE SAMPLING, AND ANALYTICAL ESTIMATES OF $P\{N \geq 1\}$ FOR STRUCTURED MOTIFS

d_1	d_2	x	y	*Direct MC*	*Algorithm A*	*Analytic*
16	18	*gttgaca*	*atataat*	$(2 \pm 1)\times 10^{-4}$	$(1.038 \pm 0.006)\times 10^{-4}$	1.01×10^{-4}
		gttgaca	*tataata*	0	$(9.00 \pm 0.05)\times 10^{-5}$	8.82×10^{-5}
		tgttgac	*tataata*	$(20 \pm 10)\times 10^{-5}$	$(9.39 \pm 0.05)\times 10^{-5}$	9.20×10^{-5}
		ttgaca	*ttataat*	$(9 \pm 3)\times 10^{-4}$	$(6.65 \pm 0.03)\times 10^{-4}$	6.55×10^{-4}
		ttgacaa	*tacaat*	$(4 \pm 2)\times 10^{-4}$	$(4.64 \pm 0.02)\times 10^{-4}$	4.57×10^{-4}
		ttgacaa	*tataata*	$(2 \pm 1)\times 10^{-4}$	$(1.798 \pm 0.009)\times 10^{-4}$	1.78×10^{-4}
		ttgacag	*tataat*	$(5 \pm 2)\times 10^{-4}$	$(3.62 \pm 0.02)\times 10^{-4}$	3.59×10^{-4}
		ttgacg	*tataat*	$(10\times 3)\times 10^{-4}$	$(9.90 \pm 0.06)\times 10^{-4}$	9.76×10^{-4}
		combined p-value		$(2.0 \pm 0.4)\times 10^{-3}$	$(2.96 \pm 0.03)\times 10^{-3}$	
5	50	*gttgaca*	*atataat*	$(1 \pm 0.3)\times 10^{-3}$	$(1.265 \pm 0.008)\times 10^{-3}$	
		gttgaca	*tataata*	$(0.4 \pm 0.2)\times 10^{-3}$	$(1.103 \pm 0.007)\times 10^{-3}$	
		tgttgac	*tataata*	$(1.8 \pm 0.4)\times 10^{-3}$	$(1.150 \pm 0.007)\times 10^{-3}$	
		ttgaca	*ttataat*	$(7.4 \pm 0.9)\times 10^{-3}$	$(7.88 \pm 0.05)\times 10^{-3}$	
		ttgacaa	*tacaat*	$(5.0 \pm 0.7)\times 10^{-3}$	$(5.50 \pm 0.04)\times 10^{-3}$	
		ttgacaa	*tataata*	$(1.5 \pm 0.4)\times 10^{-3}$	$(2.21 \pm 0.01)\times 10^{-3}$	
		ttgacag	*tataat*	$(3.1 \pm 0.6)\times 10^{-3}$	$(4.23 \pm 0.03)\times 10^{-3}$	
		ttgacg	*tataat*	$(0.9 \pm 0.1)\times 10^{-2}$	$(1.126 \pm 0.008)\times 10^{-2}$	
		combined p-value		$(2.7 \pm 0.2)\times 10^{-2}$	$(3.30 \pm 0.04)\times 10^{-2}$	

For both direct Monte Carlo and importance sampling, 10,000 simulation runs are executed for each entry and the results are displayed in the form $\hat{p} \pm$ standard error.

estimates are computed numerically via recursive methods with computation time that grows exponentially with $d_2 - d_1$, and are displayed only for the case $(d_1, d_2) = (16, 18)$.

We illustrate here how the importance sampling algorithms can be modified to handle more complex situations, for example, to obtain a combined p-value for all eight motifs. Consider more generally $p = P\{\max_{1\leq j\leq J}(N^{(j)} - c_j) \geq 0\}$, where $N^{(j)}$ is the total word count from the motif $\mathcal{V}^{(j)}$ and c_j is a positive integer. Let $L^{(j)}$ be the likelihood ratio when applying either Algorithm A or B with insertion of words from $\mathcal{V}^{(j)}$. For the kth simulation run, we execute the following steps.

1. Select j_k randomly from $\{1, \ldots, J\}$.
2. Generate $\mathbf{s}_0^{(k)}$ using either Algorithm A or B, with insertion of words from $\mathcal{V}^{(j)}$.

Then

$$\hat{p}_I = K^{-1}\sum_{k=1}^{K}[L^{(j_k)}(\mathbf{s}_0^{(k)})]^{-1}\left(\frac{J}{|\{j : N^{(j)}(\mathbf{s}_0^{(k)}) \geq c_j\}|}\right)\mathbf{1}_{\{N^{(j_k)}(\mathbf{s}_0^{(k)})\geq c_{j_k}\}} \tag{5.4}$$

is unbiased for p (see Appendix C). The key feature in (5.4) is the correction term $|\{j : N^{(j)}(\mathbf{s}_0^{(k)}) \geq c_j\}|$. Without this term, $\hat{p}_I$ is an unbiased estimator for the Bonferroni upper bound $\sum_{j=1}^{J} P\{N^{(j)} \geq c_j\}$. The correction term adjusts the estimator downwards when more than one thresholds c_j are exceeded.

We see from Table 3 that the variance reduction is substantial when importance sampling is used. In fact, the direct Monte Carlo estimate is often unreliable. Such savings in computation time is valuable both to the end user and also to the researcher trying to test the reliability of his or her analytical estimates on small p-values. We observe for example that the numerical estimates for $(d_1, d_2) = (16, 18)$ given in Robin et al. (2002) are quite accurate but tends to underestimate the true underlying probability.

6. DISCUSSION

The examples given here are not meant to be exhaustive but they do indicate how we can proceed in situations not covered here. For example, if we would like the order of the two words in a CRM to be arbitrary, we can include an additional permutation step in the construction of the word bank. In Section

5.2, we also showed how to simulate p-values of the maximum count over a set of word patterns. As we gain biological understanding, the models that we formulate for DNA and protein functional sites become more complex. Over the years, they have evolved from deterministic words to consensus sequences to PSWMs and then to motif modules. As probabilistic models for promoter architecture gets more complex and context specific, importance sampling methods are likely to be more widely adopted in the computation of p-values.

7. APPENDIX

A. Generating palindromes and invented repeats

We first show how words $\mathbf{v}_m$ can be generated with probability mass function

$$q(\mathbf{v}_m) = \pi(v_1)\sigma(\mathbf{v}_m)\sigma(\mathbf{v}_m^c)/\eta,$$

with $\eta = \sum_{\mathbf{v}_m} \pi(v_1)\sigma(\mathbf{v}_m)\sigma(\mathbf{v}_m^c)$ a computable normalizing constant. Apply the backward recursive relations

$$\eta_i(x) = \sum_{y\in\chi} \sigma(xy)\sigma(y^c x^c)\eta_{i+1}(y) \quad \text{for all} \quad x \in \chi \quad \text{and} \quad i = 1, \ldots, m-1, \tag{A.1}$$

initialized with $\eta_m(x) = 1$ for all x. Then $\eta = \sum_{x\in\chi} \pi(x)\eta_1(x)$. Let Q be the desired probability measure for generating $\mathbf{v}_m$ with probability mass function q. Then the Markovian property

$$\begin{aligned} Q\{v_1 = x\} &= \pi(x)\eta_1(x)/\eta, \\ Q\{v_{i+1} = y | v_i = x\} &= \sigma(xy)\sigma(y^c x^c)\eta_{i+1}(y)/\eta_i(x) \quad \text{for} \quad i = 1, \ldots, m-1, \end{aligned} \tag{A.2}$$

allows us to generate v_i sequentially via transition matrices.

To generate words $\mathbf{v}_m$ with probability mass function $q(\mathbf{v}_m) = \pi(v_1)\sigma(\mathbf{v}_m\mathbf{v}_m^c)/\eta$, let $\eta_m(x) = \sigma(xx^c)$ instead of $\eta_m(x) = 1$ and proceed with (A.1) and (A.2).

B. Generating highscoring motifs from PSWMs

Let S be the score with respect to a given PSWM W and let $\theta > 0$. We provide here a quick recursive algorithm for generating $\mathbf{v}_m$ from the probability mass function

$$q_\theta(\mathbf{v}_m) = e^{\theta S(\mathbf{v}_m)}\pi(v_1)\sigma(\mathbf{v}_m)/\Lambda(\theta), \tag{A.3}$$

with $\Lambda(\theta) = \Sigma_{\mathbf{v}m} e^{\theta S(\mathbf{v}_m)}\pi(v_1)\sigma(\mathbf{v}_m)$ a computable normalizing constant. Since $\log \Lambda(\theta)$ is convex, the solution of $\frac{d}{d\theta}[\log \Lambda(\theta)] = t$ can be found using a bijection search. We take note of the backward recursive relations

$$\begin{aligned} \Lambda_m(\theta, x) &= e^{\theta w_m(x)}, \\ \Lambda_i(\theta, x) &= e^{\theta w_i(x)} \sum_{y\in\chi} \sigma(xy)\Lambda_{i+1}(\theta, y) \quad \text{for all} \quad x \in \chi \quad \text{and} \quad i = 1, \ldots, m-1, \end{aligned} \tag{A.4}$$

from which we can compute $\Lambda(\theta) = \sum_{x\in\chi} \pi(x)\Lambda_1(\theta, x)$. Let Q denote the desired probability measure for generating $\mathbf{v}_m = v_1 \cdots v_m$ from q_θ. By (A.3) and (A.4), we can simply generate the letters v_i sequentially, using transition matrices defined by the Markovian relations

$$\begin{aligned} Q\{v_1 = x\} &= \pi(x)\Lambda_1(\theta, x)/\Lambda(\theta), \\ Q\{v_{i+1} = y | v_i = x\} &= e^{\theta w_i(x)}\sigma(xy)\Lambda_{i+1}(\theta, y)/\Lambda_i(\theta, x) \quad \text{for} \quad i = 1, \ldots, m-1. \end{aligned} \tag{A.5}$$

C. Unbiasedness of $\widehat{p}_I$ in (5.4)

We shall show here that $\widehat{p}_I$ in (5.4) is unbiased for $p = P\{\max_{1\le j\le J}(N^{(j)} - c_j) \ge 0\}$. Let $A_j = \{s_0 : N^{(j)}(\mathbf{s}_0) \ge c_j\}$ and let Q_j be a probability measure such that $L^{(j)}(\mathbf{s}_0) = Q_j(\mathbf{s}_0)/P(\mathbf{s}_0) > 0$ for any $\mathbf{s}_0 \in A_j$. Let $A = \cup_{j=1}^J A_j$. Then with the convention $0/0 = 0$,

$$J^{-1}\sum_{j=1}^{J}E_{Q_j}\left\{[L^{(j)}(\mathbf{s}_0)]^{-1}\left(\frac{J}{|\{\ell:\mathbf{s}_0\in A_\ell\}|}\right)\mathbf{1}_{\{\mathbf{s}_0\in A_j\}}\right\}=E\left(\frac{\sum_{j=1}^{J}\mathbf{1}_{\{\mathbf{s}_0\in A_j\}}}{|\{\ell:\mathbf{s}_0\in A_\ell\}|}\right)=P\{\mathbf{s}_0\in A\},$$

and hence $\widehat{p}_I$ is indeed unbiased.

D. Asymptotic optimality

To estimate $p:=P\{N(\mathbf{s})\geq c\}$ using direct Monte Carlo, simply generate K independent copies of $\mathbf{s}$, denoted by $\mathbf{s}^{(1)},\ldots,\mathbf{s}^{(K)}$, under the original probability measure P, and let

$$\widehat{p}_{\mathrm{D}}=K^{-1}\sum_{k=1}^{K}\mathbf{1}_{\{N(\mathbf{s}^{(k)})\geq c\}}.$$

To simulate p using importance sampling, we need to first select a probability measure $Q\neq P$ for generating $\mathbf{s}^{(1)},\ldots,\mathbf{s}^{(K)}$. The estimate of p is then

$$\widehat{p}_{\mathrm{I}}:=K^{-1}\sum_{k=1}^{K}L^{-1}(\mathbf{s}^{(k)})\mathbf{1}_{\{N(s^{(k)})\geq c\}},\quad\text{where}\quad L(\mathbf{s})=Q(\mathbf{s})/P(\mathbf{s}).$$

We require $Q(\mathbf{s})>0$ whenever $N(\mathbf{s})\geq c$, so as to ensure that $\widehat{p}_I$ is unbiased for p.

The relative error (RE) of a Monte Carlo estimator $\widehat{p}=\widehat{p}_{\mathrm{D}}$ or $\widehat{p}_I$, is given by $\sqrt{\mathrm{Var}(\widehat{p})}/p$. We say that $\widehat{p}$ is asymptotically optimal if for any $\epsilon>0$, we can satisfy RE $\leq\epsilon$ with $\log K=o(|\log p|)$ as $p\to 0$ (Sadowsky and Bucklew, 1990; Dupuis and Wang, 2005). Since $\mathrm{RE}(\widehat{p}_{\mathrm{D}})=\sqrt{(1-p)/(Kp)}$, direct Monte Carlo is not asymptotically optimal. The question we would like to answer here is: Under what conditions are Algorithms A and B asymptotically optimal?

The examples described in Sections 3–5 involve word families that can be characterized as $\mathcal{V}_m$. We may also include an additional subscript m to a previously defined quantity to highlight its dependence on m, for example p_m, q_m, β_m and n_m. We say that x_m and y_m have similar logarithmic value relative to m, and write $x_m\simeq y_m$, if $|\log x_m-\log y_m|=o(m)$ as $m\to\infty$. It is not hard to see that if $x_m\simeq y_m$ and $y_m\simeq z_m$, then $x_m\simeq z_m$. In Algorithm A, it is assumed implicitly that $n_m\geq\ell_{\max}(=\ell_{\max,m}):=\max_{\mathbf{v}\in\mathcal{V}_m}\ell(\mathbf{v})$ and we shall also assume $n_m\geq c\ell_{\max}$ when using Algorithm B. To fix the situation, let $\rho_i=c/n_m$ for all i in Algorithm B. Let $\beta_{\min}(=\beta_{\min,m})=\min_{\mathbf{v}\in\mathcal{V}_m}\beta_m(\mathbf{v})$, $\beta_{\max}(=\beta_{\max,m})=\max_{\mathbf{v}\in\mathcal{V}_m}\beta_m(\mathbf{v})$, $\sigma_{\min}=\min_{x,y\in\chi}\sigma(xy)(>0)$, $\sigma_{\max}=\max_{x,y\in\chi}\sigma(xy)(<1)$ and $\pi_{\min}=\min_{x\in\chi}\pi(x)(\geq\sigma_{\min})$. Let $\lfloor\cdot\rfloor$ denote the greatest integer function, P_x denote probability conditioned on $s_1=x$ or $v_1=x$ and P_π denote probability conditioned on s_1 or v_1 following the stationary distribution.

In the following lemma, we provide conditions for asymptotic optimality and check them in Appendices D.1–D.3 for the word families discussed in Sections 3–5.

Lemma 1. *If* $\log n_m\simeq 1$ *and*

$$p_m\leq\alpha^m\ \textit{for some}\ 0<\alpha<1, \tag{A.6}$$

$$\ell_{\max}\simeq 1, \tag{A.7}$$

$$\beta_{\min}\simeq\Big(\sum_{\mathbf{v}\in\mathcal{V}_m}\sigma(\mathbf{v})\Big)^{-1}, \tag{A.8}$$

then both Algorithms A and B are asymptotically optimal.

Proof. Let $r_m=\sum_{x\in\chi}P_x\{\mathbf{s}_\ell\in\mathcal{V}_m\ \text{ for some }\ell\geq 1\}$. Since $\sum_{\mathbf{v}\in\mathcal{V}_m}\sigma(\mathbf{v})\geq r_m\geq\ell_{\max}^{-1}\sum_{\mathbf{v}\in\mathcal{V}_m}\sigma(\mathbf{v})$, by (A.7) and (A.8),

$$r_m\simeq\sum_{\mathbf{v}\in\mathcal{V}_m}\sigma(\mathbf{v})\simeq\beta_{\min}^{-1}. \tag{A.9}$$

By (6.1), $|\log p_m|\geq m|\log\alpha|$ for all large m and hence it suffices for us to show $K_m\simeq 1$.

If $n_m\simeq 1$, then by (A.9) and the inequalities $\binom{n_m}{c}r_m^c\geq p_m\geq(\sigma_{\min}r_m)^c$,

$$(n_m\beta_{\min}^{-1})^c \simeq (n_m r_m)^c \simeq p_m. \tag{A.10}$$

Consider next the case $n_m/\ell_{\max} \to \infty$. Since $\log n_m \simeq 1$, there exists integers ξ_m such that $\xi_m \simeq 1$, $\xi_m = o(n_m)$ and $\log\ n_m = o(\xi_m)$. Let $\kappa_m = \lfloor n_m/(\ell_{\max} + \xi_m) \rfloor$ and $g_m = P_\pi\{s_\ell \in \mathcal{V}_m \text{ for some } \ell \geq 1\}$. By (A.6), $\alpha^m \geq p_m \geq (g_m\sigma_{\min})^c$ and hence $g_m \to 0$. Since the underlying Markov chain is uniformly ergodic,

$$\sup_{x,\, y \in \chi} |P_x\{s_{k+1} = y\} - \pi(y)| \leq \eta^k \quad \text{for some } 0 < \eta < 1. \tag{A.11}$$

By considering the sub-cases of at least c words $\mathbf{v} \in \mathcal{V}_m$ starting at positions 1, $(\ell_{\max} + \xi_m) + 1, \ldots, (\kappa_m - 1)(\ell_{\max} + \xi_m) + 1$, it follows from (A.11) that

$$p_m \geq 1 - \sum_{j=0}^{c-1} \binom{\kappa_m}{j} g_m^j (1 - g_m)^{\kappa_m - j} - (\kappa_m - 1)\eta^{\xi_m} = 1 - (1 + o(1)) \sum_{j=0}^{c-1} \frac{(\kappa_m g_m)^j}{j!} e^{-\kappa_m g_m} - o(1).$$

By (A.6), $\kappa_m g_m \to 0$ and this implies $\kappa_m r_m \to 0$. Since $(\ell_{\max} + \xi_m) \simeq 1$, it follows that $\kappa_m \simeq n_m$ and hence by the inequalities

$$\binom{n_m}{c} r_m^c \geq p_m \geq \binom{\kappa_m}{c} (\sigma_{\min} r_m)^c (1 - r_m)^{\kappa_m - c},$$

(A.10) again holds. By using a subsequence argument if necessary, it follows that (A.10) holds as long as $\log\ n_m \simeq 1$.

For Algorithm A, by (2.3) and (2.4),

$$\mathrm{RE}(\widehat{p}_{\mathrm{I}}) \leq p_m^{-1} K_m^{-1/2} \sup_{\mathbf{s}} L^{-1}(\mathbf{s}) \mathbf{1}_{\{N(\mathbf{s}) \geq 1\}} \leq p_m^{-1} K_m^{-1/2} n_m \sigma_{\max} \beta_{\min}^{-1}$$

and the desired relation $K_m \simeq 1$ follows from (A.10) with $c = 1$.

For Algorithm B, it follows from (2.6) that if $N(\mathbf{s}) \geq c$, then $L(\mathbf{s}) \geq (1 - c/n_m)^{n_m} [c\beta_{\min}/(n_m\sigma_{\max})]^c$ and hence by (2.4),

$$\mathrm{RE}(\widehat{p}_{\mathrm{I}}) \leq p_m^{-1} K_m^{-1/2} \sup_{\mathbf{s}} L^{-1}(\mathbf{s}) \mathbf{1}_{\{N(\mathbf{s}) \geq c\}} \leq (1 + o(1)) p_m^{-1} K_m^{-1/2} [e n_m \sigma_{\max}/(c\beta_{\min})]^c,$$

and again $K_m \simeq 1$ follows from (A.10). ■

D.1. Inverted repeats

Consider the word family (3.1) with $d_2 \simeq 1$. Then (A.7) holds. Since $p_m \leq (d_2 - d_1) n_m \sigma_{\max}^{2m-1}$, (A.6) holds when $n_m = O(\gamma^m)$ for some $\gamma < \sigma_{\max}^{-2}$. It remains to check (A.8). Since $\sum_{\mathbf{v} \in \mathcal{V}_m} q_m(\mathbf{v}) = \sum_{\mathbf{v} \in \mathcal{V}_m} \beta_m(\mathbf{v})\sigma(\mathbf{v}) = 1$,

$$\beta_{\min} \leq \Big(\sum_{\mathbf{v} \in \mathcal{V}_m} \sigma(\mathbf{v}) \Big)^{-1} \leq \beta_{\max}. \tag{A.12}$$

Let $\mathbf{u}_m$ be generated with probability proportional to $\pi(u_1)\sigma(\mathbf{u}_m)\sigma(\mathbf{u}_m^c)$ when creating the word bank $\mathcal{V}_m$. Then there exists a constant $C > 0$ such that

$$\beta_m(\mathbf{u}_m \mathbf{z} \mathbf{u}_m^c) = C\pi(u_1)\sigma(\mathbf{u}_m \mathbf{z})\sigma(\mathbf{u}_m^c)/\sigma(\mathbf{u}_m \mathbf{z} \mathbf{u}_m^c) = C\pi(u_1)/\sigma(z_d u_1^c).$$

Hence $\beta_{\min} \simeq \beta_{\max}$ and (A.8) follows form (A.12).

D.2. Word patterns derived from PSWMs

For the word family (4.2), condition (A.7) is always satisfied. Let the entries of the PSWM be non-negative integers and assume that the column totals are fixed at some $C > 0$. It follows from large deviations theory (Dembo and Zeitouni, 1998) that if $t(= t_m) \geq E_\pi S(\mathbf{v}) + \zeta m$ for some $\zeta > 0$, then

$$P_\pi\{S(\mathbf{v}) \geq t\} = O(\lambda^m) \quad \text{for some} \quad 0 < \lambda < 1. \tag{A.13}$$

Since $p_m \leq n_m P_\pi\{S(\mathbf{v}) \geq t\}$, (A.6) holds if $n_m = O(\gamma^m)$ for some $\gamma < \lambda^{-1}$.

To simplify the analysis in checking (A.8), select the tilting parameter $\theta(=\theta_m)$ to be the root of $E_{q\theta}[S(\mathbf{v})] = t + \delta_m$ for some positive $\delta_m = o(m)$ satisfying $m^{-1/2}\delta_m \to \infty$ as $m \to \infty$, instead of the root of $E_{q\theta}[S(\mathbf{v})] = t$, as suggested in the statement containing (4.5). The implicit assumption is that $\sum_{i=1}^{m}\{\max_{v\in\chi} w_i(v)\} > t + \delta_m$ for all m. Since the entries of the transition matrices derived in Appendix B are uniformly bounded away from zero, it follows from a coupling argument that $\mathrm{Cov}_{q\theta}(w_i(v_i), w_j(v_j)) = O(\tau^{|i-j|})$ for some $0 < \tau < 1$ and hence $\mathrm{Var}_{q\theta}(S(\mathbf{v})) = O(m)$. By (4.3) and Chebyshev's inequality,

$$e^{\theta(t+2\delta_m)} \sum_{\mathbf{v}\in\mathcal{V}_m} \sigma(\mathbf{v})/\Lambda(\theta) \geq \sum_{\mathbf{v}:|S(\mathbf{v})-t-\delta_m|\leq\delta_m} q_\theta(\mathbf{v}) \geq 1 - \delta_m^{-2}\mathrm{Var}_\theta(S(\mathbf{v})) > 0 \tag{A.14}$$

for all large m. Since $\xi > 1$ in (4.4), $\beta_{\min} = \min_{\mathbf{v}\in\mathcal{V}_m} q_m(\mathbf{v})/\sigma(\mathbf{v}) > e^{\theta t}\pi_{\min}/\Lambda(\theta)$ and (A.8) follows from (A.12) and (A.14).

D.3. Co-occurrences of motifs

Consider the word family (5.1) with (r/m) bounded away from zero and infinity and $d_2 \simeq 1$. We check that (A.7) holds. If $t_1 \geq ES_1(\mathbf{v}) + \zeta m$ for some $\zeta > 0$, then (A.13) holds with S replaced by S_1, t replaced by t_1 and hence (A.6) holds if $n_m = O(\gamma^m)$ for some $\gamma < \lambda^{-1}$.

Let θ_j be the root of $E_{\theta j}[S_j(\mathbf{v})] = t_j + \delta_m$ for some positive $\delta_m = o(m)$ with $m^{1/2}\delta_m \to \infty$, $j = 1$ and 2, assuming that $\sum_{i=1}^{m_j}\{\max_{v\in\chi} w_i^{(j)}(v)\} > t_j + \delta_m$, where $m_1 = m$ and $m_2 = r$. Let $\mathcal{V}_m^{(1)} = \{\mathbf{v}_m : S_1(\mathbf{v}_m) \geq t_1\}$, $\mathcal{V}_r^{(2)} = \{\mathbf{u}_r : S_2(\mathbf{u}_r) \geq t_2\}$ and let $\Lambda^{(1)}(\theta_1)$, $\Lambda^{(2)}(\theta_2)$ be their respective normalizing constants, see (4.3). By the arguments in (A.14),

$$\sum_{\mathbf{v}\in\mathcal{V}_m} \sigma(\mathbf{v}) \geq \sigma_{\min}\Big(\sum_{\mathbf{v}\in\mathcal{V}_m^{(1)}} \sigma(\mathbf{v})\Big)\Big(\sum_{\mathbf{u}\in\mathcal{V}_r^{(2)}} \sigma(\mathbf{u})\Big) = e^{-\theta_1 t_1 - \theta_2 t_2 + o(m)}\Lambda^{(1)}(\theta_1)\Lambda^{(2)}(\theta_2).$$

By (5.2), $\beta_{\min} \geq e^{\theta_1 t_1 + \theta_2 t_2} d_2^{-1}\pi_{\min}^2/\{\Lambda^{(1)}(\theta_1)\Lambda^{(2)}(\theta_2)\}$ and hence (A.8) follows from (A.12).

ACKNOWLEDGMENTS

This research was partially supported by the National University of Singapore (grants C-389-000-010-101 and R-155-062-112).

DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Chan, H.P., and Zhang, N.R. 2007. Scan statistics with weighted observations. *J. Am. Statist. Assoc.* 102, 595–602.

Chiang, D.Y., Moses, A.M., Kellis, M., et al. 2003. Phylogenetically and spatially conserved word pairs associated with gene-expression changes in yeasts. *Genome Biol.* 4, R43.

Cottrell, M., Fort, J.C., and Malgouyres, G. 1983. Large deviations and rare events in the study of stochastic algorithms. *IEEE Trans. Automat. Contr.* 28, 907–920.

Dembo, A., and Zeitouni, O. 1998. *Large Deviations: Techniques and Applications*. Springer, New York.

Do, K.A., and Hall, P. 1992. Distribution estimating using concomitant of order statistics, with applications to Monte Carlo simulation for the bootstrap. *J.R. Statist. Soc. B* 54, 595–607.

Dupuis, P., and Wang, H. 2005. Dynamic importance sampling for uniformly recurrent Markov chains. *Ann. Appl. Probabil.* 15, 1–38.

Fuh, C.D., and Hu, I. 2004. Efficient importance sampling for events of moderate deviations with applications. *Biometrika* 91, 471–490.

Gusfield, D. 1997. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press, London.

Huang, H., Kao, M., Zhou, X., et al. 2004. Determination of local statistical significance of patterns in Markov sequences with applications to promoter element identification. *J. Comput. Biol.* 11, 1–14.

Johns, M.V. 1988. Importance sampling for bootstrap confidence intervals. *J. Am. Statist. Assoc.* 83, 709–714.

Lai, T.L., and Shan, J.Z. 1999. Efficient recursive algorithms for detection of abrupt changes insignals and control systems. *IEEE Trans. Automat. Contr.* 44, 952–966.

Leung M.Y., Choi K.P., Xia A., et al. 2005. Nonrandom clusters of palindromes in herpesvirus genomes. *J. Comput. Biol.* 12, 331–354.

Leung M.Y., Schachtel G.A., and Yu H.S. 1994. Scan statistics and DNA sequence analysis: the search for an origin of replication in a virus. *Nonlinear World* 1, 445–471.

Masse, M.J.O., Karlin, S., Schachtel, G.A., et al. 1992. Human cytomegalovirus origin of DNA replication (oriLyt) resides within a highly complex repetitive region. *Proc. Natl Acad. Sci. USA* 89, 5246–5250.

Mitrophanov, A.Y., and Borodovsky, M. 2006. Statistical significance in biological sequence analysis. *Briefings Bioinform.* 7, 2–24.

Pape, U., Rahmann, S., Sun, F., et al. 2008. Compound Poisson approximation of the number of occurrences of a position frequency matrix (PFM) on both strands. *J. Comput. Biol.* 15, 547–564.

Prum, B., Rodolphe, F., and de Turckheim, E. 1995. Finding words with unexpected frequencies in deoxyribonucleic acid sequences. *J.R. Statist. Soc. B* 57, 205–220.

Régnier, M. 2000. A unified approach to word occurrence probabilities. *Dis. Appl. Math.* 104, 259–280.

Reinert, G., Schbath, S., and Waterman, M. 2000. Probabilistic and statistical properties of words: an overview. *J. Comput. Biol.* 7, 1–46.

Robin, S., Daudin, J., Richard, H., et al. 2002. Occurrence probability of structured motifs in random sequences. *J. Comput. Biol.* 9, 761–773.

Sadowsky, J.S., and Bucklew, J.A. 1990. On large deviations theory and asymptotically efficient Monte Carlo estimation. *IEEE Trans. Inform. Theory* 36, 579–588.

Siegmund, D. 1976. Importance sampling in the Monte Carlo study of sequential test. *Ann. Statist.* 4, 673–684.

Spellman P.T., Sherlock, G., Zhang, M.Q., et al. 1998. Comprehensive identification of cell cycle–regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell* 9, 3273–3297.

Zhang, N.R., Wildermuth, M.C., and Speed, T.P. 2008. Transcription factor binding site prediction with multivariate gene expression data. *Ann. Appl. Statist.* 2, 332–365.

Zhou, Q., and Wong, W. 2004. CisModule: de novo discovery of cis-regulatory modules by hierarchical mixture modeling. *Proc. Natl Acad. Sci. USA* 101, 12114–112119.

Zhu, J., and Zhang, M.Q. 1999. SCPD: a promoter database of the yeast *Saccharomyces cerevisiae*. *Bioinformatics* 15, 607–611.

Address correspondence to:
Dr. Nancy Ruonan Zhang
Department of Statistics
Stanford University
Stanford, CA 94305-4065

E-mail: nzhang@stanford.edu