



HAL
open science

SeqROCTM: A Matlab toolbox for the analysis of Sequence of Random Objects driven by Context Tree Models

Noslen Hernández, Aline Duarte

► **To cite this version:**

Noslen Hernández, Aline Duarte. SeqROCTM: A Matlab toolbox for the analysis of Sequence of Random Objects driven by Context Tree Models. 2024. hal-04801683

HAL Id: hal-04801683

<https://hal.science/hal-04801683v1>

Preprint submitted on 25 Nov 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

SeqROCTM: A Matlab toolbox for the analysis of Sequence of Random Objects driven by Context Tree Models

Noslen Hernández

Department of Statistics, University of São Paulo
and

Aline Duarte *

Department of Statistics, University of São Paulo

July 23, 2021

Abstract

In several research problems we deal with probabilistic sequences of inputs (e.g., sequence of stimuli) from which an agent generates a corresponding sequence of responses and it is of interest to model the relation between them. A new class of stochastic processes, namely *sequences of random objects driven by context tree models*, has been introduced to model such relation in the context of auditory statistical learning. This paper introduces a freely available Matlab toolbox (SeqROCTM) that implements this new class of stochastic processes and three model selection procedures to make inference on it. Besides, due to the close relation of the new mathematical framework with context tree models, the toolbox also implements several existing model selection algorithms for context tree models.

Keywords: context tree, stochastic process, functional data, neurobiology, context tree model, statistical learning

Mathematical subject classification: 62M05, 60K99, 68V35, 92-04, 90C99

*This work is part of University of São Paulo project *Mathematics, computation, language and the brain*, FAPESP project *Research, Innovation and Dissemination Center for Neuromathematics* (grant 2013/07699-0). Author N. Hernández was fully supported by FAPESP fellowship 2016/22053-7.

1 Introduction

In several research problems we deal with probabilistic sequences of inputs (e.g., sequence of stimuli) from which an agent generates a corresponding sequence of responses and it is of interest to model or find some kind of relation between them. This is the case, for example, of many experiments related to the study of statistical learning in neuroscience. Statistical learning in this context refers to the ability to extract statistical regularities from the environment over time (Armstrong *et al.*, 2017; Conway, 2020; Schapiro and Turk-Browne, 2015). In this kind of experiments, humans or animals are exposed to sequences of stimuli with some statistical regularity and it is conjectured that the brain is able to retrieve/learn the statistical regularities encoded in the stimuli (von Helmholtz, 1867; Wacongne *et al.*, 2012; Garrido *et al.*, 2013). The study of this conjecture usually involves data analysis of some physiological or behavioral responses recorded from participants during the performance of a suitable task. Statistical learning is also a widely used term in computer science but it is not that meaning we are referring to here.

Motivated by an auditory statistical learning problem, a new class of stochastic process was introduced in Duarte *et al.* (2019) to model the relation between the probabilistic sequence of inputs and the corresponding sequence of responses, namely *sequences of random objects driven by context tree models*. A process in this class has two elements: a stochastic source generating the sequence of stimuli and a sequence of responses generated by the agent during the exposure to the stimuli. The source generating the stimuli is assumed to be a *context tree model* (CTM) taking values in a categorical set (Rissanen, 1983; Bühlmann and Wyner, 1999; Galves and Löcherbach, 2008). In a context tree model, at each step, the occurrence of the next symbol is solely determined by a variable-length sequence of past symbols, called *context*. Any stationary stochastic chain can be approximate by a context tree model, therefore it constitutes a powerful tool to model the probabilistic structure of sequences of stimuli (Fernández and Galves, 2002; Duarte *et al.*, 2006). The model also assumes the existence of a family of probability measures on the set of responses (hereafter called objects) indexed by the contexts characterizing the sequence of stimuli. This family of measures describes the relation between the source and the responses. More

precisely, at each step, a new object of the response sequence is chosen according to the probability measure associated to the context ending at that step in the sequence of stimuli.

Given some data, statistical model selection methods can be applied to estimate the set of contexts and, in some cases, the family of probability measures characterizing the distribution of the response sequence. The theoretical framework introduced in Duarte *et al.* (2019) addressed a model selection algorithm for the particular case in which the objects of the response sequence refer to functions (e.g., electrophysiological data). Nevertheless, the proposed mathematical framework is general enough to model other kind of objects in the response sequences such as responses taking values in a categorical set, in the real line or in \mathbb{R}^n . In the same way that context tree models have found applications in several research areas such as neuroscience, genetics (Busch *et al.*, 2009) and linguistics (Galves *et al.*, 2012), this new theoretical framework can find applications in research areas beyond neuroscience. For this reason, its computational implementation is useful.

This paper introduces the SeqROCTM Matlab toolbox, a set of computational tools for working with sequences of random objects driven by context tree models. In addition to the statistical software, this article makes other contributions. We formalize two model selection procedures for sequences of random objects driven by context tree models with categorical responses. We also formalize and define general conditions for the use of the Smaller Maximizer Criteria (SMC) to tune model selection algorithms for both context tree models and sequence of random objects driven by context tree models.

Since the stimuli sequence is generated by context tree model, the SeqROCTM toolbox also implements several model selection algorithms and tuning procedures for context tree model. Up to our knowledge, there exist only an R package implementing model selection for context tree model (Mächler and Bühlmann, 2004).

The paper is organized as follows. Section 2 briefly describes the mathematical concepts and model selection algorithms included in the SeqROCTM toolbox. Section 3 introduces the architecture of the software. Sections 4 and 5 present the main functionalities of the toolbox through illustrative examples motivated by statistical learning problems. Conclusions are given in Section 6.

2 Model selection methods for sequence of random objects driven by context tree models

We begin this section by introducing some notation and formal definitions. Let A be a finite set. Any string $u = (u_{n-m}, \dots, u_{n-1}) \in A^m$, $1 \leq m \leq n$, is denoted by u_{n-m}^{n-1} and its length by $l(u)$. Given two strings u and v of elements of A , uv denotes the string of length $l(u) + l(v)$ obtained by concatenating u and v . The string u is said to be a *suffix* of v , and denote by $u \preceq v$, if there exists a string s satisfying $v = su$. When $v \neq u$ we say that u is a *proper suffix* of v and denote $u \prec v$.

Definition 1. A *context tree* is defined as any set $\tau \subset A^* = \cup_{m=1}^{\infty} A^m$ satisfying

1. *Suffix Property.* No string $w \in \tau$ is a proper suffix of another string $s \in \tau$.
2. *Irreducibility.* No string belonging to τ can be replaced by a proper suffix without violating the suffix property.

The elements of τ are called *contexts*. The *height* and *size* of τ are defined as $l(\tau) = \max\{l(w) : w \in \tau\}$ and $|\tau|$, respectively.

In this work we deal with experiments in which an agent is exposed to an stochastic sequence of stimulus $(X_n)_n$, taking values in A . A key point is that the stochastic sequence $(X_n)_n$ is a context tree model (Rissanen, 1983; Bühlmann and Wyner, 1999). Formally, consider a stationary ergodic process $(X_n)_{n \geq 1}$, $X_n \in A$, and for any string $s \in A^*$ denote by

$$p(s) = P(X_1^{l(s)} = s).$$

Definition 2. We say $(X_n)_n$ is a *context tree model* with parameters (τ, p) if there exist a function $c_\tau : A^* \rightarrow \tau$ such that

1. for any $n \geq l(\tau)$ and any finite sequence $x_{-n}^{-1} \in A^n$ such that $p(x_{-n}^{-1}) > 0$, it holds that

$$P(X_{n+1} = a | X_1^n = x_{-n}^{-1}) = p(X_{n+1} = a | c_\tau(x_{-n}^{-1})) \text{ for all } a \in A.$$

2. no proper suffix of $c_\tau(x_{-n}^{-1})$ satisfies condition 2.

The function c_τ is said a *context function*.

A sequence of responses $(Y_n)_n$ with values in some measurable space (F, \mathcal{F}) is recorded while the agent is exposed to the stimuli sequence (e.g., neurophysiology responses such as electroencephalographic data, behavioral responses, etc.). The relation between the responses and the stimuli is modeled through a class of stochastic processes called sequence of random objects driven by context tree model (Duarte *et al.*, 2019).

Definition 3. The bivariate stochastic chain $(X_n, Y_n)_n$ taking values in $A \times F$ is a *sequence of random objects driven by context tree models* with parameters (τ, p, q) , where q is a family of probability measures on (F, \mathcal{F}) , if

1. $(X_n)_n$ is a context tree model with parameters (τ, p) ;
2. conditionally to the sequence $(X_n)_n$, $(Y_n)_n$ are independent random variables and, for any $n \geq l(\tau)$, it holds that

$$P(Y_n \in J | X_1^n = x_1^n) = q(Y_n \in J | c_\tau(x_1^n)) \text{ for any } \mathcal{F}\text{-measurable } J.$$

Given some training data $(X_n, Y_n)_n$ and, under the assumption that it was generated by a sequence of random objects driven by context tree models, the statistical problem of interest is to estimate the parameters τ and q characterizing the response sequence.

To formulate the model selection methods we consider two scenarios. A first scenario in which the responses Y_n belong to a finite set and a second scenario in which Y_n takes values in a functional space. Hereafter we will refer to these two scenarios as *categorical* and *functional* case, respectively.

Before introducing the model selection procedures a few more definitions are needed. Given a finite string $u \in A^*$ we denote by $N_n^X(u)$ the number of occurrences of the string u in the sequence (X_1, \dots, X_{n-1}) , that is

$$N_n^X(u) = \sum_{t=l(u)}^{n-1} \mathbb{1}_{\{X_{t-l(u)+1}^t = u\}}. \tag{1}$$

Definition 4. Let L be an integer such that $1 \leq L \leq n$, an *admissible context tree of maximum height L* for the sample (X_1, \dots, X_n) is any context tree τ satisfying

i) $w \in \tau$ if and only if $l(w) \leq L$ and $N_n^X(w) \geq 1$.

ii) Any string $v \in A^*$ with $N_n^X(v) \geq 1$ is a suffix of some $w \in \tau$ or has a suffix $w \in \tau$.

The set of all admissible context trees of maximal height L is denoted by $\Gamma^L(X_1^n)$.

Definition 5. Let τ be a context tree and fix a finite string $s \in A^*$. A *subtree* in τ induced by s is defined as the set $\tau_s = \{w \in \tau : s \prec w\}$. The set τ_s is called a *terminal subtree* if for all $w \in \tau_s$ it holds that $w = as$ for some $a \in A$.

Given two context trees τ_1 and τ_2 , we denote by $\tau_1 \preceq \tau_2$ (resp. \prec) if for any $w \in \tau_1$ there exists $s \in \tau_2$ such that $w \preceq s$ (resp. \prec).

The model selection procedure that will be introduced for the functional case and two out of three procedures developed for the categorical case are inspired by the algorithm Context (Rissanen, 1983). For this reason, we first describe below a general algorithm Context procedure, specifying later the difference on each case. In the sequence, we present a third model selection procedure for the categorical case which is based on BIC.

2.1 General algorithm Context

Given a sample $(X_1, Y_1), \dots, (X_n, Y_n)$, fix an integer $1 \leq L \leq n$ and let $\mathcal{T}_n^L(X_1^n)$ be the context tree of maximum size in $\Gamma^L(X_1^n)$. We shorten this maximal candidate context tree $\hat{\tau} = \mathcal{T}_n^L(X_1^n)$ by successively pruning the terminal subtrees according to some statistical criterion.

For any string $u \in A^*$ such that $\hat{\tau}_u$ is a terminal subtree, we decide to prune or not $\hat{\tau}_u$ verifying a statistical criterion, say $\mathbf{stat}(u)$. If $\mathbf{stat}(u)$ is satisfied, we prune the subtree $\hat{\tau}_u$ in $\hat{\tau}$,

$$\hat{\tau} = (\hat{\tau} \setminus \hat{\tau}_u) \cup \{u\}. \quad (2)$$

Otherwise, if $\mathbf{stat}(u)$ is not satisfied, we keep $\hat{\tau}_u$ in $\hat{\tau}$. At each pruning step, we check a string $s \in A^*$ which induces a terminal subtree in $\hat{\tau}$ and that has not been checked yet. This pruning procedure is repeated until all the existing terminal subtrees have been checked and no more pruning is possible.

A pseudo code for the general algorithm Context is given in Algorithm 1.

Algorithm 1: General algorithm Context for sequences of random objects driven by context tree models

Input: An alphabet A , a sample $(x_1, y_1), \dots, (x_n, y_n)$ with $x_k \in A, y_k \in \mathcal{Y}$ for $1 \leq k \leq n$, a positive integer L .

Output: A context tree $\hat{\tau}$ and a family of distributions \hat{q} (only in the categorical case) indexed by the elements of $\hat{\tau}$.

```

1 Compute  $\mathcal{T}_n^L(x_1^n)$  and initialize  $\hat{\tau} \leftarrow \mathcal{T}_n^L(x_1^n)$ 
2  $\text{Flag}(s) \leftarrow$  “not checked” for all string  $s$  such that  $s \prec w \in \hat{\tau}$ 
3 while  $\exists s \in A^*$ :  $\hat{\tau}_s$  is a terminal subtree and  $\text{Flag}(s) =$  “not checked” do
4   | Compute the statistic criterion  $\text{stat}(s)$ 
5   | if  $\text{stat}(s)$  is satisfied then
6   |   |  $\hat{\tau} \leftarrow (\hat{\tau} \setminus \hat{\tau}_s) \cup \{s\}$ 
7   | else
8   |   |  $\text{Flag}(s) \leftarrow$  “checked”
9   | end
10 end
11 Compute  $\hat{q}$ 
12 return  $\hat{\tau}, \hat{q}$ 

```

The use of the general algorithm Context for the functional and categorical cases differs with respect to $\text{stat}(u)$.

In the functional case $\text{stat}(u)$ implements a two-side goodness of fit test for functional data (Cuesta-Albertos *et al.*, 2006). For the categorical case, $\text{stat}(u)$ implements two different procedures. The first one compares the conditional log-likelihoods between a string and its offspring while the second one compares the empirical distributions between a string and its offspring. The following sections describe the statistical criterion $\text{stat}(u)$ implemented in each case.

2.2 Functional case

In this section it is defined the statistic criterion used in the general algorithm Context (Algorithm 1) for the functional case.

Consider $F = L^2([0, T])$ the set of real-valued square integrable function in the interval $[0, T]$ and \mathcal{F} the Borel σ -algebra on $L^2([0, T])$. Moreover, assume that $(X_1, Y_1), \dots, (X_n, Y_n)$ is a sample of a sequence of random objects driven by context tree models with parameters (τ^*, p^*, q^*) , with q^* a probability measure on (F, \mathcal{F}) . For any string $s \in A^*$ with $l(s) \leq L$, let $I_n(s)$ be the set of indexes belonging to $\{l(s), \dots, n\}$ where the string s occurs in the sample (X_1, \dots, X_n) , that is

$$I_n(s) = \{l(s) \leq m \leq n : X_{m-l(s)+1}^m = s\}. \quad (3)$$

By definition, the set $I_n(s)$ has $N_n^X(s)$ elements. If $I_n(s) = \{m_1, \dots, m_{N_n^X(s)}\}$, we set $Y_k^{(s)} = Y_{m_k}$ for each $1 \leq k \leq N_n^X(s)$. Thus, $Y_1^{(s)}, \dots, Y_{N_n^X(s)}^{(s)}$ is the *subsample of (Y_1, \dots, Y_n) induced by the string s* .

To each element of the terminal subtree $\hat{\tau}_u$ we associate the subsample of (Y_1, \dots, Y_n) induced by it. These sets of functions are used to decide whether the subtree is pruned or not: the `stat(u)` criterion tests the equality of distribution between the sets of functions associated to the elements of $\hat{\tau}_u$.

To test whether two samples of functions have the same distribution, it is used a projective method proposed in Cuesta-Albertos *et al.* (2006). In this method, each function of both sets is projected in a gaussian direction chosen at random. This produces two new projected samples of real numbers. Then, the equality of distribution for these new samples is tested using the Kolmogorov-Smirnov test. The results presented in Cuesta-Albertos *et al.* (2006) guarantees that, if the two samples of functions have different distributions then the set of random directions where the projected samples have the same distribution has Lebesgue measure equal to zero. This implies that if we reject the null hypothesis of equality of distributions for two sets of projected samples, then we can also reject it for the corresponding functional sets.

Formally, for any string $u \in A^*$ such that $\hat{\tau}_u$ is a terminal subtree, we test the null

hypothesis

$$H_0^{(u)} : \mathcal{L} \left(Y_1^{(s)}, \dots, Y_{N_n^X(s)}^{(s)} \right) = \mathcal{L} \left(Y_1^{(v)}, \dots, Y_{N_n^X(v)}^{(v)} \right), \forall s, v \in \hat{\tau}_u, \quad (4)$$

using the test statistic

$$\begin{aligned} \Delta_n(u) = \Delta_n^W(u) &= \max_{s, v \in \hat{\tau}_u} D_n^W \left((Y_1^{(s)}, \dots, Y_{N_n^X(s)}^{(s)}), (Y_1^{(v)}, \dots, Y_{N_n^X(v)}^{(v)}) \right) \\ &= \max_{s, v \in \hat{\tau}_u} \sqrt{\frac{N_n^X(s) N_n^X(v)}{N_n^X(s) + N_n^X(v)}} KS(\hat{Q}_n^{s,W}, \hat{Q}_n^{v,W}). \end{aligned} \quad (5)$$

Here W is a realization of a Brownian bridge in the interval $[0, T]$ and $KS(\hat{Q}_n^{s,W}, \hat{Q}_n^{v,W})$ is the Kolmogorov-Smirnov distance between the empirical distributions $\hat{Q}_n^{s,W}$ and $\hat{Q}_n^{v,W}$ of the projections of the samples $Y_1^{(s)}, \dots, Y_{N_n^X(s)}^{(s)}$ and $Y_1^{(v)}, \dots, Y_{N_n^X(v)}^{(v)}$ onto the direction W , respectively.

We reject the null hypothesis $H_0^{(u)}$ when $\Delta_n(u) > \delta_\alpha$, with $\delta_\alpha = \delta_\alpha(u) = \sqrt{-1/2 \ln(\alpha/2M)}$ and $M = \binom{|\hat{\tau}_u|}{2}$. Observe that for a string u with only one pair of offspring (i.e., $|\hat{\tau}_u| = 2$) the null hypothesis $H_0^{(u)}$ is tested with significance level α (because δ_α becomes the $(1 - \alpha)$ -percentile of a KS distribution). On the other hand, for a string u with more than two offspring, $H_0^{(u)}$ is tested with an unknown significance level upper bounded by α . This is guaranteed because in the definition of δ_α we applied a terminal subtree correction, by using α/M instead of α . In this way, $H_0^{(u)}$ is tested with significance level at most α for any string u .

The statistic in equation (5) depends on the a random direction W . For this reason, to improve the stability of the estimate we test the null hypothesis (4) several times using different Brownian bridges. We conclude that the sets of functions associated to the elements of the terminal subtree $\hat{\tau}_u$ do not have the same distribution, and consequently, we do not prune the subtree $\hat{\tau}_u$, if the number of rejections exceeds a certain threshold. This threshold is derived according to a binomial distribution with probability of success α . Formally, fixed a string $s \in A^*$, a significant level α and consider N independent Brownian bridges W_1, \dots, W_N . Compute the test statistics $\Delta_n^{W_1}, \dots, \Delta_n^{W_N}$ and define

$$\bar{\Delta}_n(s) = \sum_{m=1}^N 1\{\Delta_n^{W_m} > \delta_\alpha\}. \quad (6)$$

In this case, $\text{stat}(u)$ checks whether $\bar{\Delta}_n(s) < C$, where C is the $(1 - \beta)$ -percentile of a binomial distribution with parameters N and α . The elucidation for this procedure comes from the following proposition.

Proposition 1. *For any string $u \in A^*$ and integer $N \geq 1$, consider the random variables $\Delta_n^{W_1}(u), \dots, \Delta_n^{W_N}(u)$ defined in (5) with W_1, \dots, W_N independent Brownian bridges in the interval $[0, T]$. For any significance level $\alpha > 0$ define $\delta_\alpha(u) = \sqrt{-1/2 \ln(\alpha/2M_u)}$ with $M_u = \binom{\tau_u}{2}$. Under the null hypothesis (4), for any $\beta \in (0, 1)$, it holds that,*

$$P\left(\sum_{m=1}^N \mathbb{1}_{\{\Delta_n^{W_m} > \delta_\alpha(u)\}} > C_\beta\right) \leq \beta.$$

where C_β denotes the smallest constant such that $P(\xi > C_\beta) \leq \beta$ with ξ a random variable with Binomial distribution of parameters N and α .

2.3 Categorical case

2.3.1 General algorithm Context + Conditional log-likelihood

Given $u \in A^*$ such that $N_n^X(u) \geq 1$, we denote by $N_n^{XY}(u, a)$ the number of occurrences of the string u in the sample (X_1, \dots, X_n) followed by the occurrence of the symbol a in the sample (Y_1, \dots, Y_n) , that is

$$N_n^{XY}(u, a) = \sum_{t=l(u)}^{n-1} \mathbb{1}_{\{X_{t-l(u)+1}^t = u; Y_{t+1} = a\}}. \quad (7)$$

The maximum conditional likelihood for a sample $(X_1, Y_1), \dots, (X_n, Y_n)$ is given by

$$L_{(\tau, \hat{q})}(Y_1^n | X_1^n) = \prod_{u \in \tau} L_{(u, \hat{q})}(Y_1^n | X_1^n) \quad (8)$$

with

$$L_{(u, \hat{q})}(Y_1^n | X_1^n) = \prod_{a \in A} \hat{q}(a|u)^{N_n^{XY}(u, a)}, \quad (9)$$

and $\hat{q}(a|u)$ the maximum likelihood estimator of the conditional probability $q(a|u)$, defined as

$$\hat{q}(a|u) = \frac{N_n^{XY}(u, a)}{N_n^X(u)} = \frac{N_n^{XY}(u, a)}{\sum_{a' \in A} N_n^{XY}(u, a')}. \quad (10)$$

Notice that $L_{(u,q)}(X_1^n|Y_1^n)$ is the portion of the conditional likelihood $L_{(\tau,q)}(X_1^n|Y_1^n)$ of the model (τ, p, q) given the data $(X, Y)_1^n$ induced by the context $u \in \tau$.

Consider the statistic

$$\Delta_n(u) = \sum_{b \in A} \sum_{a \in A} N_n^{XY}(bu, a) \log \frac{\hat{q}(a|bu)}{\hat{q}(a|u)}, \quad (11)$$

and fix a threshold $\delta > 0$. For any string $u \in A^*$ such that $\hat{\tau}_u$ is a terminal branch, the function $\mathbf{stat}(u)$ verifies whether the following inequality holds

$$\Delta_n(u) < \delta. \quad (12)$$

If the inequality is satisfied, we prune the subtree $\hat{\tau}_u$ of $\hat{\tau}$. Otherwise, we keep $\hat{\tau}_u$ in $\hat{\tau}$. The inequality (12) is equivalent to check whether

$$\sum_{b \in A} \log(L_{(bu,\hat{q})}(Y_1^n | X_1^n)) - \log(L_{(u,\hat{q})}(Y_1^n | X_1^n)) < \delta.$$

Notice that $\Delta_n(u)$ is the conditional log-likelihood ratio between a model with parameters (τ, p, q) and a model with parameters (τ', p, q') , where $\tau \succ \tau'$ and they differ only by one set of offspring nodes branching from u , that is $\tau' = \tau \setminus \tau_u \cup \{u\}$.

Remark 1. The idea of comparing the maximum likelihood induced by a node with the maximum likelihood induced by its offspring was originally used in the algorithm Context introduced by Rissanen (1983).

Given $\mathcal{T}_n^L = \mathcal{T}_n^L(X_1^n)$, set $C_w((X, Y)_1^n) = 0$ for all $w \in \mathcal{T}_n^L$, and, for any $u \prec w \in \mathcal{T}_n^L$ define

$$C_{u,n} = C_u((X, Y)_1^n) = \max \left\{ 1_{\{\Delta_n(u) \geq \delta\}}, \max_{b \in A} C_{bu,n} \right\}. \quad (13)$$

The context tree estimator $\hat{\tau}_{C,n}^\delta = \hat{\tau}_C^\delta((X, Y)_1^n)$ obtained with this procedure can be defined as

$$\hat{\tau}_{C,n}^\delta = \{w \preceq v \in \mathcal{T}_n^L : C_{w,n} = 0 \text{ and } C_{u,n} = 1 \text{ for all } u \prec w\}. \quad (14)$$

Notice that once we have $C_{w,n} = 1$, for a given w , equation (13) implies that for any $u \prec w$, $C_{u,n} = 1$.

Remark 2. The consistency of the original algorithm Context was proved in Rissanen (1983). In this setting (i.e., model selection procedure for context tree model) the statistic used to identify the contexts is given by

$$\Delta_n^X(u) = \sum_{b \in A} \sum_{a \in A} N_n^X(bua) \log \frac{\hat{p}(a|bu)}{\hat{p}(a|u)}.$$

The proof of consistency depends on \hat{p} through the ergodicity of p and its memory relation with τ , which in our case q also satisfies. Therefore, the consistency can be easily adapted for the formulation we are introducing here for sequences of random objects driven by context tree models.

2.3.2 General algorithm Context + Offspring empirical distributions

In this case, the function `stat(u)` inside the general algorithm Context compares the distance between the empirical distribution associate to the string u and the ones associated to its offspring.

Formally, for any finite string $u \in A^*$, define the statistic

$$\tilde{\Delta}_n(u) = \max_{b \in A} \left(\max_{a \in A} |\hat{q}(a|u) - \hat{q}(a|bu)| \right). \quad (15)$$

For any finite string $u \in A^*$ such that τ_u is a terminal subtree, the function `stat(u)` verifies whether $\tilde{\Delta}_n(u) < \delta$. If the inequality is satisfied the subtree τ_u is pruned, otherwise it is kept.

Given $\mathcal{T}_n^L = \mathcal{T}_n^L(X_1^n)$, set $\tilde{C}_w((X, Y)_1^n) = 0$ for all $w \in \mathcal{T}_n^L$, and, for any $u \prec w \in \mathcal{T}_n^L$ define

$$\tilde{C}_{u,n} = \max\{1_{\{\tilde{\Delta}_n(u) \geq \delta\}}, \max_{b \in A} \{C_{bu,n}\}\}. \quad (16)$$

The context tree estimator $\hat{\tau}_{\tilde{C},n}^\delta = \hat{\tau}_{\tilde{C}}^\delta((X, Y)_1^n)$ obtained with this procedure can be defined as

$$\hat{\tau}_{\tilde{C},n}^\delta = \{w \preceq v \in \mathcal{T}_n^L : \tilde{C}_{w,n} = 0 \text{ and } \tilde{C}_{u,n} = 1 \text{ for all } u \prec w\}. \quad (17)$$

where $\text{suf}(w)$ refers to the largest suffix of w . Note that, as well as in the classical algorithm Context, $\tilde{C}_{u,n} = 1$ implies $\tilde{C}_{v,n} = 1$ for all $v \prec u$.

Remark 3. In Galves and Leonardi (2008) it was proved the strong consistency of the estimator (17) (with \hat{p} instead of \hat{q}) for the case of unbounded context tree models. The proof relies on a mixture property which is always satisfied in the case of finite context tree models. In particular, is also true for the law of the response sequence $(Y_n)_n$ since its time memory depends on the time memory of the associated context tree model.

2.3.3 Bayesian Information Criterion (BIC)

This section describes a model selection procedure for the categorical case using the Bayesian Information Criterion. Model selection for context tree models via BIC was first addressed in Csiszár and Talata (2006). We formalize here how the procedure introduced in Csiszár and Talata (2006) is used in our case.

Given a sample $(X, Y)_1^n$ and a constant $c > 0$, the BIC estimator for sequence of random objects driven by context tree models is defined as

$$\hat{\tau}_{BIC,n}^c = \hat{\tau}_{BIC}^c((X, Y)_1^n) = \operatorname{argmax}_{\tau \in \Gamma_n^L} \left\{ \log L_{(\tau, \hat{q})} - c \cdot df(\tau) \log(n) \right\}. \quad (18)$$

where df stands for the degree of freedom of the model. Formally, for any admissible context tree τ , we define, for each $w \in \tau$,

$$df(w) = \sum_{a \in A} 1_{\{N_n^{XY}(w,a) \geq 1\}} - 1$$

and $df(\tau) = \sum_{w \in \tau} df(w)$.

Csiszár and Talata (2006) showed that (18) can be computed efficiently through the following inductive procedure. Starting with \mathcal{T}_n^L , for any $w \in \mathcal{T}_n^L$, define the quantity $V_{w,n} = V_w((X, Y)_1^n) = n^{-c \cdot df(w)} L_{(w, \hat{q})}((X, Y)_1^n)$ and the indicator $\mathcal{X}_{w,n} = \mathcal{X}_w((X, Y)_1^n) = 0$, and for any $w \prec u \in \mathcal{T}_n^L$ define recursively the quantity

$$V_{w,n} = \max \left\{ n^{-c \cdot df(w)} L_{(w, \hat{q})}((X, Y)_1^n), \prod_{b \in A} V_{bw,n} \right\} \quad (19)$$

and the indicator

$$\mathcal{X}_{w,n} = 1 \left\{ \prod_{b \in A} V_{bw,n} > n^{-c \cdot df(w)} L_{(w, \hat{q})}((X, Y)_1^n) \right\}. \quad (20)$$

The estimate obtained solving (18) can be written as

$$\hat{\tau}_{BIC,n}^c = \{w \preceq s \in \mathcal{T}_n^L : \mathcal{X}_{w,n} = 0 \text{ and } \mathcal{X}_{u,n} = 1 \text{ for all } u \prec w\}. \quad (21)$$

Observe that, on the contrary to the algorithm Context, $\mathcal{X}_{w,n} = 1$ for a given w , does not imply that $\mathcal{X}_{u,n} = 1$ for any $u \prec w$.

Remark 4. The fact that the recursive procedure above effectively solves the BIC optimization problem and the consistency of the estimator were proved in Csiszár and Talata (2006) for the case of context tree models $(X_n)_n$. In Csiszár and Talata (2006), the analogous to equation (18) is

$$\hat{\tau}_{BIC}^c(X_1^n) = \operatorname{argmax}_{\tau \in \Gamma_n^L} \left\{ \log L_{(\tau, \hat{p})}(X_1^n) - c \cdot df(\tau) \log(n) \right\},$$

in which the maximum log-likelihood is computed using the empirical probabilities of the distribution p , instead of q . Since the distribution p affects these proofs only through the ergodic theorem and its memory dependency on τ , it is straight forward that all the proofs can be adapted to the case of the conditional log-likelihood considered in this section, which depends on \hat{q} instead of \hat{p} .

2.3.4 Tuning the model selection methods

The threshold δ used in the procedures based on the general algorithm Context and the penalization constant c involved in the model selection procedure based on BIC are hyperparameters whose values must be specified a priori. Small values of δ and c result in big context trees (big in the sense of its size) and, consequently, overfitted models while high values of these hyperparameters give rise to context trees of small size and underfitted models.

To choose the value of the hyperparameters one can use the Smallest Maximizer Criterion (SMC) (Galves *et al.*, 2012). The SMC procedure was introduced in Galves *et al.* (2012) to tune the model selection method for context tree models based on BIC. Here we extend this framework to the case of sequence of random objects driven by context tree models for tuning the model selection methods proposed in the categorical case.

The SMC procedure consists of two steps. In the first step a set of candidate models is computed, namely the *champion trees*. In the second step, an optimal model is chosen within the set of champion trees. The champion trees obtained will depend on the model selection procedure being tuned and may differ from one procedure to another.

In this section, $\hat{\tau}_n^\ell$ denote either $\hat{\tau}_{BIC,n}^\ell$, $\hat{\tau}_{C,n}^\ell$ or $\hat{\tau}_{\hat{C},n}^\ell$.

Step 1. Compute the champion trees. The champion trees constitute a set of estimated context trees $\hat{\tau}_n^\ell$ obtained by varying the value of the hyperparameter $\ell \geq 0$. When $\ell = 0$ we obtain the admissible context tree of maximum size $\hat{\tau}_n^0 = \mathcal{T}_n^L$ (the more complex model). By successively increasing the value of ℓ , we obtain a finite set of context trees totally ordered with respect to the order \succ , say $\mathcal{C}_n = \{\mathcal{T}_n^L = \hat{\tau}_0 \succ \hat{\tau}_1 \succ \dots \succ \hat{\tau}_K = \tau_{root}\}$. It is not hard to see that there exists a value $\ell = \ell_{max}$ such that for any $\ell \geq \ell_{max}$ the estimated model is the empty tree, $\tau_{root} = \emptyset$, which refers to the independent model.

A crucial fact for the consistency of SMC is that the context tree generating the sample data belongs eventually almost surely to the set of champion trees as n goes to ∞ . This is the content of the theorem below and its proof is a co-factor of Theorem 6 in Galves *et al.* (2012) and an extra argument given in Appendix B.

Proposition 2. *Assume $(X_1, Y_1), \dots, (X_n, Y_n)$ is a sample of a sequence of random objects driven by context tree model with parameters (τ^*, p^*, q^*) , with $|\tau^*| \leq L$. Consider the map $\ell \in [0, +\infty] \mapsto \hat{\tau}_n^\ell \in \Gamma^L(X_n^1)$ with $\hat{\tau}_n^\ell$ denoting either $\hat{\tau}_C^\ell((X, Y)_1^n)$, $\hat{\tau}_{BIC}^\ell((X, Y)_1^n)$ or $\hat{\tau}_{\hat{C}}^\ell((X, Y)_1^n)$ and denote by*

$$\mathcal{C}_n = \{\hat{\tau}_n^\ell : \ell \in [0, +\infty]\}. \quad (22)$$

Then \mathcal{C}_n is totally ordered with respect to \succ and eventually almost surely $\tau^ \in \mathcal{C}_n$ as $n \rightarrow \infty$.*

It is well known that the bigger the context tree, the higher its sample likelihood. When SMC was introduced for tuning the BIC model selection algorithm for context tree models, Galves *et al.* (2012) theoretically proved the existence of a change of regime in the rate in which the sample likelihood increases in the set of champion trees (Theorem 7 in Galves *et al.* (2012)). The authors also showed that such changing point in the likelihood

function occurs at the true model generating the data. A consequence of the proof of consistency of SMC is that the change of regime does not depend on the estimation method used to obtain the champion trees, but only on some properties of the set. For this reason we state the next theorem in a slightly more general form that stated in Galves *et al.* (2012) and in terms of sequences of random objects driven by context tree models.

Theorem 1. *Assume $(X_1, Y_1), \dots, (X_n, Y_n)$ is a sample of a sequence of random objects driven by a context tree model with parameters (τ^*, p^*, q^*) , with $|\tau^*| \leq L$. Given a set $\mathcal{C}_n \subset \Gamma^L(X_n^1)$ satisfying*

(i) \mathcal{C}_n is totally ordered with respect to \succ and

(ii) eventually almost surely $\tau^* \in \mathcal{C}_n$ as $n \rightarrow \infty$.

The following holds:

1. For any $\tau \in \mathcal{C}_n$, with $\tau \prec \tau^*$, there exists a constant $c(\tau^*, \tau) > 0$ such that

$$\log L_{(\tau^*, \hat{q})} - \log L_{(\tau, \hat{q})} \geq c(\tau^*, \tau)n \quad (23)$$

2. For any $\tau \prec \tau' \in \mathcal{C}_n$, with $\tau^* \preceq \tau$, there exists a constant $c(\tau', \tau) > 0$ such that

$$\log L_{(\tau', \hat{q})} - \log L_{(\tau, \hat{q})} \leq c(\tau', \tau) \log n \quad (24)$$

Theorem 1 is a co-factor of Theorem 7 in Galves *et al.* (2012) and its proof is presented in Appendix C. This theorem provides a criterion to choose the optimal model (and consequently, the optimal ℓ value) among the champion trees. That is to say, the model in \mathcal{C}_n at which the change of regime occurs. This is the scope of the second step of the SMC.

Step 2. Identify the optimal tree. To select an optimal tree $\hat{\tau}_{\hat{k}} \in \mathcal{C}_n$ we use the following consequence of Theorem 1. For any $\tau \succeq \tau' \succeq \tau^*$,

$$\lim_{n \rightarrow \infty} \frac{\log L_{(\tau, \hat{q})}((X, Y)_1^n) - \log L_{(\tau', \hat{q})}((X, Y)_1^n)}{n} = 0. \quad (25)$$

This suggests that $\hat{\tau}_{\hat{k}} \in \mathcal{C}_n$ should be the smallest context tree such that the rescaled difference between the conditional log-likelihood of $\hat{\tau}_{\hat{k}}$ and $\hat{\tau}_{\hat{k}-1}$ (its successor in the order \prec) decreases as n increases. This is done by comparing average bootstrapped conditional log-likelihood using a t-test, as follows.

- a) Fix two different sample sizes $n_1 < n_2 < n$. Obtain B independent bootstrap resamples of $(X_1, Y_1), \dots, (X_n, Y_n)$, of size n_2 , say

$$(\mathbf{X}^*, \mathbf{Y}^*)^{(b, n_2)} = \{(X_1^*, Y_1^*)^b, \dots, (X_{n_2}^*, Y_{n_2}^*)^b\}, \quad b = 1, \dots, B.$$

Similarly, let $(\mathbf{X}^*, \mathbf{Y}^*)^{(b, n_1)}, b = 1, \dots, B$ be another set of independent bootstrap samples of size n_1 constructed by truncating the sequences $(\mathbf{X}^*, \mathbf{Y}^*)^{(b, n_2)}$ to size n_1 .

- b) For each $\hat{\tau}_k \in \mathcal{C}_n, k = K, \dots, 2$ and its successor $\hat{\tau}_{k-1} \in \mathcal{C}_n$ ($\hat{\tau}_k \prec \hat{\tau}_{k-1}$) compute the rescaled log-likelihood differences

$$D_b^k(n_j) = \frac{\log L_{(\hat{\tau}_k, \hat{q}_k)}((X^*, Y^*)^{(b, n_j)}) - \log L_{(\hat{\tau}_{k-1}, \hat{q}_{k-1})}((X^*, Y^*)^{(b, n_j)})}{n_j^{0.9}}, \quad (26)$$

for $j = 1, 2$ and $b = 1, \dots, B$.

Apply a one-side t-test to compare the mean of the samples

$$\{D_b^{(\hat{\tau}_k, \hat{\tau}_{k-1})}(n_1), b = 1, \dots, B\} \text{ and } \{D_b^{(\hat{\tau}_k, \hat{\tau}_{k-1})}(n_2), b = 1, \dots, B\}.$$

- c) Select as optimal tree $\hat{\tau}_{\hat{k}}$ the smallest champion tree such that the test rejects the equality of the means in favor of the alternative hypothesis $E(D^{(\hat{\tau}_k, \hat{\tau}_{k-1})}(n_1)) < E(D^{(\hat{\tau}_k, \hat{\tau}_{k-1})}(n_2))$.

Step 2 involves the computation of bootstrap resamples of a sequence of random objects driven by context tree models. The toolbox implements different bootstrap strategies for that. Before introducing them, we describe the bootstrap schemes implemented to resample a context tree model (X_1, \dots, X_n) .

- (Parametric bootstrap) The bootstrap samples are obtained by drawing from a parameterized distribution (Bühlmann, 2002) in the following way:

- Choose a hyperparameter value $l \geq 0$ and estimate the model $(\hat{\tau}, \hat{p})$ using the data (X_1, \dots, X_n) .
- Generate the bootstrap samples by simulating from the approximated distribution $\hat{F} = \hat{F}_{(\hat{\tau}, \hat{p})}$,

$$(X_1^*, \dots, X_n^*) \sim \hat{F}_{(\hat{\tau}, \hat{p})}.$$

- (Block bootstrap) Split the sample (X_1, \dots, X_n) into non-overlapping blocks (see Figure 1a). These blocks are built by using a *renewal context* of X_1^n to split the sequence. A renewal context is a string from which the next symbols can be generated without knowing further information from the past. A resample is obtained by repeatedly sampling uniformly a block from the set of blocks and concatenating them. In the toolbox, the user can specify the renewal context, or it can be computed from the estimated model $(\hat{\tau}, \hat{p})$.

A bootstrap resampling $(X^*, Y^*)_1^n$ of the bivariate chain $(X, Y)_1^n$ can be obtained with two different procedures:

- (Parametric bootstrap)
 - (a) Choose a hyperparameter value $\ell \geq 0$ and estimate $(\hat{\tau}, \hat{q})$ from $(X, Y)_1^n$.
 - (b) Obtain a bootstrap resampling $(X^*)_1^n$ of the sequence X_1^n using one of the bootstrap strategies for context tree models described above. The user can also choose not to resample the sequence X_1^n .
 - (c) Generate a sequence $(Y^*)_1^n$ using the distribution $\hat{q}(\cdot | c_{\hat{\tau}}((X^*)_1^n))$.
- (Block bootstrap) Split the sample $(X_1, Y_1), \dots, (Y_n, X_n)$ into non-overlapping blocks using a renewal context (see Figure 1b). In this case, a renewal context is a string from X_1^n such that from it is possible to generate both the next symbols of sequence $(X_n)_n$ and its associates responses $(Y_n)_n$, without further information from the past. The bootstrap samples can be obtain by repeatedly sampling uniformly from the set of blocks and concatenating them.

Remark 5. Bühlmann (2000) proposed a procedure based on Risk functions for tuning the algorithm Context. The SeqROCTM toolbox implements this tuning procedure for the particular case of the Final prediction error risk. This procedure is available in the toolbox for tuning the model selection procedures for context tree models and the model selection procedures for sequence of random objects driven by context tree models (for the categorical case).

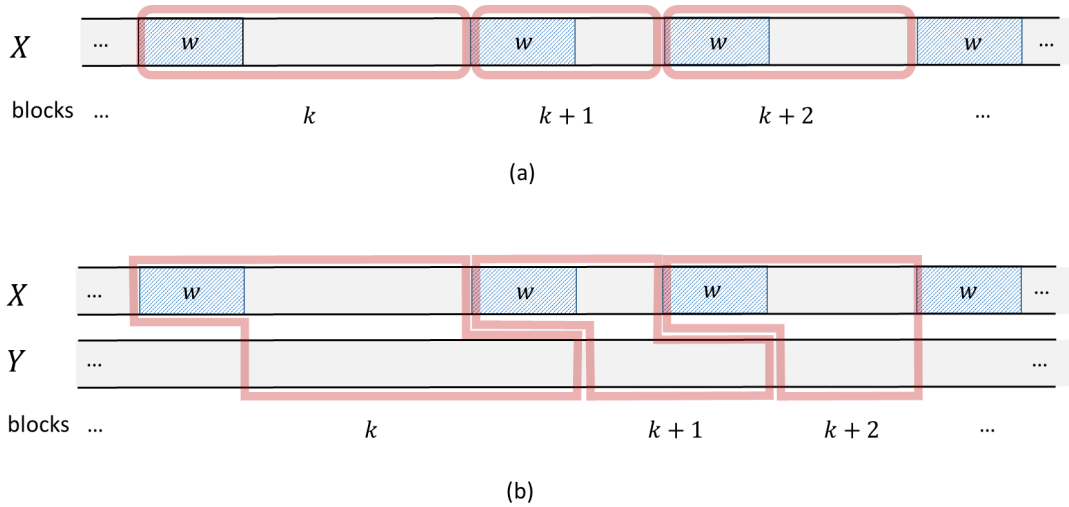


Figure 1: Illustration of a sequence split in blocks using a renewal context w for (a) context tree models and (b) sequence of random objects driven by context tree models

3 Software Architecture

The SeqROCTM toolbox have been designed following a modular structure. The toolbox consists of functions written in Matlab that can be grouped in four modules regarding their functionalities (see Figure 2). This architecture makes the software easy to update, either by adding new functionalities or by improving the existing ones.

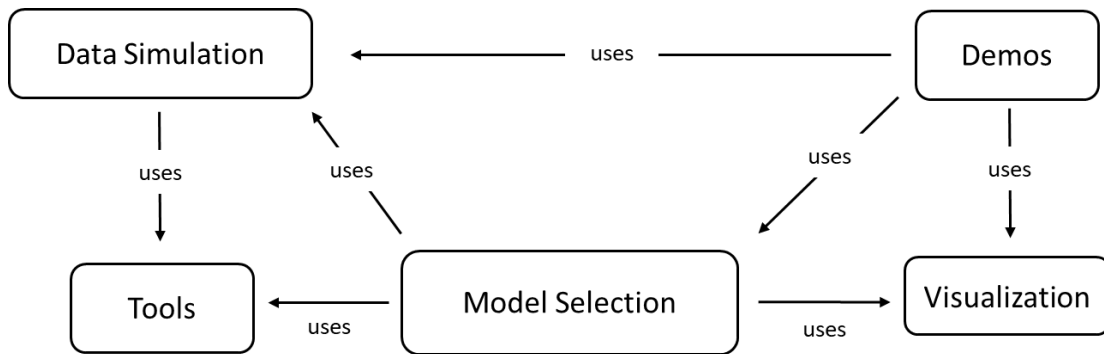


Figure 2: The software architecture of the Matlab SeqROCTM toolbox.

The module Data Simulation includes routines to simulate sequences of inputs (i.e., context tree models), to simulate the response sequence of a sequence of random objects driven by context tree models and to simulate the bivariate sequence. The module Vi-

sualization implements the algorithm described in Mill (2020) to graphically show a tree structure. This module contains also a routine to print the context tree in the console. The Tools module implements several functions that can assist the researcher during the experimental design and data analysis. Some of those functions are also invoked by functions in other modules. Some demos illustrating how to use the toolbox are included in the Demo module.

The main functions of the toolbox are in the Model Selection module. This module contains all the model selection procedures and tuning algorithms introduced in Section 2. Figure 3 presents a close-up of this module.

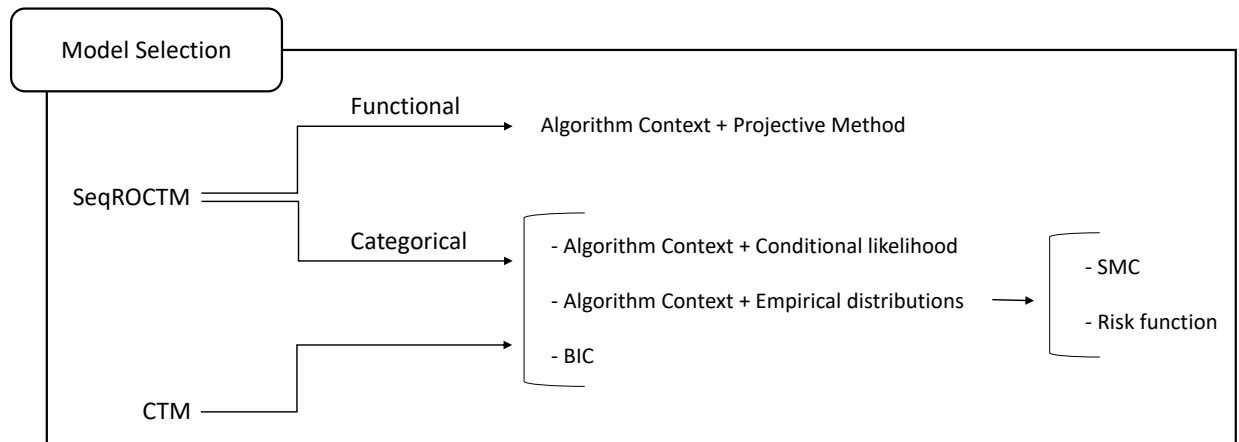


Figure 3: The different algorithms included in the Model Selection module of the SeqROCTM toolbox.

The novelty implemented in this toolbox it is illustrated in Figure 3 by the branch growing from the SeqROCTM node, that is, the mathematical framework to make inference in the class of sequences of random objects driven by context tree models. Nevertheless, due to the close relation to model selection in context tree models, we end up including in the toolbox several existing model selection algorithms for context tree models.

Up to our knowledge, there is only an R-package implementing model selection in context tree models introduced by Mächler and Bühlmann (2004). This package implements the algorithm Context and a tuning procedure for the algorithm Context based on Risk functions. Our toolbox also works as an alternative tool for this purpose.

All the implementations are self-contained. The only external function the toolbox uses is the function `permn` (van der Geest, 2019).

4 Illustrative example: The Goalkeeper game

This section presents the major functionalities of the SeqROCTM toolbox through an illustrative example.

The Goalkeeper game is a video game developed by the NeuroMat team (<https://game.numec.prp.usp>) as a tool to investigate the conjecture that the brain does statistical model selection (Castro, 2006). During the game, the kicker can throw the ball in three directions: *left*, *center* or *right*. The agent, playing the role of the Goalkeeper in a soccer penalty shootout, has to defend as much penalties as possible by predicting, at each step, in which direction the kicker will throw the ball. The kicker’s choices are randomly generated according to a context tree model. The Goalkeeper game has been used in the experimental protocol of some neurobiological experiments (Stern *et al.*, 2020).

Here, for simplicity, instead of collecting some data using the game, we will simulate different participant strategies to generate the responses. We show how the toolbox can be used to assess the participant strategy from the data. The aim is that the estimated strategy matches the one used to generate the responses matches.

To generate the sequence of kick directions, we define a context tree model that generates sequences according to the following rule: after a shot to the *left* the kicker always send the ball to the *center*, after a shot to the *right* he always send the ball to the *left*, but after a shot to the *center*, if one step back he sent the ball to the *center*, then he send the ball to the *left*. Otherwise, if one step back he sent the ball to the *left*, then he send the ball to the *right* with probability 0.8 and to the *center* with probability 0.2.

Formally, the directions *left*, *center* and *right* are represented by the symbols 0, 1 and 2, respectively. These symbols will conform the alphabet. When using the toolbox, an alphabet is always represented by a vector of consecutive positive integers from zero to the number of elements minus one. A context tree is defined through a cell array of vectors, each vector representing a context (i.e., a leaf of the tree). The distributions associated

to the contexts are specified by a matrix with the number of rows equals the number of contexts and the number of columns equals the number of symbols in the alphabet. In this way, the k -th row contains the distribution associated to the k -th context in the cell array defining the context tree. The following source code defines these variables according to the example.

```

% alphabet of three symbols
A = [0,1,2];

% context tree containing four contexts 0, 2, 01, 11
tau = {0, 2, [0,1], [1,1]};

% distributions associated to each contexts (4x3 matrix)
% e.g., first row indicates the distribution of context 0, that is p(0|0)=0,
    p(1|0)=1, p(2|0)=0
p = [0, 1, 0 ; 1, 0, 0; 0, 0.2, 0.8; 1, 0, 0 ];

% visualize the context tree
draw_contexttree(tau, A);

```

To generate a sequence of stimuli according to a given context tree model we use the function `generatesampleCTM`, which receives as inputs a context tree, the probability distributions associated to the contexts, an alphabet and the length of the sequence we want to generate. The code below generates a sequence of inputs of length 300, using the variables already defined.

```

% length of the sequence
seq_length = 300;

% sequence of stimuli (context tree model) in a row vector
X = generatesampleCTM(tau, p, A, seq_length);

```

The variable `X` contains the sequence of the kicker choices. We will define three different strategies for the goalkeeper and generate a response sequence for each of them, say `Y1`,

$Y2$ and $Y3$. When applying the model selection procedures to the data $(X1, Y1)$, $(X2, Y2)$ and $(X3, Y3)$, the desired result is to recover the strategy used to simulate the goalkeeper responses on each case.

The three strategies used to simulate the goalkeeper responses are the following:

- Strategy 1. Every time the goalkeeper see the shot of the kicker to the left, he will defend the next shot to the center. After a shot to the center, the goalkeeper will defend to the right. And after a shot to the right, the goalkeeper will defend to the left. Using the variables already defined, this strategy can be translated as follows: If $X_n = 0$, then $Y_{n+1} = 1$. If $X_n = 1$, then $Y_{n+1} = 2$. If $X_n = 2$, then $Y_{n+1} = 0$.
- Strategy 2. The goalkeeper learns the relevant pasts (i.e., the contexts) of the sequence X and, at each step, he identifies the context associated to the current past and chooses the direction with grater probability of being generated after that context. This is the strategy that maximizes the probability of matches. This means that whenever the goalkeeper see a shot to the center preceded by a shot to a left, he will defend the next shot to the right. On the contrary, if the shot to the center is preceded by another shot to a center, the goalkeeper will defend the next shot to the left. Besides, if the kicker shot the ball to the left, the goalkeeper will defend the next shot to the center and if the kicker shot the ball to the right, the goalkeeper will defend the next ball to the left. This means that if $X_n = 0$, then $Y_{n+1} = 1$. If $X_n = 1$ and $X_{n-1} = 0$, then $Y_{n+1} = 2$. If $X_n = 1$ and $X_{n-1} = 1$, then $Y_{n+1} = 0$. If $X_n = 2$, then $Y_{n+1} = 0$.
- Strategy 3. The goalkeeper pays no attention to the temporal dependences encoded in the kicker strategy, at each step, randomly chooses in an independent and uniform way *left*, *center* or *right*.

The source code below defines the context tree and the distributions used to simulate the goalkeeper responses according to the described strategies. To generate the response sequence, the function `generatesampleYSeqROCTM` is used.

For the categorical case, which is the case of the current example, the observed sequence

of responses must be stored in a row vector. For the functional case, the response sequence must be specified by a matrix containing on each column a chunk of function (this will be exemplified later, in the illustrative example presented in Section 4).

```
% Strategy 1
ctx1 = {0, 1, 2};
q1 = [0 1 0; 0 0 1; 1 0 0];
[X1, Y1] = generatessampleYSeqROCTM(X, ctx1, q1, A);
```

```
% Strategy 2
ctx2 = {0, 2, [0,1], [1,1]};
q2 = [0, 1, 0 ; 1, 0, 0; 0, 0, 1; 1, 0, 0 ];
[X2, Y2] = generatessampleYSeqROCTM(X, ctx2, q2, A);
```

```
% strategy 3
ctx3 = {};
q3 = [1/3 ; 1/3; 1/3 ];
[X3, Y3] = generatessampleYSeqROCTM(X, ctx3, q3, A);
```

Now that we have some data, i.e., $(X1, Y1)$, $(X2, Y2)$ and $(X3, Y3)$, we will exemplified how the functions responsible for model selection can be used. For the current example, we will use the function `tune_SeqROCTM`, which receives as mandatory inputs the data and the alphabet. There exists a lot of optional name-value pairs arguments, which could be specified also as input of the function. The following source code shows how to invoke the `tune_SeqROCTM` function specifying a different estimation method for each data and SMC as tuning procedure for all the cases.

```
% some parameters value
c_min = 0;
c_max = 1000; % high enough, such as to obtain the empty tree
max_height = 6;
alpha = 0.05;
```

```

% tune the SeqROCTM model for each strategy
[~,~, r1] = tune_SeqROCTM(X1, Y1, A, 'TuningMethod', 'smc',          ...
                           'EstimationMethod', 'context_cL',      ...
                           'MaxTreeHeight', max_height,          ...
                           'ParameterLowerBound', c_min,         ...
                           'ParameterUpperBound', c_max,         ...
                           'Alpha', alpha);

[~,~, r2] = tune_SeqROCTM(X2, Y2, A, 'TuningMethod', 'smc',          ...
                           'MaxTreeHeight', max_height,          ...
                           'EstimationMethod', 'context_empD',    ...
                           'ParameterLowerBound', c_min,         ...
                           'ParameterUpperBound', c_max,         ...
                           'Alpha', alpha);

[~,~, r3] = tune_SeqROCTM(X3, Y3, A, 'TuningMethod', 'smc',          ...
                           'MaxTreeHeight', max_height,          ...
                           'EstimationMethod', 'bic',             ...
                           'ParameterLowerBound', c_min,         ...
                           'ParameterUpperBound', c_max,         ...
                           'Alpha', alpha,                        ...
                           'BootNsamples', 200,                 ...
                           'BootStrategy', 'blocks');

% show the results of the estimation procedures
figure
for i = 1 : 3
subplot(2,3,i)
% get the structure of the corresponding model
eval(['r = r' num2str(i) ';' ]);
% get the values from the structure r

```

```

nleaves = cellfun(@(x) size(x,2), r.champions);
ML = r.fvalues;
idtree = r.idxOptTree;
cutoff = r.prmvalues;
% draw the curve
plot(nleaves, ML, '*--b')
hold on; plot(nleaves(idtree), ML(idtree), 'ro');
text(nleaves(idtree)+0.5, ML(idtree), ['\leftarrow C = '
    num2str(cutoff(idtree))], 'FontSize', 8);
ylabel('$$\log(L_{(\tau, \hat{q})}(Y_1^n|X_1^n))$$', 'interpreter', 'latex');
xlabel('$$|\tau|$$', 'interpreter', 'latex');
% draw the choosen context trees
subplot(2,3,3+i)
draw_contexttree(r.champions{idtree}, A, [1 0 0], 3);
end

% Calling the model selection procedure without tuning (using the default
% value of the hyperparameter)
[tau1, q1] = estimate_discreteSeqROCTM(X1, Y1, A, 'MaxTreeHeight', max_height,
    'EstimationMethod', 'context_empD');
[tau2, q2] = estimate_discreteSeqROCTM(X2, Y2, A, 'MaxTreeHeight', max_height,
    'EstimationMethod', 'context_cL');
[tau3, q3] = estimate_discreteSeqROCTM(X3, Y3, A, 'MaxTreeHeight', max_height,
    'EstimationMethod', 'bic');

% show the results in the console
print_tree(tau1);
print_tree(tau2);
print_tree(tau3);

```

Figure 4 shows the results obtained for each simulated strategy (this Figure is also generated by the code above). For each goalkeeper strategy it is shown the conditional

log-likelihood of each champion tree as a function of its size. The optimal model chosen using SMC is marked with a red circle and the corresponding context tree is graphically shown below. The optimal value of the hyperparameters (δ in the first two cases and c in the third one) is also shown in the plot. For all the strategies, the model estimated from the data using the tuning procedure matches the context tree used to simulate the goalkeeper responses.

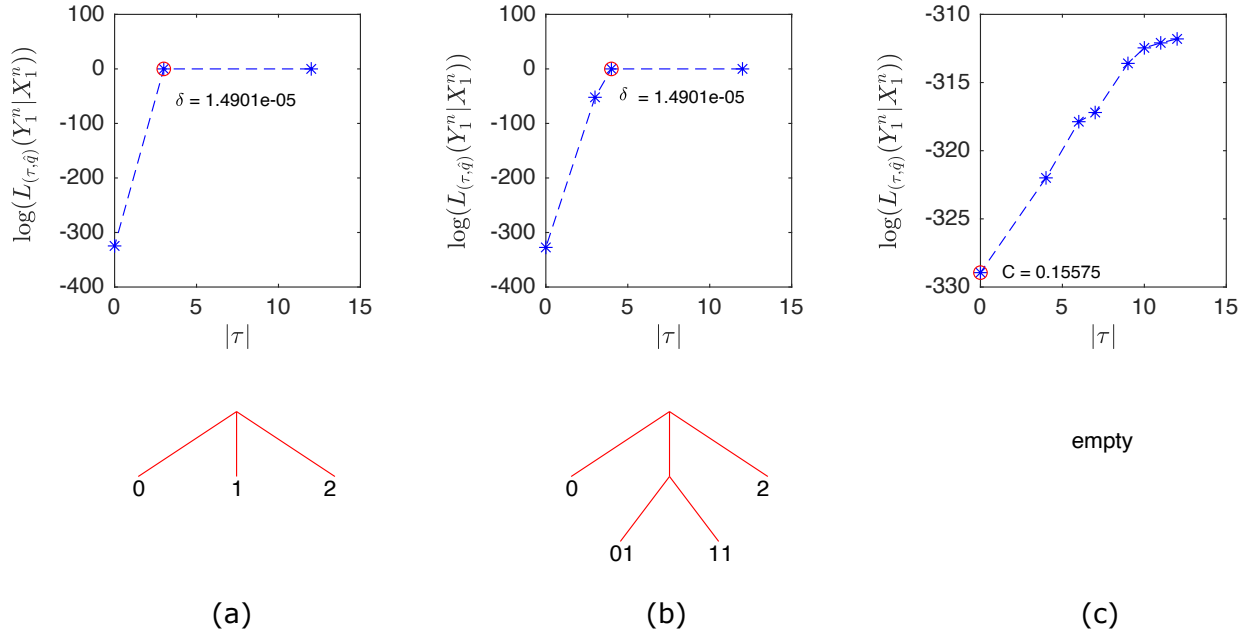


Figure 4: Result of the tuning procedure for a) strategy 1 b) strategy 2 and c) strategy 3. The first row shows plots of the logarithm of the conditional likelihood vs. the number of contexts for each model in the set of champion trees. The red circle indicates the optimal model chosen using SMC. The second row shows the context tree corresponding to the optimal model.

The source code also shows how to invoke the function `estimate_discreteSeqROCTM`, which is responsible for model selection without any tuning procedure. The user can specify as input a value for the hyperparameter (as it was done for strategy 3). If no value for the hyperparameter is given as input, a default value is used.

5 Illustrative example: Retrieving the structure of probabilistic sequences from EEG data

Humans are great at learning statistical regularities from sequences of stimuli. Having learned patterns from the inflow of sensory information, one can predict the upcoming stimuli to improve perception and decision making (Summerfield and de Lange, 2014; de Lange *et al.*, 2018). Understanding the capacity of the brain to learn statistical regularity from temporal sequences has been the focus of several researches in neuroscience. This was the focus of the recently published study by Hernández *et al.* (2021) from which we extracted the current illustrative example .

Consider a sequence of auditory stimuli generated by a context tree model. This sequence is presented to a volunteer while electroencephalographic (EEG) signals are recorded from his scalp. In this framework, the conjecture that the brain identifies statistical regularities from sequences of stimuli can be rephrased by claiming that the brain identifies the context tree used to generate the sequence of auditory stimuli. If this is the case, a signature of the context tree should be encoded in the brain activity. The question is whether this signature can be identified in the EEG data recorded during the experiment.

The auditory units used as stimulus are either *strong beats*, *weak beats* or *silent units*, represented by symbols 2, 1 and 0, respectively. The statistical regularity encoded in the sequences of stimuli can be informally described as follows. Start with the deterministic sequence

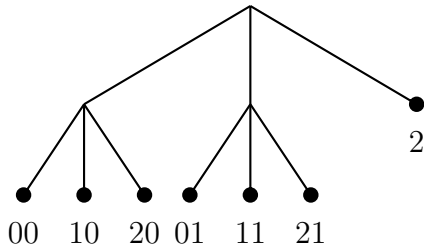
$$2\ 1\ 1\ 2\ 1\ 1\ 2\ 1\ 1\ 2\ 1\ 1\ 2\ \dots,$$

then replace each weak beat (symbol 1) by a silent unit (symbol 0) with a small probability, say 0.2, in an independent way. An example of a sequence produced in this way would be

$$2\ 1\ 1\ 2\ 0\ 1\ 2\ 1\ 1\ 2\ 0\ 0\ 2\ \dots$$

This stochastic sequence constitutes a sample of a context tree model compatible with the context tree and the family of transition probabilities shown in Figure 5.

To obtain a context tree from the EEG data the statistical model selection procedure for sequences of random objects driven by context tree models introduced for the functional



context w	$p(0 w)$	$p(1 w)$	$p(2 w)$
2	0.2	0.8	0
21	0.2	0.8	0
20	0.2	0.8	0
11	0	0	1
10	0	0	1
01	0	0	1
00	0	0	1

Figure 5: Graphical representation of the context tree and the transition probabilities associated to the contexts.

case is employed. This procedure is applied separately to each participant data. Participants are not exposed exactly to the same sequence of stimuli, but different realizations of the same context tree model.

This illustrative example presents a tiny part of a wider experimental protocol introduced in Hernández *et al.* (2021). The experimental protocol involves 19 participants, two different context tree models to generate the sequences of stimuli and 18 electrodes in which the EEG data is recorded. To show how the SeqROCTM was used here, we will consider only the EEG signals recorded in one electrode for 3 participants.

We start by exemplifying how to generate sequences of stimuli of length 700 using the context tree model of Figure 5.

```

% number of volunteers
n_volunteers = 3;

% alphabet and context tree model used to generate the sequence of stimuli
A = [0,1,2];
tau = {[0,0], [1,0], [2,0], [0,1], [1 1], [2,1], 2};
p = [0, 0, 1 ; 0, 0, 1; 0.2, 0.8, 0; 0, 0, 1; 0, 0, 1; 0.2, 0.8, 0; 0.2, 0.8,
    0];

% length of the sequences of stimuli

```

```

seq_length = 700;

% Sequences of stimuli
% matrix X of 3x700 containing on each row a sequence of stimuli
Xdata = zeros(3,700);
for v = 1 : n_volunteers
    Xdata(v,:) = generatesampleCTM(tau, p, A, seq_length);
end

```

In the following, we load the EEG data recorded from a frontal electrode (FP1) for 3 participants as well as the sequences of stimuli the participants were exposed to. This EEG data is already pre-processed and segmented. The pre-processing details are omitted because are out of the scope of this article.

```

% load sequence of stimuli and EEG data for each volunteer
names_volunteer = {'V02', 'V09', 'V19'};

X = [];
Y = cell(1,3);

for v = 1 : n_volunteers

    % load stimuli data
    vname_i = [names_volunteer{v} '_stimuli'];
    x = load(vname_i);
    x = x.(vname_i);
    X = [X; x];

    % load response data
    vname_r = [names_volunteer{v} '_response'];
    y = load(vname_r);

```

```

y = y.(vname_r);
Y{v} = y;
end

% visualize some symbols of the stimuli sequence and its corresponding EEG
% chunks for volunteer V02
figure;
id_cols = 760:768;
for i = 1 : 9
    % plot the stimuli
    ax = subplot(2, 9, i);
    text(0.5, 0.5, num2str(X(1, id_cols(i))), 'FontSize', 20);
    set( ax, 'visible', 'off')

    % plot the EEG chunk
    ax = subplot(2, 9, 9+i);
    plot(Y{1}(:, id_cols(i)));
    set( ax, 'visible', 'off')
    xlim([0 115])
end

```

Figure 6 shows some elements of the sequence of stimuli and the corresponding EEG chunks for the first volunteer.

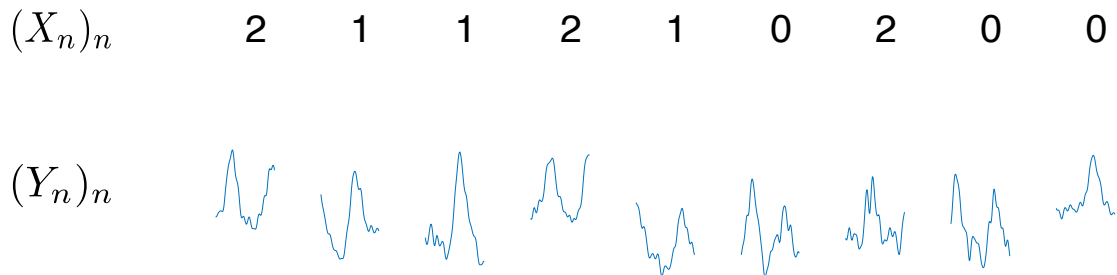


Figure 6: Symbols in the sequence of stimuli and their corresponding EEG chunks for participant V02.

We are now ready to invoke the model selection functions. The values of the parameters required by this function are specified in the source code.

```
% model selection algorithm on the data of each volunteer
nBM = 1000;
Alpha = 0.05;
Beta = 0.05;

rng(1); tree_v02 = estimate_functionalSeqRoCTM(X(1,:), Y{1}, A, 3, nBM, Alpha,
    Beta, 0);
rng(1); tree_v09 = estimate_functionalSeqRoCTM(X(2,:), Y{2}, A, 3, nBM, Alpha,
    Beta, 0);
rng(1); tree_v19 = estimate_functionalSeqRoCTM(X(3,:), Y{3}, A, 3, nBM, Alpha,
    Beta, 0);

% draw the results
figure
subplot(1,3,1)
draw_contexttree(tree_v02, A, [1 0 0], 3);
subplot(1,3,2)
draw_contexttree(tree_v09, A, [0 1 0], 3);
subplot(1,3,3)
draw_contexttree(tree_v19, A, [0 0 1], 3);
```

Figure 7 shows the context tree retrieved from the EEG data of each participant.

It can be seen that for one of the participants the retrieved context tree is the same that the one generating the sequence of stimuli. For the other two participants, the recovered tree differs from the stimuli tree by one branch. More details about this experiment can be found in the Discussion section of Hernández *et al.* (2021).

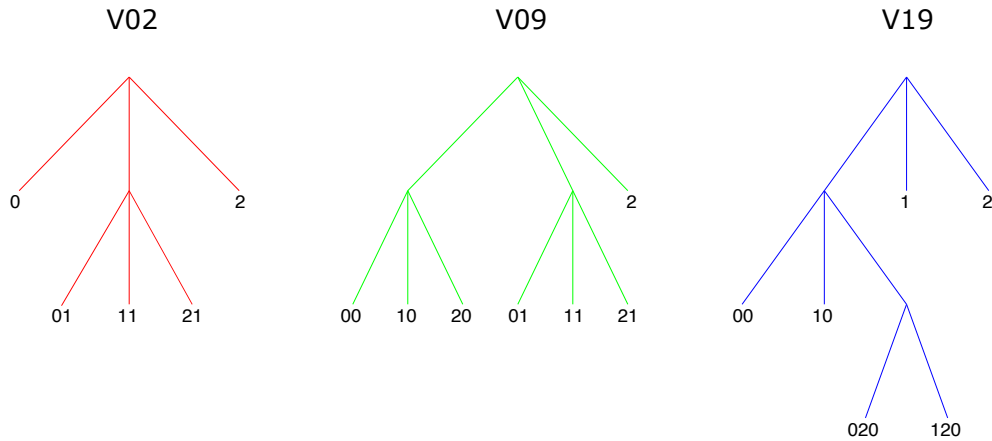


Figure 7: Context tree retrieved for each participant.

6 Conclusions

This paper introduces the Matlab SeqROCTM toolbox aimed to implement model selection procedures in a new class of stochastic process, namely sequences of random objects driven by context tree models. This is a new mathematical framework that finds nice applications in several scientific fields such as neuroscience, linguistic, genetics. The toolbox also implements different procedures for model selection of context tree models. We have described the main functionalities of the toolbox and we give examples of how to use it.

Further extensions are planned in order to solve some limitations of the current version, e.g., greater flexibility when defining the alphabet. For simplicity, in the categorical case we restrict the finite set in which Y_n takes values to the set A , where the associated context tree model takes value.

7 Acknowledge

The authors thanks professor Antonio Galves from University of São Paulo for the discussions and suggestions that contribute to this article.

A Proof of Proposition 1

Proof. Given a string $u \in A^*$, for each $a, b \in \tau_u$, we set

$$Z_{i,n}^{a,b} = D_n^{W_i}((Y_1^{(a)}, \dots, Y_{N_n(a)}^{(a)}), (Y_1^{(b)}, \dots, Y_{N_n(b)}^{(b)})).$$

Assume that the null assumption $H_0^{(u)}$ is true. Then the asymptotic properties of the Kolmogorov-Smirnov statistics implies, for each $a, b \in \tau_u$, with $a \neq b$ and $1 \leq i \leq N$, that $Z_{i,n}^{a,b}$ converges in distribution to $K = \sup_{t \in [0,1]} |B(t)|$ as $n \rightarrow \infty$, where $B = (B(t) : t \in [0, 1])$ is a Brownian bridge.

Let $c_\alpha = \sqrt{-1/2 \ln(\alpha/2)}$ be the α -percentile of the Kolmogorov distribution. When $|\tau_u| = 2$, let say $\tau_u = \{a, b\}$, it holds that $P(\Delta_n^{W_i}(u) > c_\alpha) = P(Z_{i,n}^{a,b} > c_\alpha) = \alpha$ as $n \rightarrow \infty$. If $|\tau_u| > 2$, define $M = \binom{|\tau_u|}{2}$ and in this case $P(\Delta_n^{W_i}(u) > c_{\alpha/M}) = P(\cup_{a,b \in \tau_u} Z_{i,n}^{a,b} > c_{\alpha/M}) = \bar{\alpha} \leq \alpha$ as $n \rightarrow \infty$. Therefore, taking $\delta_\alpha(u) = c_{\alpha/M}$, it holds that

$$P(\Delta_n^{W_i}(u) > \delta_\alpha(u)) \leq \alpha, \quad \text{as } n \rightarrow \infty. \quad (27)$$

In what follows, for each $1 \leq i \leq N$, we define

$$Z_{i,n} = 1\{\Delta_n^{W_i}(u) > \delta_\alpha(u)\}.$$

We will show that for any $a_1, \dots, a_N \in \{0, 1\}$,

$$\lim_{n \rightarrow \infty} P(Z_{i,n} = a_i, \dots, Z_{i,N} = a_N) = \bar{\alpha}^{\sum_{i=1}^N a_i} (1 - \bar{\alpha})^{(N - \sum_{i=1}^N a_i)} \quad (28)$$

where $\bar{\alpha}$ denotes either α , if $|\tau_u| = 2$, or $P(\cup_{a,b \in \tau_u} Z_{i,n}^{a,b} > c_{\alpha/M})$, if $|\tau_u| > 2$.

Denote $\mathcal{G} = \sigma(Y_k^{(as)}, k \geq 1, a \in A)$ and notice that conditionally on \mathcal{G} , the random variables $Z_{1,n}, \dots, Z_{N,n}$ are independent for all $n \geq 1$. By the Skorohod's representation theorem, there is a sequence of random vectors $(\tilde{Z}_{1,n}, \dots, \tilde{Z}_{N,n})_{n \geq 1}$ and a sequence of random elements $(\tilde{Y}_k^{(as)})_{k \geq 1, a \in A}$ taking values in $L^2([0, T])$, both sequences defined in the same probability space $(\tilde{\Omega}, \tilde{\mathcal{F}}, \tilde{P})$ such that

1. for each n , $(\tilde{Z}_{1,n}, \dots, \tilde{Z}_{N,n})$ has the same distribution as $(Z_{1,n}, \dots, Z_{N,n})$,
2. for each k and $a \in A$, the distribution of $\tilde{Y}_k^{(as)}$ is the same as the distribution of $Y_k^{(as)}$.

3. if $\tilde{\mathcal{G}} = \sigma(\tilde{Y}_k^{(as)}, k \geq 1, a \in A)$, then $\tilde{Z}_{1,n}, \dots, \tilde{Z}_{N,n}$ are conditionally independent given $\tilde{\mathcal{G}}$,

4. for each $1 \leq i \leq N$, $\tilde{Z}_{i,n} \rightarrow K$ almost surely with respect to \tilde{P} as $n \rightarrow \infty$.

Item 4 and the Dominate convergence theorem for conditional expectation imply that \tilde{P} -a.s as $n \rightarrow \infty$, for each $1 \leq i \leq N$ and $a_i \in \{0, 1\}$,

$$\tilde{P}(\tilde{Z}_{i,n} = a_i | \tilde{\mathcal{G}}) \rightarrow \bar{\alpha}^{a_i} (1 - \bar{\alpha})^{(1-a_i)}.$$

Therefore, by Item 3 and the Dominate convergence theorem, we have that for any $a_1, \dots, a_N \in \{0, 1\}$, as $n \rightarrow \infty$,

$$\tilde{P}(\cap_{i=1}^N \tilde{Z}_{i,n} = a_i) = \tilde{E} \left[\prod_{i=1}^N \tilde{P}(\tilde{Z}_{i,n} = a_i | \tilde{\mathcal{G}}) \right] \rightarrow \bar{\alpha}^{\sum_{i=1}^N a_i} (1 - \bar{\alpha})^{(N - \sum_{i=1}^N a_i)}$$

The limit in (28) now follows from Item 1.

Finally, Proposition 1 follows from the fact that, given two random variables $\bar{\eta}$ and η with Binomial distribution of parameters $(N, \bar{\alpha})$ and (N, α) , respectively. If $\bar{\alpha} \leq \alpha$, then $P(\bar{\eta} > k) \leq P(\eta > k)$.

□

B Proof of Proposition 2

Proof. For the BIC case the proof is a direct adaptation of Theorem 6 in Galves *et al.* (2012) using Remark 4 and the conditional log-likelihoods $L_{(\hat{\tau}, \hat{q})}((X, Y)_1^n)$ instead of $L_{(\hat{\tau}, \hat{p})}(X_1^n)$.

Algorithm Context is strongly consistent either when using conditional log-likelihood (Rissanen, 1983) or offspring empirical distributions (Galves and Leonardi, 2008) (see Remarks 2 and 3) and since the set of champion trees is countable we get the first part of the proposition. Therefore, it remains to show only the ordering of the champion trees with respect to \succ for both cases.

We shall do the proof for the algorithm Context with conditional log-likelihood. The proof for algorithm Context with offspring distributions its the same (replacing $\tilde{\Delta}_n$ by Δ_n).

Given $0 < \delta_1 < \delta_2$, denote by $\tau^i = \hat{\tau}_{C,n}^{\delta_i}$ for $i = 1, 2$. If $\tau^1 = \emptyset$ then for any w with $N_n^X(w) \geq 1$ and $l(w) \leq L$ it holds that $\Delta(w) < \delta_1 < \delta_2$ and therefore $\tau^2 = \emptyset$.

On the other hand, if $\tau^1 \neq \emptyset$, then for any $w \in \tau^1$ it is enough to show that either $w \in \tau^2$ or there exists $w' \in \tau^2$ such that $w' \prec w$. By (14), once $w \in \tau^1$, for any $s \in A^*$ such that $N_n^X(s) \geq 1$, $l(s) \leq L$ and $s \succeq w$, we have $\Delta(s) < \delta_1 < \delta_2$. Therefore, no string $s \succ w$ belongs to τ^2 , which implies that either $w \in \tau^2$ or there exists $w' \in \tau^2$ such that $w' \preceq w$. \square

C Proof of Theorem 1

The proof is a straight adaptation of Theorem 7 in Galves *et al.* (2012) for the case of conditional log-likelihoods.

Proof. To show the (1) consider any $\tau \in \mathcal{C}_n$ satisfying $\tau \prec \tau^*$ and notice that

$$\begin{aligned} & \log L_{(\tau^*, q)} - \log L_{(\tau, q)} \\ &= \sum_{w^* \in \tau^*} \sum_{a \in A} N_n^{XY}(w^*, a) \log \hat{q}(a|w^*) - \sum_{w \in \tau} \sum_{a \in A} N_n^{XY}(w, a) \log \hat{q}(a|w) \\ &= \sum_{w \in \tau} \sum_{\substack{w^* \in \tau^* \\ w \prec w^*}} \sum_{a \in A} N_n^{XY}(w^*, a) \log \hat{q}(a|w^*) - \sum_{w \in \tau} \sum_{a \in A} N_n^{XY}(w, a) \log \hat{q}(a|w). \end{aligned}$$

Dividing both sides by n , the ergodic theorem implies that $N_n^{XY}(w^*, a)/n \rightarrow q(w^*, a)$, therefore, as $n \rightarrow \infty$, the right hand side of equation above, converges to

$$\sum_{w \in \tau} \sum_{\substack{w^* \in \tau^* \\ w \prec w^*}} \sum_{a \in A} q(w^*, a) \log q(a|w^*) - \sum_{w \in \tau} \sum_{a \in A} q(w, a) \log q(a|w). \quad (29)$$

Now, Jensen's inequality implies that

$$q(w) \sum_{\substack{w^* \in \tau^* \\ w \prec w^*}} \frac{q(w^*)}{q(w)} (q(a|w^*) \log q(a|w^*)) \geq q(wa) \log q(a|w), \quad a \in A, \quad (30)$$

and the equality only holds if $q(a|w) = q(a|w^*)$ for each $a \in A$ and $\tau \in w \prec w^* \in \tau^*$, which is a contradiction with the minimality of τ^* . Therefore there exists at least one symbol $a \in A$ such that the strict inequality holds. Thus, applying inequality (30) in the left term of (29) we conclude that must be strict positive.

To prove (2), observe that

$$\begin{aligned}
& \log L_{(\tau',q)} - \log L_{(\tau,q)} \\
&= \sum_{w' \in \tau'} \sum_{a \in A} N_n^{XY}(w', a) \log \hat{q}(a|w') - \sum_{w \in \tau} \sum_{a \in A} N_n^{XY}(w, a) \log \hat{q}(a|w) \\
&\leq \sum_{w' \in \tau'} \sum_{a \in A} N_n^{XY}(w', a) \log \hat{q}(a|w') - \sum_{w \in \tau} \sum_{a \in A} N_n^{XY}(w, a) \log q^*(a|w) \\
&= \sum_{w \in \tau} \sum_{\substack{w' \in \tau \\ w \prec w'}} \sum_{a \in A} N_n^{XY}(w', a) \log \left(\frac{\hat{q}(a|w')}{q^*(a|w)} \right) \\
&= \sum_{w \in \tau} \sum_{\substack{w' \in \tau \\ w \prec w'}} N_n^X(w') D(\hat{q}(\cdot|w') \parallel q^*(\cdot|w))
\end{aligned}$$

were $D(\nu \parallel \mu) = \sum_{a \in A} \nu(a) \log(\nu(a)/\mu(a))$ is the Kullback-Leibler divergence between two probabilities measures ν and μ with support in same alphabet A .

Now, applying successively Lemmas 6.3 and 6.2 of Csiszár and Talata (2006), we can upper bound the last expression above by

$$\sum_{w \in \tau} \sum_{\substack{w' \in \tau \\ w \prec w'}} N_n^X(w') \sum_{a \in A} \frac{(\hat{q}(a|w') - q^*(a|w))^2}{q^*(a|w)} \leq \sum_{w \in \tau} \sum_{\substack{w' \in \tau \\ w \prec w'}} N_n^X(w') |A| \frac{1}{q_{min}^*} \frac{c \log n}{N_n^X(w')}.$$

with $q_{min}^* = \min_{w \in \tau^*, a \in A} \{q^*(a|w) > 0\}$. □

SUPPLEMENTARY MATERIAL

SeqROCTM toolbox: A Matlab Toolbox for the analysis of Sequences of random objects driven by context tree models. The toolbox implements model selection methods for both sequences of random objects driven by context tree models and context tree models. It includes several others algorithms like: simulation of these kind of stochastic processes, tuning of model selection procedures, distances and dissimilarity measures for context tree models, complexity measures for context tree models, visualization of the tree structure, among others. It is written purely in Matlab language and it is self-contained. The toolbox also contains all the examples and data used in the present paper as well as other demos. The toolbox is freely available at <https://github.com/noslenh/SeqROCTM-Matlab-Toolbox>.

References

- Armstrong, B. C., Frost, R., and Christiansen, M. H., The long road of statistical learning research: past, present and future. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 372(1711), 2017.
- Bühlmann, P. and Wyner, A. J., Variable length markov chains. *The Annals of Statistics*, 27(2):480–513, 1999.
- Bühlmann, P., Model selection for variable length markov chains and tuning the context algorithm. *Annals of the Institute of Statistical Mathematics*, 52:287–315, 2000.
- Bühlmann, P., Sieve bootstrap with variable-length markov chains for stationary categorical time series. *Journal of the American Statistical Association*, 97(458):443–471, 2002.
- Busch, J. R., Ferrari, P. A., Flesia, A. G., Fraiman, R., Grynberg, S. P., and Leonardi, F., Testing statistical hypothesis on random trees and applications to the protein classification problem. *The Annals of Applied Statistics*, 3(2):542–563, 2009.
- Castro, B. D. Processos estocásticos conduzidos por cadeias com memória de alcance variável e o jogo do goleiro, 2006.
- Conway, C. M., How does the brain learn environmental structure? ten core principles for understanding the neurocognitive mechanisms of statistical learning. *Neuroscience and Biobehavioral Reviews*, 112:279 – 299, 2020.
- Csiszár, I. and Talata, Z., Context tree estimation for not necessarily finite memory processes, via bic and mdl. *IEEE Transactions on Information Theory*, 52(3):1007–1016, 3 2006.
- Cuesta-Albertos, J. A., Fraiman, R., and Ransford, T., Random projections and goodness-of-fit tests in infinite-dimensional spaces. *Bulletin of the Brazilian Mathematical Society, New Series*, 37(4):477–501, 2006.
- Lange, F. P.de , Heilbron, M., and Kok, P., How Do Expectations Shape Perception? *Trends in Cognitive Sciences*, 22(9):764–779, September 2018.

- Duarte, D., Galves, A., and Garcia, N. L., Markov approximation and consistent estimation of unbounded probabilistic suffix trees. *Bulletin of the Brazilian Mathematical Society*, 37(4):581–592, Dec 2006.
- Duarte, A., Fraiman, R., Galves, A., Ost, G., and Vargas, C. D., Retrieving a context tree from eeg data. *Mathematics*, 7(5), 2019.
- Fernández, R. and Galves, A., Markov approximations of chains of infinite order. *Bulletin of the Brazilian Mathematical Society*, 33(3):295–306, Nov 2002.
- Galves, A. and Leonardi, F., Exponential inequalities for empirical unbounded context trees. *In and Out of Equilibrium 2*, 2008.
- Galves, A. and Löcherbach, E., Stochastic chains with memory of variable length. *TICSP Series*, 38:117–133, 2008.
- Galves, A., Galves, C., García, J. E., Garcia, N. L., and Leonardi, F., Context tree selection and linguistic rhythm retrieval from written texts. *Ann. Appl. Stat.*, 6(1):186–209, 2012.
- Garrido, M. I., Sahani, M., and Dolan, R. J., Outlier responses reflect sensitivity to statistical structure in the human brain. *PLOS Computational Biology*, 9(3), 2013.
- Hernández, N., Duarte, A., Ost, G., Fraiman, R., Galves, A., and Vargas, C. D., Retrieving the structure of probabilistic sequences of auditory stimuli from eeg data. *Scientific Reports*, 11(3520), 2021.
- Mächler, M. and Bühlmann, P., Variable length markov chains: Methodology, computing, and software. *Journal of Computational and Graphical Statistics*, 13(2):435–455, 2004.
- Mill, B., Drawing presentable trees. *Python Magazine*, <https://lmlib.github.io/pymag-trees/>, 2020.
- Rissanen, J., A universal data compression system. *IEEE Trans. Inf. Theor.*, 29(5):656–664, 1983.

- Schapiro, A. and Turk-Browne, N. Statistical learning. In Toga, A. W., editor, *Brain Mapping*, pages 501 – 506. Academic Press, Waltham, 2015.
- Stern, R. B., d’Alencar, M. S., Uscapi, Y. L., Gubitoso, M. D., Roque, A. C., Helene, A. F., and Piemonte, M. E. P., Goalkeeper game: A new assessment tool for prediction of gait performance under complex condition in people with parkinson’s disease. 50(12), 2020.
- Summerfield, C. and Lange, F. P.de , Expectation in perceptual decision making: neural and computational mechanisms. *Nature Reviews Neuroscience*, 15(11):745–756, November 2014.
- Geest, J.van der , Permutations with repetition, all or a subset. *MATLAB Central File Exchange*, 2019.
- Helmholtz, H.von . *Handbuch der physiologischen Optik*, volume III. Leopold Voss, 1867. translated by The Optical Society of America in 1924 from the third german edition, 1910, Treatise on physiological optics, Vol. III.
- Wacongne, C., Changeux, J., and Dehaene, S., A neuronal model of predictive coding accounting for the mismatch negativity. *The Journal of Neuroscience*, 32(11):3665–3678, 2012.