WAITING TIME DISTRIBUTION OF GENERALIZED LATER PATTERNS

Donald E. K. Martin,[*] *North Carolina State University*
John A. D. Aston,[**] *Warwick University, UK and Academia Sinica, Taiwan*

Abstract

In this paper the concept of later waiting time distributions for patterns in multi-state trials is generalized to cover a collection of compound patterns that must all be counted pattern-specific numbers of times, and a practical method is given to compute the generalized distribution. The solution given applies to overlapping counting and two types of non-overlapping counting, and the underlying sequences are assumed to be Markovian of a general order. Patterns are allowed to be weighted so that an occurrence is counted multiple times, and patterns may be completely included in longer patterns. The probabilities are computed through an auxiliary Markov chain. As the state space associated with the auxiliary chain can be quite large if its setup is handled in a naïve fashion, an algorithm is given for generating a "minimal" state space that leaves out states that can never be reached. For the case of overlapping counting, a formula that relates probabilities for intersections of events to probabilities for unions of subsets of the events is also used, so that the distribution is also computed in terms of probabilities for competing patterns. A detailed example is given to illustrate the methodology.

Keywords: Auxiliary Markov chain; competing patterns; completely included simple patterns; higher-order Markovian sequence; multi-state trials; sooner and later waiting time distributions

[*] Corresponding author; Postal address: NCSU Department of Statistics, 2501 Founders Drive, Campus Box 8203, 209E Patterson Hall, Raleigh, North Carolina 27695-8203, USA. Email address: martin@stat.ncsu.edu. Phone number: 1-919-515-1936.
[**] Postal address: Centre for Research in Statistical Methodology, Department of Statistics, University of Warwick, Coventry, UK. Email address: j.a.d.aston@warwick.ac.uk.

1. Introduction

This paper addresses the computation of generalized later waiting time distributions. Traditionally, later waiting time problems deal with the time until the occurrence of all of the members of a group of patterns, whereas sooner waiting time problems deal with the time to the first occurrence of one of them. Research on these distributional problems was pioneered by Ebneshahrashoob and Sobel (1990), and includes Ling (1992), Ling and Low (1993), Kolev and Minkova (1999a, 1999b), and Fu and Chang (2003).

The computation of distributions associated with increasingly complex patterns is driven by statistical applications in many fields, such as reliability of engineering systems, national and computer security, quality control, and psychology. The reader is referred to Balakrishnan and Koutras (2002) for a survey of the theory and applications of distributions of patterns. Below, we consider a prototypical application in the field of bioinformatics.

The survival and replication of cells depends on the recognition, by many different agents, of sites (*patterns* or *words*) on DNA. For example, polymerases recognize promoters and start transcription, the first step of protein synthesis, which is in turn regulated by other agents, like repressors or activators. Since the extent, chronology, and cell specificity of transcription are modulated by the interaction of transcription factors and their binding sites, the location of binding sites for unknown factors that regulate a collection of co-regulated genes is an important task, though not a simple one. Binding sites of the same transcription factor are generally short and degenerate (similar but not identical) patterns, called *motifs*, that can vary greatly, and may be located quite far from the corresponding coding region, either upstream, downstream, or in the introns. The binding sites of multiple interacting factors often play a role in the regulation of a single gene (Pavesi et al., 2004; Sinha and Tompa, 2002). An example is the developmental gene CYIIIa from the sea urchin *Strongylocentrotus purpuratus*, the promoter region of which is 2,300 base pairs upstream from the coding region, and whose expression is regulated by at least nine different transcription factors that bind at over 20 specific sites (Kirchhamer et al., 1996). Also, a single isolated pattern occurrence may be insufficient to properly exert the regulatory function (van Helden et al., 1998).

One method that biologists use to identify potential binding sites is to search for patterns that are significantly over-represented in regulatory regions of sets of genes sharing common properties such as expression profile or biological function (see, for example, van Helden et al., 2000; Hampson et al., 2002; Mariño-Ramirez et al., 2004; Sumazin et al., 2005). The exceptionality of the pattern counts is determined using a model (typically a Markov model of some order) for the background noise. The parameters of the model are determined from the sequence used to obtain the pattern count, and the model is then used to determine what is "expected."

Robin et al. (2005) gave a description of the rationale behind modeling DNA sequences and using statistical methods to locate exceptional patterns along them. In that book, results are given on both exact and approximate (normal, Poisson, or compound Poisson) probabilities for counts of single patterns, where overlapping occurrences are counted. In contrast, Biggins and Cannings (1987) considered a model for a restriction enzyme that cuts the DNA sequence in such a way that doesn't recognize a pattern that overlaps a previous one, and in such a model, a non-overlapping counting technique would be used.

In the spirit of the previous references, we develop a tool for computing pattern distributions in data modeled as being Markovian of a general order. In doing so, we take into account specific features that may be encountered in difficult computational problems such as determining exceptional patterns in DNA sequences: variation of patterns and interaction among patterns that serve a single role, the need for multiple pattern occurrences in a small region to carry out a function, and the different counting techniques that may be used. However, since the authors cannot anticipate all of the types of pattern systems whose occurrence probabilities a researcher in bioinformatics or another field may want to quantify, we pose the problem in more generality than may have been encountered in any specific problem to date. Instead of restricting the pattern search to similar patterns, we allow collections of patterns that are grouped based on any criteria. The count for a group of patterns is incremented whenever one of the patterns of the group is counted. Several groups of patterns are allowed, and each group must be counted a pattern-specific number of times. By allowing "or" relations among pattern groups, and "and" relations between groups, great flexibility in modeling systems of patterns is realized. In addition, different methods of counting patterns is allowed: overlapping counting, and also two types of non-overlapping counting, system-wide, and within-patterns. Finally, patterns are allowed to be repeated either within groups or in different groups, and patterns can be completely included in longer patterns of the system.

The setup of this paper is similar to that in Aston and Martin (2005), where the waiting time distribution of interest is the first time that a group of patterns is counted its specified number of times. However, in addition to the fact that the state space required to compute the "later" waiting time is larger than the state space for the "sooner" waiting time of the latter reference, and in fact contains it, in that paper weighted patterns were not allowed, and completely included patterns were not dealt with explicitly.

An auxiliary Markov chain is developed to make a rather intractable combinatorial problem one that can be handled. Cox (1955) used a similar approach, converting a stochastic process representing lifetimes into an auxiliary Markov process to simplify the analysis, and noted that the technique was well known at the time. Fu and Koutras (1994) reformulated the method for processes with a finite state space in discrete time (calling it *finite Markov chain imbedding*), providing an elegant framework for handling intricate run and pattern distributions.

Due to the generality that we allow, the state space associated with the auxiliary Markov chain used to compute the generalized later waiting time distribution can be quite large. To make computation more efficient, we determine a “minimal” state space by leaving out states that have zero probability of occurring. We also drop states that can occur only in the initialization stage (for times t less than the order of Markovian dependence m) after time $t = m$. Efficient computation is an important feature of this work.

In the case of overlapping counting, a formula for the probability of the intersection of events totally in terms of probabilities of unions of subsets of the events is used, in conjunction with previous results for competing patterns (Aston and Martin, 2005), to give an alternative method of computing the waiting time distribution of generalized later patterns.

The paper is organized as follows. The next section gives formal definitions of the system of patterns under study and some preliminary material needed for the description of the computational method that is laid out in Section 3, most notable of which is the automation of setting up the state space of the auxiliary Markov chain. A detailed example is used in Section 3 for pedagogic purposes. The final section is a discussion.

2. Definitions and preliminaries

For notational purposes, let $X_{-m+1}, X_{-m+2}, ..., X_0, X_1, X_2, ...$ be a multi-state m-th order Markovian sequence with finite state space S_X of cardinality $|S_X| \geq 2$. The associated initial probabilities and time-invariant transition probabilities of the sequence are respectively denoted by

$$\pi(x_{-m+1}, \ldots, x_0) \equiv P(X_{-m+1} = x_{-m+1}, \ldots, X_0 = x_0) \ ;$$

$$p(x_t \,|\, x_{t-m},\ldots,x_{t-1}) \equiv P(X_t = x_t \,|\, X_{t-m} = x_{t-m},\ldots,X_{t-1} = x_{t-1})\ .$$

2.1 Notation for patterns

A *simple pattern* $\Lambda_i = \Lambda_{i,1}\ldots\Lambda_{i,k_i}$ is composed of a specified sequence of k_i symbols of S_X, where the symbols may be repeated. A *compound pattern* $\Lambda^{(j)} = \left(\Lambda_1,\ldots,\Lambda_{\eta_j}\right)$ is a collection of η_j simple patterns, such that the occurrence count of the compound pattern is incremented whenever one of its simple patterns is counted as having occurred. No restrictions are placed on the simple patterns making up the compound patterns. Simple patterns may be repeated either within the same compound pattern or within different ones. The effect of repeating a simple pattern within a compound pattern is that the occurrence of the repeated simple pattern can lead to the count of the compound pattern being incremented multiple times. Also, the case where one or more simple patterns lie completely within a longer pattern of the same compound pattern or of a different one is not excluded. Such patterns are referred to as *completely included simple patterns* (CISP).

Let $\Psi = \left\{\Lambda^{(1)}, r_1,\ldots,\Lambda^{(c)}, r_c\right\}$, $c \ge 1$, be a system consisting of a collection of c compound patterns $\Lambda^{(1)},\ldots,\Lambda^{(c)}$, along with associated numbers of occurrences $r_1,\ldots,r_c$, respectively. Without loss of generality, the compound patterns are assumed to be unique since if two compound patterns are identical, say $\Lambda^{(j_1)} = \Lambda^{(j_2)}$ with $r_{j_1} \ge r_{j_2}$, then the waiting time for the occurrence of all of the compound patterns of the system Ψ is the same if $\left(\Lambda^{(j_2)}, r_{j_2}\right)$ is removed.

In the situation where the c compound patterns "compete" to be the first to be counted their specified number of times, with certain restrictions on the allowed patterns, Aston and Martin (2005) called Ψ *competing patterns*, and computed the associated generalized sooner waiting time distribution. Here we require that all of the compound patterns of Ψ be counted their specified numbers of times, and call the system *generalized later patterns.*

If the generalized later pattern is denoted by L, then the waiting time distribution of interest is given by

$$P(L \le t),\ t \in \mathbb{N}. \tag{2.1}$$

If $c=1$, then (2.1) reduces to the waiting time for the r_1-th count of the compound pattern $\Lambda^{(1)}$. If $r_j = 1$ for all j, and, in addition, each compound pattern consists of only one simple pattern, then (2.1) reduces to the later waiting time distribution considered in the literature. Considering the generality of patterns allowed, counting techniques, and model orders, the theory presented here, in conjunction with the work on competing patterns, provides a general framework encompassing a wide range of waiting time distributions associated with patterns.

Before discussing the various counting techniques that will be used, we give additional notation dealing with overlap of patterns. For a simple pattern $\Lambda_i = \Lambda_{i,1},\ldots,\Lambda_{i,k_i}$ of length k_i, sub-patterns of the form $\Lambda_{i,1},\ldots,\Lambda_{i,d}$, $d = 1,\ldots,k_i - 1$, are called *prefixes*, and sub-patterns of the form $\Lambda_{i,q},\ldots,\Lambda_{i,k_i}$, $q = 1,\ldots,k_i$, are called *suffixes*. Notice that the full pattern is a suffix of itself. Also note that the definition of a prefix, suffix, or CISP applies to any simple pattern, and is not restricted to simple patterns of the system Ψ. In particular, the definitions also apply to m-tuples $(x_{t-m+1},\ldots,x_t)$ that will be encountered later.

Patterns are said to *overlap* if a suffix of one is a prefix of the other (a pattern can overlap itself as well). For example AT overlaps TT since T is a suffix of AT and a prefix of TT, whereas TT overlaps itself, since T is both one of its suffixes and prefixes.

A CISP $\Lambda_i = \Lambda_{i,1},\ldots,\Lambda_{i,k_i}$ can be a suffix to a longer simple pattern $\Lambda_l = \Lambda_{l,1},\ldots,\Lambda_{l,k_l}$, lie within Λ_l as a non-suffix ($\left(\Lambda_{i,1},\ldots,\Lambda_{i,k_i}\right)=\left(\Lambda_{l,q_1},\ldots,\Lambda_{l,q_2}\right)$, $1\le q_1 \le q_2 < k_l$), or both (if it is included in Λ_l more than once and is one of its suffixes). For example, TT is included in $CTTT$ as both a suffix and a non-suffix, the two occurrences of TT overlapping.

We define E to be the set of prefixes for Ψ, and let E_j denote the set of prefixes of compound pattern $\Lambda^{(j)}$, with $E_{j,i}$ being the set of prefixes for the simple patterns of $\Lambda^{(j)}$, $i=1,\ldots,\eta_j$. Then $E=\bigcup_{j=1}^{c} E_j = \bigcup_{j=1}^{c}\bigcup_{i=1}^{\eta_j} E_{j,i}$. Analogous definitions are given for suffix sets F, F_j, and $F_{j,i}$.

By the inclusion-exclusion principle, the number of prefixes in E_j is given by

$$\left|E_j\right| = \sum_{i=1}^{\eta_j}\left|E_{j,i}\right| - \sum_{i_1<i_2}\left|E_{j,i_1}\cap E_{j,i_2}\right| + \sum_{i_1<i_2<i_3}\left|E_{j,i_1}\cap E_{j,i_2}\cap E_{j,i_3}\right| \ldots - \ldots + (-1)^{\eta_j - 1}\left|\bigcap_{\nu=1}^{\eta_j} E_{j,i_\nu}\right|. \quad (2.2)$$

In (2.2), $\left|E_{j,i}\right| = k_i - 1$, and for $1 < l \le \eta_j$ $\sum_{i_1<i_2<\ldots<i_l}\left|\bigcap_{\nu=1}^{l} E_{j,i_\nu}\right|$ gives the sum of the number of prefixes common to l of the sets $E_{j,i}$, the sum taken over the $\binom{\eta_j}{l}$ combinations of sets. Recalling that the longest prefix in $E_{j,i}$ is of length $k_i - 1$, we have, for $1 < l \le \eta_j$,

$$\left|\bigcap_{\nu=1}^{l} E_{j,i_\nu}\right| = \begin{cases} \max\left\{\omega : (\Lambda_{i_1,1},\ldots,\Lambda_{i_1,\omega}) = \ldots = \ldots(\Lambda_{i_l,1},\ldots,\Lambda_{i_l,\omega}) \text{ and } 1\le\omega\le\min(k_{i_1},k_{i_2},\ldots,k_{i_l})\text{-1}\right\} \\ 0 \quad \text{if there is no common prefix for the } l \text{ sets} \end{cases}.$$

2.2 Counting techniques

We say that a simple pattern $\Lambda_i = \Lambda_{i,1}, \ldots, \Lambda_{i,k_i}$ occurs at position $q \geq k_i$ of the sequence $X_1, \ldots, X_n$ if $\left(X_{q-k_i+1}, \ldots, X_q\right) = \left(\Lambda_{i,1}, \ldots, \Lambda_{i,k_i}\right)$, but whether Λ_i is counted when it occurs is determined by the counting technique and whether it has a prefix that is the suffix of another simple pattern that is counted to re-start counting. Both *overlapping* and *non-overlapping* methods of counting are used. With overlapping counting, all simple patterns that occur are counted, regardless of whether they overlap another simple pattern or have CISP. With *system-wide non-overlapping* (SWNO) counting, when a simple pattern occurs and is counted, the reckoning of any simple pattern in the system Ψ re-starts from that point, so that any simple pattern prefixes that are active are ignored. With *within-pattern non-overlapping* (WPNO) counting, a simple pattern that occurs and is counted only re-starts pattern reckoning for the compound pattern of the simple pattern that has just been counted.

For $\Lambda^{(1)} = (AGGTA, CG, GT, C)$, $\Lambda^{(2)} = (CGTT, AG, GT, GT)$, $\Lambda^{(3)} = (AT, CT, GT, TT)$, and the DNA sequence $CTCGTT$ of length $n = 6$, the simple pattern C occurs at position 1, CT occurs at position 2, C at position 3, CG at position 4, GT at position 5, and both $CGTT$ and TT at position 6. All of these occurrences are counted under overlapping counting, whereas for WPNO and SWNO counting, CG is not counted because the simple pattern C of the same compound pattern is counted at position 3, re-starting counting; and the occurrence of GT at position 5 re-starts counting for all of the compound patterns since it lies in all of them, and thus $CGTT$ and TT are not counted at position 6. For SWNO counting, CT, which occurs at position 2, is also not counted since C re-starts counting, whereas CT is counted under WPNO counting, since it is in a different compound pattern from C. The counted simple patterns for this example along with the corresponding occurrence locations are listed in Table 1.

The assumption is made that only simple patterns of compound patterns that have not previously been counted their associated number (r_j) of times re-start counting under non-overlapping counting, since those are the only simple patterns that need to be considered when waiting for the occurrence of the generalized later pattern. In this way, under non-overlapping counting, simple patterns that contain one or more CISP can be counted after the compound patterns containing the CISP have been counted their required number of times. For example, CT of $\Lambda^{(3)}$ is counted under SWNO counting if $\Lambda^{(1)}$ has previously been counted r_1 times. However, if a compound pattern has a simple pattern and also patterns that are completely contained in the simple pattern as non-suffixes, the simple pattern containing the CISP will never be counted, and thus is not included in the system when the computations are carried out. As examples, for $\Lambda^{(1)}$, $\Lambda^{(2)}$, and $\Lambda^{(3)}$ above, under non-overlapping counting, $AGGTA$, CG and $CGTT$ may be dropped from the system, since they each contain a CISP that is of the same compound pattern that they are in as a non-suffix, and thus they can never be counted (e.g. GT is included in $AGGTA$ and thus $AGGTA$ will never be counted).

If there are no overlapping patterns in the system and also no CISP that are contained as non-suffixes, then every pattern that occurs will be counted, and the counts using the various counting techniques will coincide.

Table 1 here

3. Computation of the Later Waiting Time Distribution

The development of an auxiliary Markov chain $\{Y_t\}_{t=0}^{\infty}$ that simplifies the computation of the later waiting time distribution for the generalized later pattern in $X_1, X_2, \ldots$ is now discussed, and formulas for computing the later waiting time distribution are then given. The auxiliary chain $\{Y_t\}_{t=0}^{\infty}$ will be such that there is a one-to-one correspondence between classes of its states and states of $\{X_t\}_{t=-m+1}^{\infty}$, so that probabilities may be computed through $\{Y_t\}_{t=0}^{\infty}$ using basic properties of Markov chains. A detailed example is used to illustrate formation of the auxiliary Markov chain.

3.1 Setup of Markov chain $\{Y_t\}_{t=0}^{\infty}$

In general, the transient states of the state space S_Y of $\{Y_t\}_{t=0}^{\infty}$ are vectors of the form $Y_t = \left[a_{t,1}, b_{t,1}, \ldots, a_{t,c}, b_{t,c}, (x_{t-m+1}, \ldots, x_t)\right]$. The last component, the m-tuple $\tilde{x}_t \equiv (x_{t-m+1}, \ldots, x_t)$, gives the last m values of the sequence $\{X_t\}_{t=-m+1}^{\infty}$ at time t. This component must be included to embed the m-th order Markovian sequence $\{X_t\}_{t=-m+1}^{\infty}$ into a Markov chain. The other components are pairs $\left(a_{t,j}, b_{t,j}\right)$ for each compound pattern $\Lambda^{(j)}$, $j = 1, \ldots, c$. Here $b_{t,j}$ is the number of counts of compound pattern $\Lambda^{(j)}$ by time t. Also, $a_{t,j}$ denotes a prefix of E_j that keeps track of progress into the simple patterns of compound pattern $\Lambda^{(j)}$. The state space S_Y also contains an absorbing state α that denotes that all of the compound patterns have been counted their prescribed number of times.

We now consider the automation of the setup of the state space S_Y of $\{Y_t\}_{t=0}^{\infty}$, and its cardinality $|S_Y|$. A simple but inefficient approach to setting up S_Y would be to allow it to have $\beta\delta\lambda + 1$ states, where β is the total number of possible pattern-progress components $a_{t,j}$,

$j = 1,\ldots,c$ for the c compound patterns, δ is the total number of possible values for pattern counts $b_{t,j}$, and λ is the number of possible m-tuples $(x_{t-m+1},\ldots,x_t)$, with β, δ, and λ being determined in isolation of each other. The one in $\beta\delta\lambda + 1$ is added for the absorbing state. Note that $\delta = \left(\prod_{j=1}^{c}(r_j + 1)\right) - 1$ (pattern $\Lambda^{(j)}$ can occur $0,1,\ldots,r_j$ times, $j = 1,\ldots,c$, with the exception that not all patterns can have occurred r_j times, since that case is represented by the absorbing state α). Also, $\lambda = |S_X|^m$ and $\beta = \prod_{j=1}^{c}\left(|E_j| + 1\right)$, the one being added to account for the possibility of no pattern progress for simple patterns of the respective compound patterns, a case that is denoted by $\varnothing$ in this text. Allowing all of the $\beta\delta\lambda$ transient states in the state space of the auxiliary chain is inefficient because many of the states have probability zero of occurring. For efficient computation, we seek a state space that leaves out non-realizable states.

We call vector state components *consistent* if no state component contradicts another, i.e. if they can occur simultaneously. The "minimal" state space that we seek is one that has only consistent components, since states with components that are not consistent can never occur. In what follows we give more specifics on what consistency implies for the various component types, and subsequently how consistent components are determined. We first concentrate on determining sets $\Omega(\tilde{x}_t) \subset (E_1 \cup \varnothing) \times (E_2 \cup \varnothing) \times \ldots \times (E_c \cup \varnothing)$ of consistent pattern-progress components $(a_{t,1},\ldots,a_{t,c})$ as a function of $\tilde{x}_t$. For notation purposes, let $\Omega_j(\tilde{x}_t)$, $j = 1,\ldots,c$ be minimal sets of prefixes of $E_j \cup \varnothing$ that are necessary for computing the later waiting time distribution, sets containing the values that the $a_{t,j}$ can be.

3.1.1. *Determining sets $\Omega(\tilde{x}_t)$ for pattern progress*

The steps below are carried out for each m-tuple $\tilde{x}_t$ to populate sets $\Omega(\tilde{x}_t)$ of pattern-progress components that are required to compute the later waiting time distribution. For the pattern-progress components $(a_{t,1},\ldots,a_{t,c})$ to be consistent with each other, if a_{t,j_1} and a_{t,j_2} are of the same length, then they must be the same, whereas if a_{t,j_1} is shorter than a_{t,j_2}, then a_{t,j_1} must be a suffix of a_{t,j_2} (with the understanding that a pattern-progress component equal to $\varnothing$ is consistent with all other $a_{t,j}$). For an $a_{t,j}$ to be consistent with $\tilde{x}_t$, one must be a suffix of the other.

Overlapping counting. Under overlapping counting, only steps (1) and (2) below are needed to determine $\Omega(\tilde{x}_t)$.

(1) In this step, exactly one value $a_{t,j}$ is entered into each $\Omega_j(\tilde{x}_t)$. The entered value is the longest suffix of $\tilde{x}_t$ that is also an element of E_j (if no suffix of $\tilde{x}_t$ is an element of E_j, then $a_{t,j} = \varnothing$). Note that we set $a_{t,j} = \varnothing$ for a compound pattern $\Lambda^{(j)}$ that has been counted r_j times. The c-tuple $(a_{t,1},\ldots,a_{t,c})$ that results by taking the element of each $\Omega_j(\tilde{x}_t)$ determined in this step is included in $\Omega(\tilde{x}_t)$. Consistency of the c pattern-progress values $a_{t,j}$, $j = 1,\ldots,c$ follows since they are all suffixes of $\tilde{x}_t$.

(2) If $\tilde{x}_t$ is completely included in one or more simple patterns of $\Lambda^{(j)}$, then any element of E_j that ends with $\tilde{x}_t$ is added to $\Omega_j(\tilde{x}_t)$ (in addition to the one element of $\Omega_j(\tilde{x}_t)$ from step (1) above). The new states that are added to $\Omega(\tilde{x}_t)$ during this step are combinations that are formed

by taking any new elements $a_{t,j}$ of $\Omega_j(\tilde{x}_t)$ and combining them with consistent elements of the other $\Omega_{j'}(\tilde{x}_t)$, $j' \neq j$. We note that in all cases, we only add an $a_{t,j}$ to $\Omega_j(\tilde{x}_t)$ or to $\Omega(\tilde{x}_t)$, and thus a state(s) to S_Y, if it is not already there.

Here and in the remainder of the paper, we use the compound patterns $\Lambda^{(1)} = (AGGTA, CG, GT, C)$, $\Lambda^{(2)} = (CGTT, AG, GT, GT)$, and $\Lambda^{(3)} = (AT, CT, GT, TT)$ that were introduced in Section 2.2 to illustrate the steps. The state space $S_X = \{A, C, G, T\}$ (possible values of DNA nucleotides) for the X_t's. We set $m = 3$ and $r_1 = 2$, $r_2 = 3$, $r_3 = 2$, and thus $\Psi = \left\{\Lambda^{(1)}, 2, \Lambda^{(2)}, 3, \Lambda^{(3)}, 2\right\}$.

The respective prefix sets for $\Lambda^{(1)}$, $\Lambda^{(2)}$ and $\Lambda^{(3)}$ and overlapping counting method are $E_1 = \left\{A, AG, AGG, AGGT, C, G\right\}$, $E_2 = \left\{C, CG, CGT, A, G\right\}$, and $E_3 = \left\{A, C, G, T\right\}$. The pattern-progress components corresponding to the 64 possible 3-tuples (x_{t-2}, x_{t-1}, x_t) for this example and both overlapping and non-overlapping counting are listed in Tables 2a-2d of the Appendix. For the tables, we assume that none of the simple patterns of the system have occurred prior to the m-tuple. Here we discuss the computation of pattern progress for $(x_{t-2}, x_{t-1}, x_t) = (G, G, T)$. The suffixes of (G, G, T) are GGT itself, GT and T. None of these are elements of E_1 or E_2, and only T is in E_3, and thus during step (1), $a_{t,1} = a_{t,2} = \varnothing$ are entered into $\Omega_1(G, G, T)$ and $\Omega_2(G, G, T)$, and $a_{t,3} = T$ is entered into $\Omega_3(G, G, T)$, and thus $(\varnothing, \varnothing, T) \in \Omega(G, G, T)$. In step (2), since GGT is completely included in $AGGTA$ and is a suffix of the element $AGGT \in E_1$, $AGGT$ is added to $\Omega_1(G, G, T)$. We combine $AGGT$ with the elements of $\Omega_2(G, G, T)$ and $\Omega_3(G, G, T)$ to form $(AGGT, \varnothing, T)$, which is needed in $\Omega(G, G, T)$ since its components are consistent. The other 15 3-tuples of Table 2c require no elements to be added to an

$\Omega_j(x_{t-2}, x_{t-1}, x_t)$ in step (2) when counting is overlapping, and thus for them, $\Omega(x_{t-2}, x_{t-1}, x_t)$ contains only one entry.

Non-Overlapping counting. When counting is non-overlapping there are additional considerations when forming pattern-progress components $a_{t,j}$, since counting re-starts with a completed (and counted) simple pattern. In this case, the determination of pattern-progress components $a_{t,j}$ depends on whether or not the m-tuple $\tilde{x}_t$ contains any CISP of Ψ (SWNO counting) or of $\Lambda^{(j)}$ (WPNO counting), and those two cases are discussed below.

Note that whenever counting is SWNO, one only needs to know one pattern-progress component for the whole system, and not c of them. Though this doesn't represent a reduction in the number of states in S_Y, it does mean that there are less vector components in the states, reducing storage.

When there are no CISP in $\tilde{x}_t$

If there are no CISP of $\Lambda^{(j)}$ (WPNO counting) or of Ψ (SWNO counting) in $\tilde{x}_t$, then all components for pattern progress determined by steps (1) and (2) above for overlapping counting are needed in $\Omega(\tilde{x}_t)$. We do the following additional step to search for pattern-progress components that need to be added:

(3) If a prefix of $\tilde{x}_t$ (or the entire m-tuple) of the form $x_{t-m+1}, \ldots, x_\upsilon$ ($t-m+1 \leq \upsilon \leq t$) is an element of F_j under WPNO counting, or of F when counting is SWNO, then a pattern could end at x_υ to re-start counting. In that case we apply step (1) above to the remainder $x_{\upsilon+1}, \ldots, x_t$ of

$\tilde{x}_t$ after x_υ to determine an $a_{t,j}$ to be added to $\Omega_j(\tilde{x}_t)$ under WPNO counting, and to $\Omega(\tilde{x}_t)$ under SWNO counting (with $\varnothing$ added if $\upsilon = t$). Under WPNO counting, $\Omega(\tilde{x}_t)$ is then formed as in step (2), by taking all consistent combinations formed by combining any new $a_{t,j}$ of $\Omega_j(\tilde{x}_t)$ with a consistent pattern progress component from each $\Omega_{j'}(\tilde{x}_t)$, $j' \neq j$.

When there are CISP in $\tilde{x}_t$

Let λ_1 and λ_2 be integers with $1 \le \lambda_1 \le \lambda_2 \le m$. A simple pattern $\Lambda_* = (x_{t-m+\lambda_1}, \ldots, x_{t-m+\lambda_2})$ of $\Lambda^{(j)}$ (WPNO counting), or of Ψ (SWNO counting) that is completely contained in $\tilde{x}_t$ means that either Λ_* is counted at $x_{t-m+\lambda_2}$, or another pattern is counted somewhere in $(x_{t-m+\lambda_1}, \ldots, x_{t-m+\lambda_2-1})$ to re-start counting. In either case, a pattern counted after the beginning of $\tilde{x}_t$ re-starts counting, rendering steps (1) and (2) as unnecessary.

(4) In this case, we search $\tilde{x}_t = (x_{t-m+1}, \ldots, x_t)$ for the included simple patterns, beginning with its left-most symbol x_{t-m+1}. Whenever a CISP of the form $(x_{t-m+\lambda_1}, \ldots, x_{t-m+\lambda_2})$ is located, we repeat the search for included simple patterns in the remainder of $\tilde{x}_t$ ($x_{t-m+\lambda_2+1}, \ldots, x_t$), until no more CISP are found. Denote by $t-m+\lambda_2'$ the ending location of the last CISP that is found in $\tilde{x}_t$, $1 \le \lambda_2' \le m$. Then, pattern-progress components are added to $\Omega_j(\tilde{x}_t)$ under WPNO counting, or to $\Omega(\tilde{x}_t)$ under SWNO counting, by applying step (1) above to $x_{t-m+\lambda_2'+1}, \ldots, x_t$, (or $\varnothing$ is added if $\lambda_2' = m$). In addition, if a segment of $\tilde{x}_t$ of the form $x_{t-m+1}, \ldots, x_\upsilon$ ($t-m+1 \le \upsilon \le t$) is an element of F_j (WPNO counting) or of F (SWNO counting), and $x_{t-m+1}, \ldots, x_\upsilon$ ends in the middle of the

first (leftmost) CISP $(x_{t-m+\lambda_1},\ldots,x_{t-m+\lambda_2})$ of $\tilde{x}_t$ $(t-m+\lambda_1 \le \nu < t-m+\lambda_2)$, then counting could be re-started at x_υ, and thus we repeat the search for CISP in $x_{\upsilon+1},\ldots,x_t$, applying step (1) after the last CISP is found.

To illustrate these steps for non-overlapping counting, consider again the compound patterns $\Psi=\left\{\Lambda^{(1)},2,\Lambda^{(2)},3,\Lambda^{(3)},2\right\}$, with $\Lambda^{(1)}=(AGGTA,CG,GT,C)$, $\Lambda^{(2)}=(CGTT,AG,GT,GT)$, $\Lambda^{(3)}=(AT,CT,GT,TT)$, and $m=3$. As noted earlier, $AGGTA$, $CGTT$, and CG may be deleted from the system because they contain CISP as non-suffixes. The suffix sets for the reduced system are then $F_1=\{T,GT,C\}$, $F_2=\{G,AG,T,GT\}$, and $F_3=\{T,AT,CT,GT,TT\}$, pattern endings for the respective compound patterns, and F is the union of these three sets. We consider pattern progress for $\tilde{x}_t=(G,T,T)$. Each compound pattern contains the CISP GT as a non-suffix, and compound pattern $\Lambda^{(3)}$ also has TT as a CISP. Proceeding to search GTT for CISP from left to right, GT is located. If GT is counted, then pattern reckoning re-starts (TT is not counted), and the portion of GTT after the last CISP (the T) is checked for pattern progress, leading to $\left(a_{t,1},a_{t,2},a_{t,3}\right)=\left(\varnothing,\varnothing,T\right)$ being placed in $\Omega(\tilde{x}_t)$ as pattern progress under WPNO counting (T for SWNO counting), since the simple pattern TT of $\Lambda^{(3)}$ is the only pattern that begins with T. We then check for possible additional pattern-progress components by looking for the possibility that patterns can end in the middle of the left-most CISP of GTT, rendering the CISP to not be counted. A simple pattern of $\Lambda^{(2)}$ may end after the G of GTT to re-starting counting for $\Lambda^{(2)}$, however the pattern progress from the remaining part of GTT after the G is still $\varnothing$, and thus under WPNO counting there are no components to add to $\Omega(\tilde{x}_t)$. However, under SWNO counting, the occurrence of a simple pattern of $\Lambda^{(2)}$ does re-start counting for $\Lambda^{(3)}$, and thus in the possible case where a pattern ends at G, the CISP TT of $\Lambda^{(3)}$ would be counted, rendering the need to add $\varnothing$ to $\Omega(\tilde{x}_t)$, since there would be no pattern progress after GTT if the TT is counted.

For the 3-tuple $\tilde{x}_t = (G, A, A)$, there are no CISP, and a pattern that ends after the G does not change pattern progress (step 3), i.e. pattern progress is the same for AA as for the complete 3-tuple GAA. The change in pattern progress for (G, A, A) under non-overlapping counting as compared to overlapping counting using steps (1) and (2) is then due to the deletion of $AGGTA$ from $\Lambda^{(1)}$.

3.1.2 Determining values of $b_{t,j}$ that are necessary

We now give a method to determine pattern counts $b_{t,j}$, $j = 1, \ldots, c$ associated with pattern-progress components $\Omega(\tilde{x}_t)$ and m-tuples $\tilde{x}_t$ in vector states of S_Y. One way that the number of pattern counts needed may be reduced from $\delta = \left(\prod_{j=1}^{c}(r_j + 1)\right) - 1$ is if multiple patterns must be counted simultaneously. Another is when m-tuples contain CISP of Ψ, which ensures that at least one simple pattern has occurred.

In general, for compound pattern $\Lambda^{(j)}$, $1 \le j \le c$ and its distinct simple patterns $\Lambda_i = (\Lambda_{i,1}, \ldots, \Lambda_{i,k_i})$ of length k_i, define the coincidence number

$$\kappa_{j,i} = \left|\{\Lambda_\omega : (\Lambda_{\omega,1}, \ldots, \Lambda_{\omega,k_\omega}) = (\Lambda_{i,k_i - k_\omega + 1}, \ldots, \Lambda_{i,k_i}),\ 1 \le \omega \le \eta_j\}\right| \quad (k_i \ge k_w),$$

the number of simple patterns of $\Lambda^{(j)}$ that are suffixes of Λ_i, and thus will be counted when Λ_i is counted. As Λ_i is always a suffix of itself, $\kappa_{j,i} \ge 1$. The number $\kappa_{j,i} > 1$ if Λ_i is repeated in $\Lambda^{(j)}$, or if it contains a CISP of $\Lambda^{(j)}$ as its suffix. Now, the possible counts of $\Lambda^{(j)}$ needed in components of states are those from the set of values $\chi_j = \left\{\xi_j\right\}$, where $\xi_j \le r_j$ are formed by taking $\xi_j = \sum_i d_{j,i}\kappa_{j,i}$, with the sum being taken over the distinct simple patterns of $\Lambda^{(j)}$, and $d_{j,i} \in \left\{0,1,\ldots,\left\lfloor r_j / \kappa_{j,i} \right\rfloor\right\}$ ($\lfloor \varepsilon \rfloor$ denotes the greatest integer less than or equal to ε). If r_j is not included in the set χ_j then it is added to indicate that the required number of pattern counts have been realized.

Notice that whenever one of the coincidence numbers $\kappa_{j,i}$ is equal to one, then $\chi_j = \left\{0,1,\ldots,r_j\right\}$, and thus no reduction in the number of necessary values of $b_{t,j}$ is obtained based on coinciding occurrences for patterns of $\Lambda^{(j)}$. This is the case, e.g., for $\Lambda^{(1)}$, $\Lambda^{(2)}$, and $\Lambda^{(3)}$ from above, which each have at least one simple pattern with $\kappa_{j,i} = 1$. This is also the case for the compound pattern $\Lambda^{(4)} = \left\{T, AT, AT, TAT, TAT\right\}$, for which the simple pattern T has coincidence number equal to one. However, for $\Lambda^{(5)} = \left\{T, T, AT, AT, TAT, TAT\right\}$, with distinct simple patterns $\Lambda_1 = TAT$, $\Lambda_2 = AT$, and $\Lambda_3 = T$, and respective coincidence numbers $\kappa_{5,1} = 6$, $\kappa_{5,2} = 4$, and $\kappa_{5,3} = 2$, if $r_5 = 13$ then $\chi_5 = \left\{0,2,4,6,8,10,12,13\right\}$.

Coincidence numbers and sets χ_j, $j = 1,\ldots,c$ are helpful for determining the values of $\left(b_{t,1},\ldots b_{t,c}\right)$ that are possible and thus required to be associated with m-tuples and pattern-progress components in the minimal state space. Steps (5) and (6) below are used to determine the counts $b_{t,j}$ associated with the pattern-progress components and m-tuples, for overlapping and non-overlapping counting, respectively.

(5) **Overlapping counting.** For overlapping counting, if an m-tuple has no CISP, then all possible values ξ_j of χ_j are needed as counts $b_{t,j}$ in states. However, if one or more CISP are included in the m-tuple (or in any pattern-progress component that is longer than the m-tuple), then the specific coincidence numbers of CISP patterns need to be added to each value ξ_j of χ_j, with values less than or equal to r_j being retained to obtain the possible values of $b_{t,j}$. (If one CISP is a suffix of another, then one adds the largest of the coincidence numbers to the ξ_j.)

As an example, again consider our sample system Ψ and the 3-tuple GGT, which has the two possible sets of pattern-progress values, $(a_{t,1}, a_{t,2}, a_{t,3}) = (\varnothing, \varnothing, T)$ and $(a_{t,1}, a_{t,2}, a_{t,3}) = (AGGT, \varnothing, T)$. The 3-tuple itself contains the CISP GT, which has coincidence numbers equal to one, two, and one, respectively for $\Lambda^{(1)}$, $\Lambda^{(2)}$, and $\Lambda^{(3)}$. We add those coincidence numbers to $\chi_j = \{0,1,\ldots,r_j\}$, $j = 1,2,3$ to obtain the possible values of $b_{t,j}$ associated with the pattern-progress component $(\varnothing, \varnothing, T)$, and thus $b_{t,1} \in \{1,2\}$, $b_{t,2} \in \{2,3\}$, and $b_{t,3} \in \{1,2\}$. Recalling that not all of the $b_{t,j}$ can equal r_j simultaneously, the total number of possible counts $(b_{t,1}, b_{t,2}, b_{t,3})$ needed for $(\varnothing, \varnothing, T)$ is $(2 \times 2 \times 2) - 1 = 7$. For the pattern-progress component $(AGGT, \varnothing, T)$, we instead search $AGGT$ for CISP. There are two completely included simple patterns, AG and GT. Since AG is in $\Lambda^{(2)}$ along with two copies of GT, we add 3 to all of the numbers $\chi_2 = \{0,1,\ldots,3\}$, and one to the numbers of χ_1 and χ_3 as before, keeping those values that are less than or equal to r_j. Thus $b_{t,1} \in \{1,2\}$, $b_{t,2} = 3$, and $b_{t,3} \in \{1,2\}$, and the total number of possible counts $(b_{t,1}, b_{t,2}, b_{t,3})$ needed for $(ATTG, \varnothing, T)$ based on this analysis is 3, the possibilities being $(b_{t,1}, b_{t,2}, b_{t,3}) = (1,3,1)$, $(1,3,2)$, or $(2,3,1)$. However,

$(b_{t,1}, b_{t,2}, b_{t,3}) = (2,3,1)$ is also not possible for $(ATTG, \varnothing, T)$, since any pattern-progress component $a_{t,j}$ associated with a compound pattern $\Lambda^{(j)}$ that has occurred r_j times is set to $\varnothing$.

(6) **Non-Overlapping counting.** Non-overlapping counting is handled in a manner similar to step (5) above, with an additional consideration. In the case of a state that enters the state space because a suffix of a simple pattern of the system is a prefix of the m-tuple and thus could re-start counting, the largest coincidence number of a simple pattern that could end with the suffix must be added to the values of χ_j, $j = 1, \ldots, c$ as before in step (5), so that the χ_j are all greater than zero. We also note that the number of CISP of the m-tuple that are counted may be less with WPNO than with overlapping counting, and less under SWNO counting than under WPNO counting, leading to a larger number of possible counts.

We illustrate using the 3-tuple GTT, which contains the CISP GT. Under WPNO counting, similar to the case of overlapping counting , adding coincidence numbers of GT gives possible values $b_{t,1} \in \{1,2\}$, $b_{t,2} \in \{2,3\}$, and $b_{t,3} \in \{1,2\}$, for a total number of $(b_{t,1}, b_{t,2}, b_{t,3})$ count values of seven, four associated with $(a_{t,1}, a_{t,2}, a_{t,3}) = (\varnothing, \varnothing, T)$, the other three associated with $(a_{t,1}, a_{t,2}, a_{t,3}) = (\varnothing, \varnothing, \varnothing)$ (when $b_{t,3} = 2$). The difference for WPNO counting is that the simple pattern AG can end at the first G of the 3-tuple and re-starting counting for $\Lambda^{(2)}$ so that GT is not counted. Thus there are four other possible values for counts under WPNO counting, (those with $b_{t,2} = 1$). For SWNO counting, in addition to the seven counts from $b_{t,1} \in \{1,2\}$, $b_{t,2} \in \{2,3\}$, and $b_{t,3} \in \{1,2\}$, since AG could end at G so that GT is not counted, coincidence numbers for AG of $\Lambda^{(2)}$ and TT of $\Lambda^{(3)}$ (but not those of GT) are added to the possible count values of $\Lambda^{(2)}$ and $\Lambda^{(3)}$, respectively, giving $b_{t,1} \in \{0,1,2\}$, $b_{t,2} \in \{1,2,3\}$, and $b_{t,3} \in \{1,2\}$, for an additional

$(3\times 3\times 2)-1=17$ possible values counts (associated with $\varnothing$), and a total of 24 for the 3-tuple. However, some of these are not possible due to interactions among patterns, which will be discussed below.

Pattern interactions. In the above, we have not considered the effect of interactions of simple patterns, which could render other count values $(b_{t,1},\ldots,b_{t,c})$ as impossible. We give an example using our sample patterns Ψ to illustrate the complexity of this phenomenon is as follows. Consider overlapping counting and the state $(A,1,A,0,A,0,(G,A,A))$, which would appear to be a valid state from the above algorithm, but can never be entered due to simple pattern interaction. For $\Lambda^{(1)}$ to be counted only once, neither $AGGTA$ nor CG can have been counted as they both contain CISP. GT also cannot have been counted, since it lies within all of the compound patterns, and $b_{t,j}=0$ for $j>1$. Thus the only simple pattern in $\Lambda^{(1)}$ than might have been counted is C. However the C could not directly precede the 3-tuple GAA since it begins with G, CG is a pattern of $\Lambda^{(1)}$, and $\Lambda^{(1)}$ has been counted only once. As CT is also a simple pattern of Ψ, A is the only possible symbol that can follow the C. Also, since AG and AT are both patterns and C can't have occurred to be counted another time, the only symbol that could possibly follow the A is another A, a contradiction, since a string of A's before the 3-tuple would leave an A before G, forming AG. Thus the state $(A,1,A,0,A,0,(G,A,A))$ is not possible under overlapping counting. However, the state is possible under both types of non-overlapping counting since in that case CG isn't needed in Ψ since it contains C as a non-suffix.

Due to the variety of patterns types that are allowed in Ψ, it would be extremely difficult to give a general description of how to determine which counts are rendered impossible by interactions of states. Thus, we use a different strategy. We set up the state space based on steps (1)-(6) above, and then also form the transition probability matrix as described in the next section. Any state with entrance probabilities all equal to zero (i.e. the corresponding column of the transition probability matrix is all zeroes) is then removed from the state space.

The number of values of the $(b_{t,1}, b_{t,2}, b_{t,3})$ that are needed in states of S_Y (other than initialization states) for the various values of $\tilde{x}_t$ and each counting method are listed in Tables 2a-2d. They are to be compared with $\delta = 35$, the number that would be used with the naïve approach.

Note also that if $b_{t,j} = r_j$ and counting is SWNO, pattern-progress components that enter $\Omega(\tilde{x}_t)$ because of the possibility that some simple pattern of $\Lambda^{(j)}$ is counted to re-start pattern reckoning are no longer needed. For example, under SWNO counting, for the m-tuple (G,T,T), if $\Lambda^{(2)}$ has occurred r_2 times and $\Lambda^{(3)}$ has occurred less than r_3 times, then there is no need to have $\varnothing$ as a component in $\Omega(G,T,T)$, since the occurrence of pattern AG of $\Lambda^{(2)}$ will no longer re-start counting, rendering GT to be counted and not TT.

3.1.3 *Determining initialization states and initial distribution*

The states of S_Y determined above cover time points for which all of the entries of the m-tuple $\tilde{x}_t = (x_{t-m+1}, \ldots, x_t)$ must be considered when searching for patterns. Initialization states may need to be added for time points $t = 0, \ldots, m-1$, when some (or all) of the entries of $\tilde{x}_t$ are before

x_1, and thus do not count, rendering pattern-progress components to possibly be different than those based on the full m-tuple . The most obvious of these states is at time $t=0$, when the complete m-tuple is before x_1, and thus there is no progress towards a pattern. Therefore when $t=0$, for each of the $|S_X|^m$ possible m-tuples $\tilde{x}_0$ and each compound pattern, any pattern-progress component $a_{0,j}$ is set to $\varnothing$, and the number of pattern occurrences $b_{0,j}$ are set to zero. A row vector ξ_0 (of dimension equal to $|S_Y|$) is formed to hold the initial distribution of Y_0, with $\pi(x_{-m+1},\ldots,x_0)$ as its non-zero entries at locations corresponding to the states of the form $(\varnothing,0,\varnothing,0,\varnothing,0,(x_{-m+1},\ldots,x_0))$.

When $m>1$, to handle times $t=1$ to $t=m-1$, initialization states are added (if necessary) to S_Y with pattern-progress components that are determined by applying steps (1)-(4) given above, as appropriate, to $(x_1,\ldots,x_t)$, the part of the m-tuple $\tilde{x}_t$ that is used for pattern reckoning.

In most cases, the states that are added during the initialization stage have components $b_{t,j}$ set to zero. The reason is as follows. Any pattern counted in the initialization stage $(1\le t\le m-1)$ must lie in $(x_1,\ldots,x_t)$, and thus it must lie within an m-tuple. The determination of pattern-progress components $a_{t,j}$ when there is a pattern within an m-tuple is dealt with for times $t\ge m$, as described above. The only time when an additional state with a $b_{t,j}>0$ may be needed during initialization is when a simple pattern of Ψ is completely included in another simple pattern. Then, the length of an $a_{t,j}$ for $t\ge m$ could be longer than a CISP, rendering different pattern progress than for initialization times $1\le t<m-1$ for states with positive entries for a $b_{t,j}$ emanating from the occurrence of the CISP. For example, the 3-tuple (C,G,T) represents progress into the pattern $CGTT$ of $\Lambda^{(2)}$ for times $t\ge 3$. When $t=2$, the C doesn't

count, whereas the CISP GT of $\Lambda^{(2)}$ does. The initialization state that is added to S_Y under overlapping counting has $(a_{t,1}, a_{t,2}, a_{t,3}) = (\varnothing, \varnothing, T)$ as its pattern-progress components, with $b_{2,1} = b_{2,3} = 1$ and $b_{2,2} = 2$, the number of times the compound patterns of Ψ contain GT (no initialization state needs to be added if counting is WPNO).

Any states that are added in the initialization stage are deleted from S_Y after time $t = m$, as they will no longer be needed.

3.1.4 *Determining transition probability matrices of* $\{Y_t\}_{t=0}^{\infty}$

With states formed as described above, probabilities for the state of Y_t are uniquely determined, given the state of Y_{t-1}. For $Y_{t-1} = \left[a_{t-1,1}, b_{t-1,1}, \ldots, a_{t-1,c}, b_{t-1,c}, (x_{t-m}, \ldots, x_{t-1})\right]$, with the occurrence of x_t at time t, the state of Y_t is determined as follows:

- $(x_{t-m}, \ldots, x_{t-1})$ becomes $(x_{t-m+1}, \ldots, x_t)$;
- x_t is added to the end of $a_{t-1,j}$ (overlapping and WPNO counting), and the longest suffix of the resultant that is an element of E_j gives $a_{t,j}$ ($a_{t,j} = \varnothing$ if no suffix of the resultant is in E_j, if $b_{t,j} = r_j$, or if the result of concatenating x_t to $a_{t-1,j}$ is a pattern of $\Lambda^{(j)}$ under WPNO counting);
- $b_{t,j} = \min[b_{t-1,j} + \max_i(\kappa_{j,i}),\ r_j]$ if concatenating x_t to the pattern-progress component renders it as a pattern of $\Lambda^{(j)}$, and $b_{t,j} = b_{t-1,j}$ otherwise, where $\max_i(\kappa_{j,i})$ is the maximum of all coincidence numbers for the simple patterns Λ_i of $\Lambda^{(j)}$ that occur at time t.

For SWNO counting, there is only one pattern-progress component, and $a_{t,j}$ is replaced by the one component for the system, E_j is replaced by E, and $\Lambda^{(j)}$ by Ψ in the discussion above for WPNO counting.

Since the transition of Y_{t-1} to Y_t is determined by the transition of X_{t-1} to X_t in the original sequence, the probability $p(x_t | x_{t-m}, \ldots, x_{t-1})$ is assigned as the entry of the transition probability matrix for the transition from Y_{t-1} to Y_t. For $1 \le t \le m-1$ the transition probability matrix is denoted by T_Y^*, and is of dimension $|S_Y| \times |S_Y|$. When $t \ge m$, since the "initialization states" introduced above are dropped from the state space, the new transition probability matrix T_Y is of dimension $|S_Y - \varsigma| \times |S_Y - \varsigma|$, where ς is the number of initialization states.

We illustrate the setup of the transition probability matrices using our example system Ψ, assuming that $Y_{t-1} = (\varnothing, 0, \varnothing, 0, \varnothing, 0, (A, G, G))$. First, let $t = 1$. There are four possible destinations for Y_0, based on the value of x_1. For overlapping counting, conditional on this value of Y_0, we have:

$$Y_1 = \begin{cases} (A,0,A,0,A,0,(G,G,A)) \text{ with probability } p(A|(A,G,G)) \\ (C,1,C,0,C,0,(G,G,C)) \text{ with probability } p(C|(A,G,G)) \\ (G,0,G,0,G,0,(G,G,G)) \text{ with probability } p(G|(A,G,G)) \\ (\varnothing,0,\varnothing,0,T,0,(G,G,T)) \text{ with probability } p(T|(A,G,G)) \end{cases}.$$

If $x_1 = C$, the pattern C of $\Lambda^{(1)}$ occurs, and thus $b_{1,1} = 1$. If counting is WPNO, $Y_1 = (\varnothing, 1, C, 0, C, 0, (G, G, C))$ with probability $p(C|(A,G,G))$, because counting re-starts when pattern C of $\Lambda^{(1)}$ is counted (and $Y_1 = (\varnothing, 1, 0, 0, (G, G, C))$ under SWNO counting). If $x_1 = T$, though the last two values of (x_{t-2}, x_{t-1}, x_t) are GT, a simple pattern that lies in all three compound patterns, no pattern is counted because the G at time $t = 0$ does not count. These transition probabilities are entered into the appropriate locations of T_Y^* for the initialization stage, since $t < m$. However, if $t \geq m$ and $x_t = T$ so that GT is being counted for the first time, then $Y_t = (\varnothing, 1, \varnothing, 2, T, 1, (G, G, T))$ under overlapping counting, $Y_t = (\varnothing, 1, \varnothing, 2, \varnothing, 1, (G, G, T))$ if counting is WPNO, and the value $p(T | (A, G, G))$ is entered into T_Y in the appropriate location.

3.2 Formulas for the waiting time distribution

We now give a formula to compute waiting time probabilities for the generalized later pattern using basic properties of Markov chains. For overlapping counting, we also show how to compute the distribution through probabilities for competing patterns.

Theorem 3.1. For an m-th order Markovian sequence $\{X_t\}_{t=-m+1}^{\infty}$ and associated auxiliary Markov chain $\{Y_t\}_{t=0}^{\infty}$, for overlapping, WPNO, and SWNO counting, the waiting time distribution of the generalized later pattern is given by:

$$P(L \leq t) = \begin{cases} \xi_0 (T_Y^*)^t U^*(\alpha) & 1 \leq t \leq m-1 \\ \varphi_m (T_Y)^{t-m} U(\alpha) & t \geq m \end{cases}, \tag{3.1}$$

where $U^*(\alpha)$ is a $|S_Y| \times 1$ column vector with a one at the location corresponding to the absorbing state α and zeroes elsewhere, and the $|S_Y - \varsigma| \times 1$ column vector $U(\alpha)$ is the same as $U^*(\alpha)$ except that the entries corresponding to initialization states have been deleted. Also, the $1 \times |S_Y - \varsigma|$ row vector φ_m results by deleting the entries of zero corresponding to initialization states from the product $\xi_0 (T_Y^*)^m$.

Proof. The right-hand side of the equality in (3.1) gives the probability that $Y_t = \alpha$ for times $t \geq 1$. Since there is a one-to-one correspondence between Y_t lying in its absorbing state and the occurrence of the generalized later pattern by time t, the result follows. □

Notice that the vector φ_m for time m serves to initialize the part of the Markov chain $\{Y_t\}_{t=0}^{\infty}$ that travels only through non-initialization states. By the end of the initialization stage the probability of being in any of the initialization states is zero.

Theorem 3.2. For an m-th order Markovian sequence $\{X_t\}_{t=-m+1}^{\infty}$, under overlapping counting, the waiting time distribution of the generalized later pattern may be computed in terms of probabilities for competing patterns through the equation

$$P\left(\bigcap_{i=1}^{c} C_i(t)\right) = \sum_{i=1}^{c} P\left(C_i(t)\right) - \sum_{1\leq i<j\leq c} P\left(C_i(t)\cup C_j(t)\right) + \sum_{1\leq i<j<k\leq c} P\left(C_i(t)\cup C_j(t)\cup C_k(t)\right)$$

$$- \sum_{1\leq i<j<k<l\leq c} P\left(C_i(t)\cup C_j(t)\cup C_k(t)\cup C_l(t)\right) + \cdots + (-1)^{c-1} P\left(\bigcup_{i=1}^{c} C_i(t)\right), \tag{3.2}$$

where $C_j(t)$, $j = 1,\ldots,c$ denotes the event that, by time t, the compound pattern $\Lambda^{(j)}$ occurs r_j times.

Proof Formula 3.2 follows by algebraic manipulation after applying DeMorgan's laws to the analogous result in the case of probabilities for unions of sets (see, e.g., Feller, 1968, pp. 60-62). Under overlapping counting, unions of events $C_i(t)$ correspond to waiting time probabilities of competing patterns, the first time point where a compound pattern is occurs (and is counted) its specified number of times, thus giving the result. □

Notice that throughout the text we have differentiated between patterns occurring and being counted. Theorem 3.2 applies to the occurrence of arbitrary events, and thus to overlapping counting, when all simple patterns that occur are counted. However, when counting is non-overlapping, the occurrence of patterns that are counted on the right hand side (3.2) may not be counted on the left hand side, and thus (3.2) may not be used in that case.

Theorem 3.2 requires the extension of the definition of competing patterns to include cases of multiple repeated simple patterns within the same compound pattern. Though probabilities for these patterns were not dealt with in Aston and Martin (2005), they may be computed by definitions and setups analogous to the ones given above. Note that if the system contains no overlapping patterns and no pattern completely included in a longer pattern as a non-suffix, Theorem 3.2 may be used to compute the generalized later waiting time distribution under non-overlapping counting as well since, in that case, all simple patterns that occur are counted under all counting methods.

3.3. Results for example patterns

Here we give results for the waiting time distribution of the system of patterns $\Psi = \{\Lambda^{(1)}, 2, \Lambda^{(2)}, 3, \Lambda^{(3)}, 2\}$, with $m = 3$. The initial probabilities used were from a discrete uniform initial distribution $\pi(x_{-2}, x_{-1}, x_0)$ (i.e. $\pi(x_{-2}, x_{-1}, x_0) = 1/64$ for each $(x_{-2}, x_{-1}, x_0) \in S_X^3$), and transition probabilities $p(x_t | x_{t-m}, ..., x_{t-1})$ were determined using a random number generator. The transition probabilities used in the example are available from the authors. The waiting time distribution $P(L \le t)$ is depicted in Figure 1, for $t = 1, \ldots, 100$.

[Figure 1 here]

To illustrate the state space reduction attained by searching for pattern-progress components $\Omega(\tilde{x}_t)$ and determining possible occurrence numbers $b_{t,j}$ in the manner described in Section 3.1, $|S_Y|$ = 1644, 1264, or 1129 states under SWNO, WPNO or overlapping counting, respectively, totals that include the respective number of initialization states, ς = 128, 126, and 126. The numbers of states required for each individual m-tuple (not including initialization states) are given in Tables 2a-2d of the Appendix. On the other hand, for this example, $\beta = (7)(6)(5) = 210$, $\delta = (r_1 + 1)(r_2 + 1)(r_3 + 1) - 1 = 35$, and $\lambda = 4^3 = 64$ possible 3-tuples, so that $\beta\delta\lambda + 1 = 470{,}401$, the number of states that would have been needed to compute the distribution using the simple, but very inefficient naïve approach mentioned earlier.

4. Discussion

In this paper the concept of later waiting time distributions is generalized to a system of compound patterns, called generalized later patterns. Generalized later patterns are compound patterns that are each required to be counted pattern-specific numbers of times. Probabilities are computed under overlapping counting and also using two types of non-overlapping counting, within-pattern and system-wide, and a general order of Markovian dependence is assumed. No restrictions are placed on the structure of the simple patterns making up the system. The patterns may be repeated within compound patterns (in effect, weighted to count multiple times) or included in longer patterns of the system. The very general framework, in conjunction with results on competing patterns, encompasses and generalizes previous results for waiting time distributions of patterns.

The very general setup allows the computation of many pattern distributions as special cases. For example, the joint distribution of the numbers of occurrences $N_1(t),\ldots,N_c(t)$ of specific patterns $\Lambda_1,\ldots,\Lambda_c$ in a DNA sequence may be computed through the relationship

$$P(N_1(t)\geq n_1,\ldots,N_c(t)\geq n_c)=P(L\leq t) \ ,$$

where the later waiting time L is defined by allowing each compound pattern $\Lambda^{(j)}$ to be composed of only the simple pattern Λ_j, and setting the number of pattern counts r_j that are required equal to n_j. However, much more generality is built into our setup to increase its applicability.

The later waiting time distribution is computed using an auxiliary Markov chain that has the property that waiting time probabilities of interest correspond to probabilities of the chain lying in its absorbing state. For overlapping counting, it is also shown that the later waiting time distribution may be computed in terms of waiting time probabilities for competing patterns.

Other pattern distributions not derived explicitly by the results of this paper may be computed by using the ideas outlined. For example, Stefanov et al. (2007) computed the waiting time distribution for the number of occurrences of structured motifs $w=\Lambda_1(d_1:d_2)\Lambda_2$, strings of DNA nucleotides with the property that the simple pattern Λ_1 is followed by simple pattern Λ_2, with a gap of between d_1 and d_2 nucleotides between them. In that paper it was assumed that neither of the patterns occur within the gap. In future work we plan to compute the exact distribution of the number of structured motif occurrences with structured motifs being defined in more general terms, with compound patterns replacing simple patterns, and without the assumption of the latter reference. It should also be possible to compute exact waiting time distributions using our setup when clump counting (Stefanov et al., 2007*)* is used, extending the definition so that clumps of compound patterns are allowed. Even waiting time distributions for “competing generalized later patterns” (a collection of generalized later patterns that compete to be the first to occur their pattern-specific number of times) or “generalized later competing patterns” (a collection of competing patterns, each of which must occur pattern-specific numbers of times) may be computed by a simple extension of S_Y.

It is possible to set up the transient states of the state space S_Y in such a manner that the pattern-progress components, number of pattern occurrences, and m-tuples each vary through their individually-determined set of possible values. Though the size of the state space may be easily quantified if this were done, we show that this method is very inefficient because many if not most of those states do not occur because their vector components cannot occur simultaneously. Instead, an algorithm has been presented that determines a minimal set of states, all of which may be visited with positive probability. By the nature of the setup, all of the states determined are necessary for computing the later waiting time distribution. In the example, using the minimal state space resulted in a reduction in the state space by a factor of more than 200, an increase in efficiency that could be very instrumental in the types of problems that the methodology addresses.

Acknowledgment

The authors would like to thank Alison Motsinger-Reif and also the anonymous referees for helpful suggestions concerning the manuscript.

References

Aston, J.A.D., Martin, D.E.K., 2005. Waiting time distributions of competing patterns in higher-order Markovian sequences. J. Appl. Probab. 42, 977-988.

Balakrishnan, N., Koutras, M.V. (2002). Runs and Scans with Applications. John Wiley & Sons, Inc., New York.

Biggins, J.D., Cannings, C., 1987. Markov renewal processes, counters and repeated sequences in Markov chains. Adv. Appl. Probab. 19(3), 521-545.

Cox, D.R., 1955. The analysis of non-Markovian stochastic processes by the inclusion of supplementary variables. Proceedings of the Cambridge Phil. Soc. 51, 433-441.

Ebneshahrashoob, M., Sobel, M., 1990. Sooner and later waiting time problems for Bernoulli trials: frequency and run quotas. Statist. Probab. Lett. 9, 5-11.

Feller, W., 1968. An Introduction to Probability Theory (Vol. 1). John Wiley & Sons, Inc., New York.

Fu, J.C., Koutras, M.V., 1994. Distribution theory of runs: A Markov chain approach. J. Amer. Statist. Assoc. 89, 1050-1058.

Fu, J.C., Chang, Y.M., 2003. On ordered series and later waiting time distributions in a sequence of Markov dependent trials. J. Appl. Probab. 40(3), 623-642.

Hampson, S., Kibler, D., Baldi, P., 2002. Distribution patterns of over-represented k-mers in non-coding yeast DNA. Bioinformatics 18, 513-528.

Kirchhamer, C.V., Yuh, C.-H., Davidson, E.H., 1996. Modular cis-regulatory organization of developmentally expressed genes: Two genes transcribed territorially in the sea urchin embryo, and additional examples. Proc. Natl. Acad. Sci. USA 93, 9322-9328.

Kolev, N.W., Minkova, L.D., 1999a. Run and frequency quotas in a multi-state Markov chain. Commun. Statist. Theory Meth. 28, 2223-2233.

Kolev, N.W., Minkova, L.D., 1999b. Quotas on runs of successes and failures in a multi-state Markov chain. Commun. Statist. Theory Meth. 28, 2235-2248.

Ling, K.D., 1992. A generalization of the sooner and later waiting time problems for Bernoulli trials: frequency quotas. Stat. Probab. Letters 14, 401-405.

Ling, K.D. and Low, T.Y., 1993. On the soonest and the latest waiting time distributions: succession quotas. Commun. Statist. Theory Meth., 22, 2207-2221.

Mariño-Ramírez, L., Spouge, J.L., Kange, G.C and Landsman, D., 2004. Statistical analysis of over-represented words in human promoter sequences. Nucleic Acids Res. 32(3), 949-958.

Pavesi, G., Mauri, G., Pesole, G., 2004. *In silico* representation and discovery of transcription factor binding sites. Briefings in Bioinformatics 5(3), 217-236.

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

Sinha, S., Tompa, M., 2002. Discovery of novel transcription factor binding sites by statistical overrepresentation. Nucleic Acids Res. 30, 5549-5560.

Sumuzin, P., Chen, G., Hata, N., Smith, A.D., Zhang, T., Zhang, M.Q. (2005). DWE: Discriminating Word Enumerator. Bioinformatics 21(1), 31-38.

van Helden, J., Andre, B., Collado-Vides, J., 1998. Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. J. Mol. Biol. 281, 827-842.

van Helden, J., del Olmo, M., Pérez-Ortín, J., 2000. Statistical analysis of yeast downstream sequences reveals putative polyadenylation signals. Nucleic Acids Res. 28, 1000-1010.

Table 1. Simple patterns of $\Lambda^{(1)} = (AGGTA, CG, GT, C)$, $\Lambda^{(2)} = (CGTT, AG, GT, GT)$, and $\Lambda^{(3)} = (AT, CT, GT, TT)$ that are counted in the DNA sequence $CTCGTT$ of length $n = 6$ under the various counting methods, with the locations where they are counted in parenthesis. Whenever a simple pattern is counted, the compound pattern containing it is counted as well.

	Compound Pattern		
Counting method	$\Lambda^{(1)}$	$\Lambda^{(2)}$	$\Lambda^{(3)}$
Overlapping	C (1), C (3), GT (5), CG (4)	GT (5), GT (5), $CGTT$ (6)	CT (2), GT (5), TT (6)
WPNO	C (1), C (3), GT (5)	GT (5), GT (5)	CT (2), GT (5)
SWNO	C (1), C (3), GT (5)	GT (5), GT (5)	GT (5)

Appendix

Table 2a. Pattern-progress components for times $t \geq m$ and $\Psi = \{\Lambda^{(1)}, 2, \Lambda^{(2)}, 3, \Lambda^{(3)}, 2\}$, with $m = 3$, $\Lambda^{(1)} = \{AGGTA, CG, GT, C\}$, $\Lambda^{(2)} = \{CGTT, AG, GT, GT\}$, and $\Lambda^{(3)} = \{AT, CT, GT, TT\}$. Starred entries are those for which at least one of the compound patterns must have occurred its prescribed number of times; in the other entries, it is assumed that no compound pattern has occurred its prescribed number of times. Also given is the total number of states (not including initialization states) associated with each $\tilde{x}_t$ (including those when one or more compound patterns have occurred the prescribed number of times.)

$\tilde{x}_t$	$(a_{t,1}, a_{t,2}, a_{t,3})$ under overlapping counting	$(a_{t,1}, a_{t,2}, a_{t,3})$ under WPNO counting	Components under SWNO counting	Number of states Overlap / WPNO / SWNO
(A,A,A)	(A, A, A)	$(\varnothing, A, A)$	A	35 / 35 / 35
(A,A,C)	(C, C, C)	$(\varnothing, \varnothing, C)$	$\varnothing$	23 / 23 / 31
(A,A,G)	(AG, G, G)	$(G, \varnothing, G)$	$\varnothing$	26 / 26 / 34
(A,A,T)	$(\varnothing, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	$\varnothing$	23 / 23 / 23
(A,C,A)	(A, A, A)	$(\varnothing, A, A)$	A	23 / 23 / 23
(A,C,C)	$(\varnothing, C, C)$ *	$(\varnothing, \varnothing, C)$ *	$\varnothing$ *	11 / 11 / 19
(A,C,G)	$(\varnothing, CG, G)$ *	(G, G, G)	G	11 / 23 / 23
(A,C,T)	$(\varnothing, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	T	15 / 15 / 27
(A,G,A)	(A, A, A)	$(\varnothing, A, A)$	A	26 / 26 / 26
(A,G,C)	(C, C, C)	$(\varnothing, \varnothing, C)$	$\varnothing$	17 / 17 / 23
(A,G,G)	(AGG, G, G)	(G, G, G)	G	26 / 26 / 26
(A,G,T)	$(\varnothing, \varnothing, T)$ *	$(\varnothing, \varnothing, \varnothing)$	T	3 / 11 / 28
(A,T,A)	(A, A, A)	$(\varnothing, A, A)$	A	23 / 23 / 23
(A,T,C)	(C, C, C)	$(\varnothing, \varnothing, C)$	$\varnothing$	15 / 15 / 19
(A,T,G)	(G, G, G)	(G, G, G)	G	23 / 23 / 23
(A,T,T)	$(\varnothing, \varnothing, \varnothing)$ *	$(\varnothing, \varnothing, T)$	T	11 / 23 / 23

* at least one pattern is guaranteed to have occurred its prescribed number of times

Table 2b. Pattern-progress components for times $t \geq m$ and $\Psi = \{\Lambda^{(1)}, 2, \Lambda^{(2)}, 3, \Lambda^{(3)}, 2\}$, with $m = 3$, $\Lambda^{(1)} = \{AGGTA, CG, GT, C\}$, $\Lambda^{(2)} = \{CGTT, AG, GT, GT\}$, and $\Lambda^{(3)} = \{AT, CT, GT, TT\}$. Starred entries are those for which at least one of the compound patterns must have occurred its prescribed number of times; in the other entries, it is assumed that no compound pattern has occurred its prescribed number of times. Also given is the total number of states (not including initialization states) associated with each $\tilde{x}_t$ (including those when one or more compound patterns have occurred the prescribed number of times.)

$\tilde{x}_t$	$(a_{t,1}, a_{t,2}, a_{t,3})$ under overlapping counting	$(a_{t,1}, a_{t,2}, a_{t,3})$ under WPNO counting	Components under SWNO counting	Number of states Overlap / WPNO / SWNO
(C,A,A)	(A, A, A)	$(\varnothing, A, A)$	A	23 / 23 / 23
(C,A,C)	$(\varnothing, C, C)$ *	$(\varnothing, \varnothing, C)$ *	$\varnothing$ *	11 / 11 / 19
(C,A,G)	(AG, G, G)	$(G, \varnothing, G)$	$\varnothing$	17 / 17 / 22
(C,A,T)	$(\varnothing, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	$\varnothing$	15 / 15 / 15
(C,C,A)	$(\varnothing, A, A)$ *	$(\varnothing, A, A)$	A *	11 / 11 / 11
(C,C,C)	$(\varnothing, C, C)$ *	$(\varnothing, \varnothing, C)$ *	C *	11 / 11 / 11
(C,C,G)	$(\varnothing, CG, G)$ *	$(\varnothing, CG, G)$ *	G *	11 / 11 / 11
(C,C,T)	$(\varnothing, \varnothing, T)$ *	$(\varnothing, \varnothing, \varnothing)$ *	T *	7 / 7 / 15
(C,G,A)	$(\varnothing, A, A)$ *	$(\varnothing, A, A)$	A	11 / 23 / 23
(C,G,C)	$(\varnothing, C, C)$ *	$(\varnothing, \varnothing, C)$ *	$\varnothing$ *	11 / 11 / 19
(C,G,G)	$(\varnothing, G, G)$ *	(G, G, G)	G	11 / 23 / 23
(C,G,T)	$(\varnothing, CGT, T)$ *	$(\varnothing, \varnothing, \varnothing)$ *	$\varnothing$ *	3 / 3 / 3
(C,T,A)	(A, A, A)	$(\varnothing, A, A)$	A	15 / 15 / 23
(C,T,C)	$(\varnothing, C, C)$ *	$(\varnothing, \varnothing, C)$ *	$\varnothing$ *	7 / 7 / 19
(C,T,G)	(G, G, G)	(G, G, G)	G	15 / 15 / 23
(C,T,T)	$(\varnothing, \varnothing, \varnothing)$ *	$(\varnothing, \varnothing, T)$	$\varnothing$	7 / 15 / 19

* at least one pattern is guaranteed to have occurred its prescribed number of times

Table 2c. Pattern-progress components for times $t \geq m$ and $\Psi = \{\Lambda^{(1)}, 2, \Lambda^{(2)}, 3, \Lambda^{(3)}, 2\}$, with $m = 3$, $\Lambda^{(1)} = \{AGGTA, CG, GT, C\}$, $\Lambda^{(2)} = \{CGTT, AG, GT, GT\}$, and $\Lambda^{(3)} = \{AT, CT, GT, TT\}$. Starred entries are those for which at least one of the compound patterns must have occurred its prescribed number of times; in the other entries, it is assumed that no compound pattern has occurred its prescribed number of times. Also given is the total number of states (not including initialization states) associated with each $\tilde{x}_t$ (including those when one or more compound patterns have occurred the prescribed number of times.)

$\tilde{x}_t$	$(a_{t,1}, a_{t,2}, a_{t,3})$ under overlapping counting	$(a_{t,1}, a_{t,2}, a_{t,3})$ under WPNO counting	Components under SWNO counting	Number of states Overlap / WPNO / SWNO
(G,A,A)	(A, A, A)	$(\varnothing, A, A)$	A	34 / 35 / 35
(G,A,C)	(C, C, C)	$(\varnothing, \varnothing, C)$	$\varnothing$	23 / 23 / 31
(G,A,G)	(AG, G, G)	$(G, \varnothing, G)$	$\varnothing$	25 / 26 / 34
(G,A,T)	$(\varnothing, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	$\varnothing$	22 / 23 / 23
(G,C,A)	(A, A, A)	$(\varnothing, A, A)$	A	23 / 23 / 23
(G,C,C)	$(\varnothing, C, C)$ *	$(\varnothing, \varnothing, C)$ *	$\varnothing$ *	11 / 11 / 19
(G,C,G)	$(\varnothing, CG, G)$ *	(G, G, G)	G	11 / 23 / 23
(G,C,T)	$(\varnothing, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	T	15 / 15 / 27
(G,G,A)	(A, A, A)	$(\varnothing, A, A)$	A	34 / 35 / 35
(G,G,C)	(C, C, C)	$(\varnothing, \varnothing, C)$	$\varnothing$	23 / 23 / 31
(G,G,G)	(G, G, G)	(G, G, G)	G	34 / 35 / 35
(G,G,T)	$(\varnothing, \varnothing, T)$, $(AGGT, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	$\varnothing$	9 / 7 / 7
(G,T,A)	(A, A, A)	$(\varnothing, A, A)$	A	7 / 11 / 26
(G,T,C)	$(\varnothing, C, C)$ *	$(\varnothing, \varnothing, C)$ *	$\varnothing$	3 / 5 / 23
(G,T,G)	(G, G, G)	(G, G, G)	G	7 / 11 / 26
(G,T,T)	$(\varnothing, \varnothing, \varnothing)$ *	$(\varnothing, \varnothing, T)$	$T, \varnothing$	3 / 11 / 21

* at least one pattern is guaranteed to have occurred its prescribed number of times

Table 2d. Pattern-progress components for times $t \geq m$ and $\Psi = \{\Lambda^{(1)}, 2, \Lambda^{(2)}, 3, \Lambda^{(3)}, 2\}$, with $m = 3$, $\Lambda^{(1)} = \{AGGTA, CG, GT, C\}$, $\Lambda^{(2)} = \{CGTT, AG, GT, GT\}$, and $\Lambda^{(3)} = \{AT, CT, GT, TT\}$. Starred entries are those for which at least one of the compound patterns must have occurred its prescribed number of times; in the other entries, it is assumed that no compound pattern has occurred its prescribed number of times. Also given is the total number of states (not including initialization states) associated with each $\tilde{x}_t$ (including those when one or more compound patterns have occurred the prescribed number of times.)

$\tilde{x}_t$	$(a_{t,1}, a_{t,2}, a_{t,3})$ under overlapping counting	$(a_{t,1}, a_{t,2}, a_{t,3})$ under WPNO counting	Components under SWNO counting	Number of states Overlap / WPNO / SWNO
(T,A,A)	(A, A, A)	$(\varnothing, A, A)$	A	24 / 24 / 35
(T,A,C)	(C, C, C)	$(\varnothing, \varnothing, C)$	$\varnothing$	16 / 16 / 31
(T,A,G)	(AG, G, G)	$(G, \varnothing, G)$	$\varnothing$	18 / 18 / 34
(T,A,T)	$(\varnothing, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	$\varnothing$	12 / 12 / 23
(T,C,A)	(A, A, A)	$(\varnothing, A, A)$	A	16 / 16 / 23
(T,C,C)	$(\varnothing, C, C)$ *	$(\varnothing, \varnothing, C)$ *	$\varnothing$ *	8 / 8 / 19
(T,C,G)	$(\varnothing, CG, G)$ *	(G, G, G)	G	8 / 16 / 23
(T,C,T)	$(\varnothing, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	T	8 / 8 / 27
(T,G,A)	(A, A, A)	$(\varnothing, A, A)$	A	24 / 24 / 35
(T,G,C)	(C, C, C)	$(\varnothing, \varnothing, C)$	$\varnothing$	16 / 16 / 31
(T,G,G)	(G, G, G)	(G, G, G)	G	24 / 24 / 35
(T,G,T)	$(\varnothing, \varnothing, T)$	$(\varnothing, \varnothing, \varnothing)$	$\varnothing$	4 / 4 / 7
(T,T,A)	(A, A, A)	$(\varnothing, A, A)$	A	12 / 23 / 23
(T,T,C)	(C, C, C)	$(\varnothing, \varnothing, C)$	$\varnothing$	8 / 15 / 19
(T,T,G)	(G, G, G)	(G, G, G)	G	12 / 23 / 23
(T,T,T)	$(\varnothing, \varnothing, \varnothing)$ *	$(\varnothing, \varnothing, T), (\varnothing, \varnothing, \varnothing)$	$T, \varnothing$	11 / 12 / 23

* at least one pattern is guaranteed to have occurred its prescribed number of times

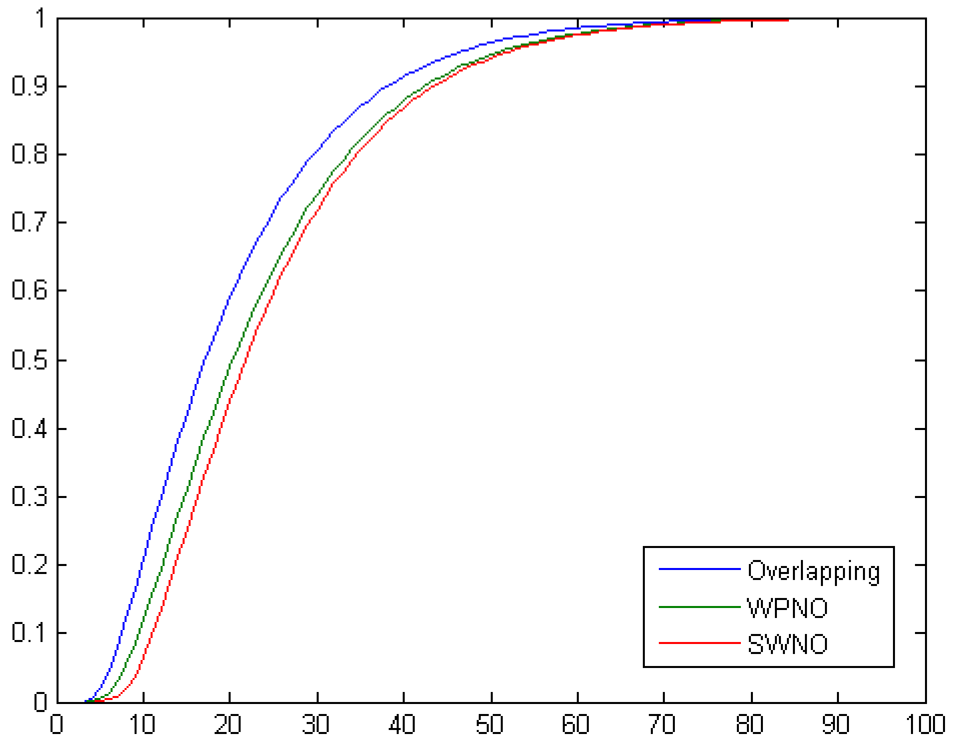

Figure 1: Later waiting time probabilities $P(L \le t)$, for $t = 1,\ldots,100$. The probabilities are associated with the system $\Psi = \{\Lambda^{(1)}, 2, \Lambda^{(2)}, 3, \Lambda^{(3)}, 2\}$, for a third-order Markovian sequence with random transition probabilities and a discrete uniform initial distribution.