Scan statistic distribution through slack equations

Donald E. K. Martin
North Carolina State University, Statistics Department, 4272 SAS Hall, 2301 Stinson Drive, Raleigh NC 27695-8203

Abstract
Scan statistics are used in many areas of applied probability and statistics to study local clumping of patterns. Testing based on a scan statistic requires tail probabilities. Whereas the distribution of various scan statistics has been studied extensively, most of the results are approximations, due to the difficulties associated with the computation. Results have been given to compute exact p-values for the statistic over a binary sequence that is independent or first-order Markovian. However, in many practical applications, the variables under study take on multiple values, and/or a model with higher-order dependence provides a better fit. The present paper fills this gap by obtaining the distribution of the univariate scan statistic for multi-state trials that are Markovian of a general order of dependence. A deterministic finite automaton is developed to index the computation, and a matrix corresponding to automaton transitions is used to update probabilities. Examples are given to illustrate the algorithm.

Key Words: clustering of patterns, deterministic finite automaton, higher-order Markovian trials, multi-state trials, one-dimensional scan statistic

1. Introduction

For a sequence $\mathbf{X} = X_1, \ldots, X_n$ of random variables taking values in a finite discrete space, the discrete *scan statistic* is typically defined by

$$S(w) = \max_i \left\{ \sum\nolimits_{j=i-w+1}^{i} X_j \text{ for } i = w, w+1, \ldots, n \right\}. \tag{1}$$

The statistic is frequently used to detect local clustering of patterns in $\mathbf{X}$. As examples, Wagner (1999) used the statistic to detect clusters of transcription factor binding sites in DNA nucleotides, Hoh and Ott (2000) to search for new genes, and Sun et al. (2006) to identify choromosomal regions that are associated with disease, DNA copy number variations, and other genome-based measurements.

Tail probabilities of $S(w)$ are needed for statistical tests. However, most of the results in this area are approximations and bounds on the probabilities [see, e.g. Naus (1982), Glaz (1989), Loader (1991), and Chen (1998)]. Results for the exact distribution of $S(w)$ are relatively scarce, even in the case of independent binary trials, due to the difficulty in computing the distribution. Naus (1974) obtained an exact combinatorial result for i.i.d. Bernoulli trials that by its nature has limited applicability. Fu (2001) computed p-values for the statistic that apply to independent and first-order Markovian binary trials, and displayed the statistic's entire distribution for window sizes as large as

$w = 5$. Ebneshahrashoob et al. (2005) set up a system of conditional probability generating functions and used a matrix corresponding to the system to extend the latter results to larger window sizes. However, to date there have been no exact results for multi-state trials or higher-order Markovian sequences, both of which can be very useful in modeling.

An application where multi-state trials arose was in locating significant clusters of charges in protein sequences (Karlin et al. 1989). In the study, arginine, lysine and histidine were represented by “1” (a positive charge), aspartic and glutamic acid by “-1” (a negative charge), and all other residues by “0,” a neutral charge. An application where a higher-order Markov model is particularly useful is to model the “background noise” when searching for exceptional patterns in biological sequences [Robin et al. (2005)].

We allow **X** to be a sequence of multi-state trials that is higher-order Markovian, and compute the exact distribution of the scan statistic (1). The computation is carried out by forming a deterministic finite automaton with final states that indicate that $S(w) \geq s$ for each fixed value of s, setting up an initial distribution and transition probability matrix for automaton state transitions, and using vector-matrix updates of probabilities. Thus the computation has some similarities to the automaton-based approach to compute distributions of patterns of Ribeca and Ranieri (2008), as well as the finite Markov chain embedding approach used in Fu (2001), with deviations to improve computational efficiency that will be mentioned in what follows.

The paper is organized as follows. The next section contains details of the algorithm. Section 3 gives examples to illustrate the algorithm and its use, and the final section is a summary.

2. Computation of the Distribution of the Scan Statistic

Let **X** be a stationary mth-order Markovian sequence with realized values denoted by $x_1, \ldots, x_n$, where $x_i \in \Sigma = \{\sigma_0, \sigma_1, \ldots, \sigma_{k-1}\}$ with $\sigma_0 < \sigma_1 < \ldots < \sigma_{k-1}$. For simplicity we focus on the case where $\sigma_{j+1} = \sigma_j + \delta$ for some integer δ and $j = 0, 1, \ldots, k-2$, though the general case can be handled in a similar manner. In the linear case that is assumed, we can without loss of generality use $\Sigma = \{0, 1, \ldots, k-1\}$ so that the scan statistic takes values in $\Phi = \{\nu_0, \nu_1, \ldots, \nu_{w(k-1)}\} = \{0, 1, \ldots, w(k-1)\}$, since if $\Sigma' = \{\sigma'_0, \sigma'_1, \ldots, \sigma'_{k-1}\}$ with $\sigma'_j = \sigma'_0 + \delta j$ for $j = 0, 1, \ldots, k-1$ and some fixed δ, the elements of $\Phi' = \{\nu'_0, \nu'_1, \ldots, \nu'_{w(k-1)}\}$ correspond in a one-to-one fashion to Φ through $\nu'_j = w\sigma'_0 + \delta s_j$.

We compute probabilities $P(S(w) \geq s)$ that give p-values for a given initial distribution π over m-tuples $\tilde{x}_m \in \Sigma^m$ and transition matrix T. Here $\tilde{x}_t \equiv (x_{t-m+1}, \ldots, x_t)$, $t = m, m+1, \ldots, n$ and T gives transition probabilities for transitions of $\tilde{x}_t$ to $\tilde{x}_{t+1}$, or equivalently probabilities $P(x_{t+1} | \tilde{x}_t)$. The basic steps of the algorithm are:

- Set up a deterministic finite automaton [DFA, Hopcroft et al. (2001)] with states that index progress into terminal strings that indicate that $S(w) \geq s$.
- Use π to set up an initial distribution $\tilde{\pi}$ over automaton states, and T to set up a transaction probability matrix Ω for transitions of those states.
- Carry out matrix-vector updates to compute the desired probability.

Next we give more details on the computation algorithm.

2.1 Setup of automaton states

For a fixed s, let Δ_s be the set of automaton states that is needed to compute $P\left(S(w)\geq s\right)$, and let $\Delta\equiv\bigcup_{s=2}^{w(k-1)}\Delta_s$. For a state $d=d_1\cdots d_j\in\Delta$ of length j, define the *window sum* (denoted by $win\left(d\right)$) to be $\sum_{i=1}^{j}d_i$.

The shortest strings in Δ_s are those of $\Sigma_s^m\subset\Sigma^m$, the m-tuples $\tilde{x}_m$ for which $win\left(\tilde{x}_m\right)<s$. This implies that all transient automaton states carry the last m values of the $\mathbf{X}$ sequence. Σ_s^m is required in Δ_s due to the Markovian assumption.

We then add to Δ_s the following strings of lengths $j=m+1,\ldots,w-1$:

$$D_{s,j}=\left\{d\,\middle|\,d_1>0;\ \max\left[1,s-\left(k-1\right)\left(w-j\right)\right]\leq win\left(d\right)\leq\min\left[j(k-1),s-1\right]\right\}.$$

The inequality

$$s-\left(k-1\right)\left(w-j\right)\leq win\left(d\right)\tag{2}$$

guarantees that from every state of $D_s\equiv\bigcup_j D_{s,j}$, a sum $d_1+d_2+\cdots+d_w\geq s$ is possible. On the other hand, the inequality

$$win\left(d\right)\leq s-1\tag{3}$$

ensures that all states of D_s are transient. We add the absorbing state A_s to Δ_s to correspond to terminal DFA strings d with $win\left(d\right)\geq s$. Then $\Delta_s=\Sigma_s^m\cup\left(\bigcup_{j=m+1}^{w-1}D_{s,j}\right)\cup A_s$, a disjoint union.

No string d of length w is needed because either such strings have $win\left(d\right)\geq s$ (and thus are represented by the absorbing state), or their window sums can't possibly reach s because their length is already w.

The states of D_s may be set up using what we call *slack* variables $\varsigma_{1,j}$ and $\varsigma_{2,j}$ that respectively indicate when the inequalities (2) and (3) no longer are satisfied. The slack variables for string $x=x_1,\ldots,x_j$ are

$$\begin{aligned}\varsigma_{1,j}&=w\left(k-1\right)-s-\left[j\left(k-1\right)-win\left(x\right)\right]\\&=\left(w-j\right)\left(k-1\right)-\left(s-win\left(x\right)\right);\end{aligned}\tag{4}$$

$$\varsigma_{2,j}=s-1-win\left(x\right).\tag{5}$$

Slack $\varsigma_{1,j}$ decreases from the initial (before any symbols are observed) "leeway" $w\left(k-1\right)-s$ by $k-1-x_i$ for each symbol $x_i<k-1$. Its negative value indicates that the string can no longer reach s, and thus isn't needed in the state space. On the other hand, $\varsigma_{2,j}$ decreases from the initial value $s-1$ by x_i. If $\varsigma_{2,j}<0$, then $win\left(x\right)\geq s$. An m-

tuple of Σ^m not satisfying (3) for $j = m$ (i.e. with $\varsigma_{2,m} < 0$) is not contained in Σ_s^m, and strings of length $j > m$ are only in D_s if $\varsigma_{1,j} \geq 0$ and $\varsigma_{2,j} \geq 0$.

The number of strings in Δ is $|\Delta| = k^m + (k-1)\left(k^m + \cdots + k^{w-2}\right) = k^{w-1}$. This shouldn't be surprising since strings of Δ correspond in a one-to-one fashion to the $(w-1)$-tuples Σ^{w-1} by deleting all leading zeroes from the latter strings (the one exception being the m-tuple consisting of all zeroes, which corresponds to the $(w-1)$-tuple consisting of all zeroes). An advantage of computing p-values is that the size of each $|\Delta_s|$ is typically much less than k^{w-1}.

In the Appendix we derive a combinatorial formula for the number of states $|\Delta_s|$ of Δ_s. Using those results, Table 1 and 2 give $|\Delta_s|$ for $m = 1$, $\Sigma = \{0,1,2\}$, and various values of w. This information is useful for dimensioning vectors when programming the algorithm. Note that $|\Delta_s|$ is maximized for $s = \lfloor \xi \rfloor$, where $\xi \equiv \frac{w(k-1)}{2} + 1$. Also, we observe symmetry in $|\Delta_s|$ (actually in $|D_s|$) for different values of s. This symmetry is explored next. We first define what we mean by "complementary" strings $x^{(a)}$ and $x^{(b)}$ and corresponding complementary values $s^{(a)}$ and $s^{(b)}$.

Table 1. State space size $|\Delta_s|$, $s = 2,3,\ldots,2w$ for $\Sigma = \{0,1,2\}$, $k = 3$, $m = 1$, and various values of w.

s	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$	$w=9$	$w=10$	$w=11$	$w=12$
2	5	6	7	8	9	10	11	12	13
3	11	16	22	29	37	46	56	67	79
4	17	31	51	78	113	157	211	276	353
5	20	46	91	162	267	415	616	881	1222
6	17	52	127	267	505	883	1453	2278	3433
7	11	46	142	358	785	1555	2851	4918	8075
8	6	31	127	394	1017	2305	4741	9043	16237
9		16	91	358	1108	2908	6766	14356	28315
10		7	51	267	1017	3140	8351	19856	43253
11			22	162	785	2908	8954	24069	58279
12			8	78	505	2305	8351	25654	69577
13				29	267	1555	6766	24069	73790
14				9	113	883	4741	19856	69577
15					37	415	2851	14356	58279
16					10	157	1453	9043	43253
17						46	616	4918	28315
18						11	211	2278	16237
19							56	881	8075
20							12	276	3433
21								67	1222
22								13	353
23									79
24									14

Table 2. $\left|\Delta_s\right|$ for $w=13$, 14 and 15, $s=\xi=w+1$, $\Sigma=\{0,1,2\}$ $(k=3)$ and $m=1$.

s	$w=13$	$w=14$	$w=15$
14	212942		
15		616228	
16			1787608

Definition 2.1. Strings $x^{(a)} \equiv \left(x_1^{(a)}, x_2^{(a)}, \ldots, x_j^{(a)}\right)$ and $x^{(b)} \equiv \left(x_1^{(b)}, x_2^{(b)}, \ldots, x_j^{(b)}\right)$ are called complementary if $x_1^{(a)} + x_1^{(b)} = k$ and $x_i^{(a)} + x_i^{(b)} = k-1$ for any $i \in \{2, \ldots, j\}$ **.**

Definition 2.2. Complementary integer values $s^{(a)}$ and $s^{(b)}$ for the problems $P\left(S(w) \geq s^{(r)}\right)$, $r=a,b$ are such that $s^{(a)} = \xi - q$ and $s^{(b)} = \xi + q$ for some $q < \xi$.

Theorem 2.2. Consider complementary strings $x^{(a)}$ and $x^{(b)}$ of arbitrary length $j \in \{m, m+1, \ldots, w-1\}$ and complementary values $s^{(a)}$ and $s^{(b)}$ respectively corresponding to $x^{(a)}$ and $x^{(b)}$. If $\left(\varsigma_{1,j}^{(r)}, \varsigma_{2,j}^{(r)}\right)$ denotes the value of $\left(\varsigma_{1,j}, \varsigma_{2,j}\right)$ for string $x^{(r)}$, $r=a,b$, then $\left(\varsigma_{1,j}^{(a)}, \varsigma_{2,j}^{(a)}\right) = \left(\varsigma_{2,j}^{(b)}, \varsigma_{1,j}^{(b)}\right)$.

Proof. First note that by definition, $\sum_{i=1}^{j} x_i^{(a)} + \sum_{i=1}^{j} x_i^{(b)} = kj - (j-1)$, $m \leq j \leq w-1$. We have from (4) and (5),

$$\varsigma_{1,j}^{(a)} = (w-j)(k-1) - \left(s^{(a)} - \sum_{i=1}^{j} x_i^{(a)}\right),$$

$$= w(k-1) - j(k-1) - \left[\left(\frac{w(k-1)}{2}\right) + 1 - q\right] + kj - \left(j - 1 + \sum_{i=1}^{j} x_i^{(b)}\right)$$

$$= \xi + q - 1 - \sum_{i=1}^{j} x_i^{(b)} = \varsigma_{2,j}^{(b)}.$$

Also,

$$\varsigma_{2,j}^{(a)} = s^{(a)} - 1 - \sum_{i=1}^{j} x_i^{(a)},$$

$$= \left(\frac{w(k-1)}{2} + 1 - q\right) - 1 - \left(kj - (j-1) - \sum_{i=1}^{j} x_i^{(b)}\right)$$

$$= w(k-1) - j(k-1) - \frac{w(k-1)}{2} - 1 - q + \sum_{i=1}^{j} x_i^{(b)}$$

$$= (w-j)(k-1) - \left(\xi + q - \sum_{i=1}^{j} x_i^{(b)}\right) = \varsigma_{1,j}^{(b)}.$$

■

Since $\left(\varsigma_{1,j}^{(a)}, \varsigma_{2,j}^{(a)}\right) = \left(\varsigma_{2,j}^{(b)}, \varsigma_{1,j}^{(b)}\right)$ and the states of D_s are those that have non-negative slack variables, we have the following corollary.

Corollary 1. Let $s^{(a)}$ and $s^{(b)}$ be complementary values. Then for every state in the set $D_{s^{(a)}}$ for computing $\Pr\left(S(w) \ge s^{(a)}\right)$, there is a corresponding state in $D_{s^{(b)}}$ for the problem of computing $\Pr\left(S(w) \ge s^{(b)}\right)$. This implies that $\left|D_{s^{(a)}}\right| = \left|D_{s^{(b)}}\right|$.

This discussion leads to the following method to obtain state spaces when the complete distribution of $S(w)$ is needed.

- First form the states of $D_{s^{(b)}}$ for $s^{(b)} = \left\lfloor \xi + \frac{1}{2} \right\rfloor, \ldots, w(k-1)$ using slack variables as filters, as described above.
- Form states $x^{(a)} = \left(x_1^{(a)}, x_2^{(a)}, \ldots, x_j^{(a)}\right) \in D_{s^{(a)}}$ for $s^{(a)} = \xi - 1, \xi - 2, \ldots, 2$ from states $x^{(b)} = \left(x_1^{(b)}, x_2^{(b)}, \ldots, x_j^{(b)}\right) \in D_{s^{(b)}}$ formed above using the symmetric of states for complementary values $s^{(a)} = \xi - q$ and $s^{(b)} = \xi + q$. This is carried out by setting $x^{(a)} = \left(k - x_1^{(b)}, k-1-x_2^{(b)}, k-1-x_3^{(b)}, \ldots, k-1-x_j^{(b)}\right)$.
- Add in m-tuples $\tilde{x}_m$ with $win(\tilde{x}_m) < s$ for the various values of s.

If $s=1$, $\Pr\left(S(w) \ge s\right) = 1 - \Pr\left(S(w)=0\right) = 1 - \tilde{\pi}\left(\overbrace{0,0,\cdots,0}^{m}\right)\left[p\left(0 \middle| \overbrace{0,0,\cdots,0}^{m}\right)\right]^{n-m}$.

2.2 Setup of initial distribution and transition matrix for automaton states

To set up the initial distribution $\tilde{\pi}$ over automaton states, set $\tilde{\pi}(\tilde{x}_m) = \pi(\tilde{x}_m)$, $\tilde{x}_m \in \Sigma_s^m$, and $\tilde{\pi}(A_s) = 1 - \varsigma$, where ς is the sum of probabilities for m-tuples $\tilde{x}_m \in \Sigma_s^m$. $\tilde{\pi}(d) = 0$ for all other states $d \in \Delta_s$.

For state $d \in \Delta_s$ of length j and symbol $x \in \Sigma$, if $\varsigma_{2,j} - x < 0$, then d transitions to A_s on symbol x. If $\varsigma_{2,j} - x \ge 0$ then $d \to d'$, where d' is the longest suffix (ending) of dx that is a state of Δ_s. (Here dx denotes the concatenation of x to the right of d). The transition probability associated with the transition $d \to d'$ (or $d \to A_s$) is exactly the entry of T for $\tilde{x}_m \to \tilde{x}'_m$, where $\tilde{x}_m$ is the m-tuple suffix of d, and $\tilde{x}'_m$ is the m-tuple suffix of d' (or of dx when $d \to A_s$). Transition probabilities for automaton states are stored in a matrix Ω.

2.3 Implementation of computation

The vector $\psi_n = \tilde{\pi} \times \Omega^{n-m}$ has elements $\psi_{n,d}$ that give the probability that the automaton lies in state d at time n. Then $\Pr\left(S(w) \ge s\right) = \psi_{n,A_s}$, the probability of the absorbing state.

Since there are only k non-zero elements in each row of Ω, that matrix is stored in a sparse fashion. Probability vectors ψ_r are updated sequentially using $\psi_r = \psi_{r-1}\Omega$, $r = m+1, \ldots, n$, with initial condition $\psi_m = \tilde{\pi}$. Each update requires about $k\left|\Delta_s\right|$

multiplications and additions, for a total of $(n-m)k|\Delta_s|$ of each operation. Thus the computation is linear in n, but $|\Delta_s|$, though smaller than $|\Delta| = k^{w-1}$, nonetheless grows very quickly with window size w.

Note that we could compute Ω^{n-m} in relatively few multiplications using "matrix doubling" (as was done in Martin and Coleman 2011), however the sparseness of Ω^u goes away as the power of u increases, so that the advantage of using sparse matrix algebra disappears. Multiplying two square matrices of order $|\Delta_s|$ uses $O\left(|\Delta_s|^3\right)$ operations. Thus for large w (and corresponding large values of $|\Delta_s|$), it is more economical to sequentially update ψ_r instead of first forming Ω^{n-m} and then multiplying $\tilde{\pi}$ by it.

3. Numerical Examples

We wrote a FORTRAN program to implement the algorithm described in Section 2. The program contains a subroutine designed for computing the complete distribution by using the complementary nature of the state spaces for $s^{(b)} = \xi + q$ and $s^{(a)} = \xi - q$. It also has a subroutine for computing p-values $\Pr(S(w) \geq s)$ for specified values of s. Below are basic examples to illustrate the steps of the algorithm, and to give some numerical output.

Consider first a case where $\mathbf{X}$ is a first-order Markov chain ($m=1$), with $\Sigma = \{0,1,2\}$ so that $k=3$. Tables 1 and 2 given earlier give the number of states used to compute $\Pr(S(w) \geq s)$. If $w=4$ and $s = 7 \text{ and } 3$, complementary values that are symmetric about $\xi = \frac{w(k-1)}{2} + 1 = 5$, $\Sigma_7^1 = \Sigma_3^1 = \Sigma$, $D_7 = \{12,21,22,122,212,221,222\}$ and $D_3 = \{20,11,10,200,110,101,100\}$, where we have listed states so that state i of D_7 is complementary to state i of D_3 (for example state 12 of D_7 and 20 of D_3 are complementary, with respective slack values $(0,4)$ and $(4,0)$).

Using the transition matrix

$$T_1 = \begin{pmatrix} 0.5 & 0.2 & 0.3 \\ 0.4 & 0.2 & 0.4 \\ 0.6 & 0.1 & 0.3 \end{pmatrix}$$

and stationary initial distribution $\pi = \left(\frac{52}{101}, \frac{17}{101}, \frac{32}{101}\right)$ (computed using stationary $\pi T = \pi$) as input to the algorithm (the corresponding states are listed in lexigraphical order, so that, for example, $P(X_1 = 0) = 52/101$), we obtained p -values for $w = 1,\ldots,12$ and $s = 2w, 2w-1,\ldots,1$ (Table 3). The CPU times were all less than a second for $4 \leq w \leq 8$, but increase greatly with w. The computations were terminated if $\Pr(S(w) \geq s) > 0.99995$ since probabilities are listed to four significant digits and Table

Table 3. Probabilities $\Pr\left(S(w) \ge s\right)$, $s = 1, 2, \ldots, 2w$ for $n = 100$, transition matrix T_2, $\Sigma = \{0,1,2\}$ $(k = 3)$, $m = 1$, and various values of w.

s	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$	$w=9$	$w=10$	$w=11$	$w=12$
1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1
6	0.9959	0.9999	1	1	1	1	1	1	1
7	0.8058	0.9878	0.9994	1	1	1	1	1	1
8	0.4509	0.8684	0.9839	0.9985	0.9999	1	1	1	1
9		0.4202	0.8334	0.9708	0.9960	0.9995	0.9999	1	1
10		0.1605	0.5264	0.8344	0.9600	0.9922	0.9986	0.9998	1
11			0.1664	0.5131	0.8051	0.9418	0.9855	0.9967	0.9993
12			0.0503	0.2382	0.5359	0.7889	0.9243	0.9769	0.9936
13				0.0587	0.2464	0.5216	0.7605	0.9016	0.9650
14				0.0152	0.0927	0.2734	0.5186	0.7375	0.8789
15					0.0197	0.1030	0.2760	0.5023	0.7094
16					0.0045	0.0336	0.1212	0.2834	0.4898
17						0.0065	0.0400	0.1282	0.2810
18						0.0013	0.0117	0.0497	0.1369
19							0.0021	0.0149	0.0551
20							0.0004	0.0040	0.0194
21								0.0007	0.0054
22								0.0001	0.0014
23									0.0002
24									0.0000
Time	0.0s	0.0s	0.0s	0.1s	0.8s	6.5s	59.6s	9m14s	82m44s

$\Pr\left(S(w) \ge s-1\right) > \Pr\left(S(w) \ge s\right)$. In Table 4, p-values were computed for $w = 13$, 14, and 15 and $s = 2w, 2w-1, \ldots$; in this case the computation was terminated when $\Pr\left(S(w) \ge s\right) > 0.05$. These computations are useful for determining critical values of test procedures.

We also display output for the case where **X** is a second-order Markov chain ($m = 2$), with $\Sigma = \{0,1\}$. Table 5 gives $|\Delta_s|$ for $w = 4, \ldots, 20$ and $s = w, w-1, \ldots, 2$. Only $\max|\Delta_s| = \left|\Delta_{|\xi|+1}\right|$ is shown for $w = 18$, 19, and 20. Table 6 has complete distributions for the various values of w, and using transition matrix

$$T_2 = \begin{pmatrix} 0.7 & 0.3 & 0 & 0 \\ 0 & 0 & 0.4 & 0.6 \\ 0.6 & 0.4 & 0 & 0 \\ 0 & 0 & 0.9 & 0.1 \end{pmatrix}$$

and the corresponding stationary initial distribution $\pi = \left(\frac{3}{7}, \frac{3}{14}, \frac{3}{14}, \frac{1}{7}\right)$ as input.

As in Ebneshahrashoob et al. (2005), we also computed $P\left(S\left(w\right) \geq w\right)$ for large values of n and w. These probabilities may be used to obtain the distribution of the longest success run by varying w. For $n = 10^6$, $\Pr\left[X_t = 1 \middle| X_{t-1} = 1\right] = 0.75$, $\Pr\left[X_t = 1 \middle| X_{t-1} = 0\right] = 0.25$ and the stationary probability of success $p = 0.5$, our probabilities $P\left(S\left(w\right) \geq w\right)$ matched those listed in Table 3 of the latter paper. We also show output for

$$T_3 = \begin{pmatrix} 0.7 & 0.3 & 0 & 0 \\ 0 & 0 & 0.4 & 0.6 \\ 0.6 & 0.4 & 0 & 0 \\ 0 & 0 & 0.4 & 0.6 \end{pmatrix}$$

and the stationary initial distribution $\pi = \left(\frac{4}{11}, \frac{2}{11}, \frac{2}{11}, \frac{3}{11}\right)$ in Tables 7 and 8 for window sizes $w \in \left\{40, 50, 60, 70, 80\right\}$ and $n = 10^6$.

Table 4. Probabilities $\Pr\left(S(w) \geq s\right)$, $s = 2w, 2w-1, \ldots$ for $n = 100$, transition matrix T_2, $\Sigma = \left\{0,1,2\right\}$, $m = 1$, and various values of w.

s	$w=13$	$w=14$	$w=15$
20	0.0611		
21	0.0225	0.0648	
22	0.0074	0.0259	0.0682
23	0.0019	0.0089	0.0285
24	0.0005	0.0027	0.0106
25	0.0001	0.0007	0.0034
26	0.0000	0.0002	0.0010
27		0.0000	0.0002
28		0.0000	0.0000
29			0.0000
30			0.0000
Time	4m6s	49m22s	575m17s

Summary

We have given a method to compute exact p-values or the complete distribution of the one-dimensional scan statistic for multi-state trials and higher-order Markovian sequences. The basic algorithm is to set up a deterministic finite automaton with final states that correspond to strings that indicate that $\left[S\left(w\right)\geq s\right]$, form a transition matrix for transitions of automaton states, and then update probabilities held in a probability vector using matrix-vector updates. Symmetry in the number of states was used to make the computation more efficient when the entire distribution is desired.

Table 5. State space size $\left|\Delta_s\right|$, $s=2,3\ldots,w$ for $\Sigma=\left\{0,1\right\}$, $m=2$, and various values of w.

s	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$	$w=9$	$w=10$	$w=11$	$w=12$
2	5	6	7	8	9	10	11	12	13
3	7	11	16	22	29	37	46	56	67
4	6	11	21	36	57	85	121	166	221
5		7	16	36	71	127	211	331	496
6			8	22	57	127	253	463	793
7				9	29	85	211	463	925
8					10	37	121	331	793
9						11	46	166	496
10							12	56	221
11								13	67
12									14

s	$w=13$	$w=14$	$w=15$	$w=16$	$w=17$	$w=18$	$w=19$	$w=20$
2	14	15	16	17	18			
3	79	92	106	121	137			
4	287	365	456	561	681			
5	716	1002	1366	1821	2381			
6	1288	2003	3004	4369	6189			
7	1717	3004	5006	8009	12377			
8	1717	3433	6436	11441	19449			
9	1288	3004	6436	12871	24311			
10	716	2003	5006	11441	24311	48621	92379	
11	287	1002	3004	8009	19449			184757
12	79	365	1366	4369	12377			
13	15	92	456	1821	6189			
14		16	106	561	2381			
15			17	121	681			
16				18	137			
17					19			

Table 6. Distributions $\Pr\left(S(w) \geq s\right)$, $s = 1, 2, \ldots, w(k-1)$ for $n = 1000$, transition matrix T_2, $\Sigma = \{0,1\}$, $m = 2$, and various values of w. "Time" stands for CPU time.

s	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$	$w=9$	$w=10$	$w=11$	$w=12$
1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1
4	0.7243	1	1	1	1	1	1	1	1
5		0.1203	0.9962	1	1	1	1	1	1
6			0.0127	0.5579	0.9998	1	1	1	1
7				0.0013	0.1030	0.8724	0.9996	1	1
8					0.0001	0.0135	0.3183	0.9404	0.9991
9						0.0000	0.0016	0.0602	0.5288
10							0.0000	0.0002	0.0091
11								0.0000	0.0000
12									0.0000
Time	0.0s	0.0s	0.0s	0.0s	0.0s	0.1s	0.1s	0.2s	0.6s

s	$w=13$	$w=14$	$w=15$	$w=16$	$w=17$	$w=18$	$w=19$	$w=20$
1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1
9	0.9394	0.9974	1	1	1	1	1	1
10	0.1496	0.6262	0.9306	0.9941	0.9997	1	1	1
11	0.0013	0.0297	0.2432	0.6489	0.9114	0.9876	0.9990	0.9999
12	0.0000	0.0002	0.0051	0.0636	0.3021	0.6475	0.8870	0.9771
13	0.0000	0.0000	0.0000	0.0008	0.0134	0.0995	0.3289	0.6294
14		0.0000	0.0000	0.0000	0.0001	0.0025	0.0259	0.1263
15			0.0000	0.0000	0.0000	0.0000	0.0004	0.0058
16				0.0000	0.0000	0.0000	0.0000	0.0001
17					0.0000	0.0000	0.0000	0.0000
18						0.0000	0.0000	0.0000
19							0.0000	0.0000
20								0.0000
Time	1.3s	5.1s	14.5s	68.3s	5m11s	17m41s	83m16s	376m39s

Table 7. $\Pr\left(S(w) \geq w\right)$ for $n = 10^6$, transition matrix T_3, $\Sigma = \{0,1\}$, $m = 2$, and various values of w. CPU times are listed below the probabilities.

$w = 40$	$w = 50$	$w = 60$	$w = 70$	$w = 80$
0.0004 5.2s	2.45e-6 6.5s	1.48e-8 7.7s	8.95e-11 8.9s	5.41e-13 10.1s

Table 8. $\Pr\left(S(w) \geq w-1\right)$ for $n = 10^6$, transition matrix T_3, $\Sigma = \{0,1\}$, $m = 2$, and various values of w. CPU times are listed below the probabilities.

$w = 40$	$w = 50$	$w = 60$	$w = 70$	$w = 80$
0.0070 1m34s	5.361e-5 2m28s	4.000e-7 3m35s	2.756e-9 4m56s	1.907e-11 6m31s

Appendix

In this Appendix we obtain an expression for the number of states in Δ_s.

Theorem A.1. Let $\Sigma = \{0,1,\ldots,k-1\}$ and consider the string $x_1 \cdots x_c$. The number of integer solutions $\eta(k,c,\mu)$ of $x_1 + x_2 + \ldots + x_c = \mu$, where μ is a positive integer and $x_i \in \Sigma$, is

$$\eta(k,c,\mu) = \sum_{j=0}^{\lfloor \mu/k \rfloor} (-1)^j \binom{c}{j} \binom{c-1+\mu-jk}{c-1}. \tag{A.1}$$

Proof: The result is obtained by considering the series expansion of $\left(\sum_{i=0}^{k-1} y^i\right)^c$ for $|y| < 1$ [see Murty, (1981)]. Taking y^{x_1} from the first factor, y^{x_2} from the second, …, and y^{x_c} from the c th factor and multiplying, we obtain $y^{x_1+x_2+\ldots+x_c}$, and thus the coefficient of y^{μ} in the expansion is precisely the total number of solutions of $x_1 + x_2 + \ldots + x_c = \mu$.

Since $\sum_{i=0}^{k-1} y^i = \left(1-y^k\right)\left(1-y\right)^{-1}$,

$$\left(\sum_{i=0}^{k-1} y^i\right)^c = \left(1-y^k\right)^c \left(1-y\right)^{-c}$$
$$= \left[\sum_{j=0}^{c} \binom{c}{j} (-1)^j y^{jk}\right]\left[\sum_{r=0}^{\infty} \binom{c+r-1}{c-1} y^r\right].$$

The term of order μ is

$$\sum \binom{c}{j}\binom{c+r-1}{c-1}(-1)^j y^{jk+r} ,$$

where the summation runs over values of j and r such that $jk+r=\mu$; $r \geq 0$, $0 \leq j \leq c$, or

$$r=\mu - jk \text{ ; } \frac{\mu}{k} \geq j \text{ ; } 0 \leq j \leq c \text{ , i.e. } 0 \leq j \leq \frac{\mu}{k} .$$

Hence the required coefficient is $\eta(k,c,\mu) = \sum_{j=0}^{\lfloor \mu/k \rfloor} (-1)^j \binom{c}{j}\binom{c-1+\mu-jk}{c-1}$. ■

Theorem A.2 The number of automaton states in Δ_s is

$$|\Delta_s| = 1 + \sum_{\mu=0}^{\min[m(k-1),s-1]} [\eta(k,m,\mu)]$$
$$+ \sum_{c=m+1}^{w-1} \left\{ \sum_{\mu=\max[1,s-(w-c)(k-1)]}^{\min[c(k-1),s-1]} [\eta(k,c,\mu)] - \sum_{\mu=\max[1,s-(w-c)(k-1)]}^{\min[(c-1)(k-1),s-1]} [\eta(k,c-1,\mu)] \right\}$$

Proof: The number of strings of length m with window sums $x_1 + x_2 + \cdots + x_m = \mu$ is $\eta(k,m,\mu)$ for $\mu = 0,1,\ldots,\min[m(k-1),s-1]$. Based on this fact, $|\Sigma_s^m| = \sum_{\mu=0}^{\min[m(k-1),s-1]} \eta(k,m,\mu)$. To obtain $|D_s|$, $\sum_{c=m+1}^{w-1} \left(\sum_{\mu=\max[1,s-(w-c)(k-1)]}^{\min[c(k-1),s-1]} [\eta(k,c,\mu)] \right)$ gives the number of strings of lengths $c = m+1,\ldots,w-1$ with window sums μ in the acceptable range. However, we must subtract the number of strings $\eta(k,c-1,\mu)$ that have the same window sum μ but begin with zero. For strings that begin with zero, the upper bound for μ may be smaller because the string length after the first symbol is only $c-1$ and not c . The additional state is the absorbing state. The result follows. ■

For $s = w(k-1)$ or $s = w(k-1)-1$, we give a more compact form of $|D_s|$.

Theorem A.3 If $s = w(k-1)$, $|D_s| = w-m-1$ so that $|\Delta_s| = k^m + w - m$. For $s = w(k-1)-1$,

$$|D_s| = \begin{cases} \binom{w+1}{2} - \binom{m+2}{2}, & k = 3,4,\ldots \\ \binom{w+1}{2} - \binom{m+2}{2} - (w-m), & k = 2 \end{cases}$$

and again $|\Delta_s| = |D_s| + k^m + 1$.

Proof: If $s = w(k-1)$, $\varsigma_{1,0} = 0$, and all strings in D_s must be of the form $(k-1,k-1,\cdots,k-1)$. There is then exactly one string in D_s for each of the lengths $m+1,\ldots,w-1$, for a total of $w-m-1$ of such strings.

If $s = w(k-1)-1$ and $k \geq 3$, there is exactly one m-tuple with $\varsigma_{1,m} = 1$, m with $\varsigma_{1,m} = 0$, and the rest have $\varsigma_{1,m} < 0$. Strings of length j, $j = m,\ldots,w-2$ with $\varsigma_{1,j} = 0$ produce one automaton string of length $j+1$ (by concatenating $k-1$ as the last component), while the one string of each length with $\varsigma_{1,j} = 1$ produces two automaton strings (by concatenating $k-1$ to produce a string with $\varsigma_{1,j+1} = 1$ or by concatenating $k-2$). Thus D_s has $j+1$ strings of length j, $j = m+1,\ldots,w-1$, for a total of number of strings equal to

$$m+2+\cdots+w = \frac{w(w+1)}{2} - \frac{(m+1)(m+2)}{2}.$$

If $k = 2$, the m-tuple $\left(0,\overbrace{1,\ldots,1}^{m-1}\right)$ has slack $\varsigma_{1,j} = 0$, however this state isn't included in the automaton, and its offspring shouldn't be either. There is one such string for each of the lengths $m+1,\ldots,w-1$, for a total of $w-m-1$ strings that should be subtracted. In addition, the string $\left(\overbrace{1,1,\ldots,1}^{w-1}\right)$ has $\varsigma_{1,w-1} = 1$, however for that string $\varsigma_{2,w-1} = -1$ as it is absorbed. Thus it should not be counted, and a total of $(w-m-1)+1 = w-m$ strings should be subtracted from the count when $k > 2$ to obtain the count when $k = 2$.

In each case, k^m m-tuples and also an absorbing state are included in Δ_s, and thus $|\Delta_s| = |D_s| + k^m + 1$. ■

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grants DMS-0805577 and DMS-1107084.

References

J. Chen, *Approximations and Inequalities for Discrete Scan Statistics*. Ph. D. dissertation, University of Connecticut, Storrs, CT, 1998.

M. Ebneshahrashoob, T. Gao, and M. Wu, "An efficient algorithm for exact distribution of discrete scan statistics," *Methodology and Computing in Applied Probability*, vol. 7, pp. 459-471, 2005.

J. C. Fu, "Distributions of the scan statistic for a sequence of bistate trials." *Journal of Applied Probability*, vol. 38, pp. 908-916, 2001.

J. Glaz, "Approximations and bounds for the distribution of the scan statistic." *Journal of the American Statistical Association,* vol. 84, pp. 560-569, 1989.

J. Hoh and J. Ott, "Scan statistics to scan markers for susceptible genes." *Proceedings of the National Academy of Science USA*, vol. 97, pp. 9615-9617, 2000.

J. E. Hopcroft, R. Motwani, JD Ullman (2001). *Introduction to Automata Theory, Languages and Computation*. Addison Wesley.

S. Karlin, B.E. Blaisdell, E.S. Mocarski, and V. Brendel, "A method to identify distinctive charge configurations in protein sequences, with application to human herpesvirus polypeptides," *Journal of Molecular Biology*, vol. 205, pp. 165-177, 1989.

C. R. Loader, "Large-deviation approximations to the distribution of scan statistics." *Advances in Applied Probability, vol.* 23, pp. 751-771, 1991.

D. E. K. Martin and D. Coleman, "Distributions of Clump Statistics for a Collection of Words." *Journal of Applied Probability*, vol. 48, pp. 1049-1059, 2011.

V. N. Murty, "Counting the integer solutions of a linear equation with unit coefficients." *Mathematics Magazine*, vol. 54 (2), pp. 79-81, 1981.

J. Naus, "Probabilities for a generalized birthday problem," *Journal of the American Statistical Association*, 69, 810-815, 1974.

J. Naus, "Approximations for distributions of scan statistics," *Journal of the American Statistical Association,* vol. 77, pp. 377-385, 1982.

P. Ribeca, and E. Raineri, "Faster exact Markovian probability functions for motif occurrences: a DFA-only approach." *Bioinformatics* , vol. 24(24), pp. 2839-2848, 2008.

S. Robin, F. Rodolphe, and S. Schbath, *DNA, Words and Models*, Cambridge University Press, 2005.

Y. V. Sun, D. M. Jacobsen, and S. L. R. Kardia (2006). "ChromoScan: a scan statistic application for identifying chromosomal regions in genomic studies," *Bioinformatics*, vol. 22(23), 2945-2947.

A. Wagner, "Genes regulated cooperatively by one or more transcription factors and their identification in whole eukaryotic genomes," *Bioinformatics*, vol. 15, pp. 776-784, 1999.