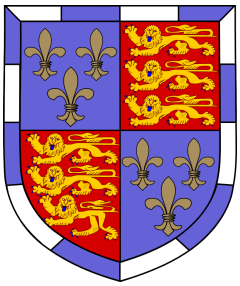
Time series modelling and inference with Bayesian Context Trees



Ioannis Papageorgiou

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

St John's College

June 2023

I would like to dedicate this thesis to my family.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, footnotes, tables and equations and has fewer than 150 figures.

Ioannis Papageorgiou
June 2023

Acknowledgements

First and foremost, I would like to immensely thank Professor Ioannis Kontoyiannis. His constant guidance, help and generous support have been crucial in completing this thesis. Apart from his academic guidance, I have been very lucky to find in him not only a mentor but also a good friend who was always there. So, many thanks for everything.

I would like to especially thank Professor Ramji Venkataramanan and Professor Simon Godsill for their tremendous help and for interesting conversations. Also, I am sincerely grateful to Professor Carl Rasmussen, whose supervision in my Master's thesis has been an inspiration to pursue a PhD degree, and to Professor Sumeetpal Singh who has also played an important role in my academic development. Special thanks to Dr Jossy Sayir for many interesting discussions and great times in the weekly group meetings.

I would like to really thank Valentinian Lungu for being a great and easy-to-work-with collaborator and co-author. Also, thanks for the interesting robots questions.

I would like to express my gratitude to everyone in St. John's College. They have really embraced me and made this place feel as a second home to me since arriving in the UK as an undergraduate. Also, I would like to thank the College for its financial support in the last years of my PhD, and EPSRC for its generous support in the previous years.

I have been privileged to make some wonderful friends while in Cambridge. Thank you Lampros Gavalakis, Dimitris Los, Dimitris Lolas and Giorgos Batzolis; life in Cambridge wouldn't have been that wonderful without your everyday company. Special thanks to Andreas Theocharous, Kostas Kavvadias and Antonis Ragkousis for our music room sessions, to Andreas Economou for exciting game nights, and to Sofia Kaberou for her support and a lot of interesting conversations. Many thanks to all members of our basketball team, and to Nick, Zoe, Elena, Alejandro and Stergios for their friendship in our undergraduate years.

My long-time friends have always been a great support and inspiration to me; so thank you Paris, George, Dimitris, Philip, Alex and Alex. Also, special thanks to Anna, Athina, Nikos and Jenny for their great company and support at the difficult times of the pandemic.

Lastly, but most importantly, I would like to thank my parents, grandparents and my entire family, without whose boundless love and support in any possible and impossible way I wouldn't have been able to complete this thesis.

Abstract

Time series arise all over the sciences and engineering, with numerous important applications in many different fields. In the 'Big Data Era', the tasks of time series modelling, inference and prediction have become more critical than ever. In this thesis, we introduce a collection of statistical ideas and algorithmic tools to build a general Bayesian framework for modelling and inference with time series data, which is found to be effective in a number of practical settings, including both discrete and real-valued observations.

For discrete-valued time series, we describe a novel Bayesian framework based on variable-memory Markov chains, called *Bayesian Context Trees* (BCT). This is a rich class of high-order Markov chains that admit parsimonious representations by allowing the memory of the process to depend on the values of the most recent observations. A general prior structure is introduced and a collection of methodological and algorithmic tools are developed, allowing for efficient, *exact* Bayesian inference in this setting. It is shown that the *evidence* (averaging over all models and parameters) can be computed exactly, and that the *a posteriori* most likely models can be precisely identified. The relevant algorithms have only linear complexity in the length of the data and can be updated sequentially, facilitating online prediction. We provide extensive experimental results illustrating the efficacy of our methods in a number of statistical tasks, as well as theoretical results that further justify their use.

The proposed approach is then extended to real-valued time series, where it is employed to develop a general hierarchical Bayesian framework for building mixture models. At the top level, a set of discrete contexts (or *'states'*) are extracted from quantised versions of the observations. The set of all relevant contexts are represented as a *context tree*. At the bottom level, a different real-valued time series model is associated with each state. This defines a very general framework that can be used in conjunction with any existing model class to build flexible and *interpretable* mixture models. We show that, again, effective computational tools can be developed allowing for efficient, exact Bayesian inference. The utility of the general framework is illustrated when autoregressive models are used as the base model, resulting in a nonlinear AR mixture, and when conditional heteroscedastic models are used, resulting in a flexible mixture model that gives a systematic way of modelling the well-known volatility asymmetries in financial data. The proposed methods are found to outperform the state-of-the-art techniques in various settings, both with simulated and real-world data.

Contents

Chapter 1

Introduction

Time series data arise all over the sciences and engineering, in a wide range of areas including economics, finance, neuroscience, genomics, the health sciences, the social sciences, meteorology, and ecology, among others. As such, the tasks of time series modelling, inference and prediction are critical tasks in statistics and machine learning, with countless applications in various settings. Although a number of previous approaches have been developed for these tasks throughout the years, the need for yet more flexible and effective model classes and inference methods in the 'Big Data Era' is more striking than ever. In this direction, broadly speaking, the goal of this thesis is to develop a general framework for modelling and inference with time series data that is applicable and effective in a number of practical settings.

Elaborating further, in view of the above general motivation and direction, the main purpose and contribution of this thesis will be to:

1. Develop flexible model classes that are appropriate and effective in various time series settings. The motivation leading to the development and the precise formulation of the model class in each case is to overcome the specific challenges that arise in the corresponding settings, as well as the limitations of previous approaches.

2. Introduce a Bayesian inference framework for the resulting model classes. This includes introducing an appropriate prior structure, together with associated methodological and algorithmic tools that allow for efficient and effective, *exact* Bayesian inference.

3. Evaluate the performance and the practical utility of the proposed methods, comparing with state-of-the-art approaches in real-world applications. The emphasis here is on the core statistical tasks of model selection, estimation and prediction, but more specialised tasks including entropy estimation and change-point detection are also considered.

4. Provide theoretical results and justifications that further support the use of the proposed methodology in practice.

Throughout this thesis we only consider discrete-time data, which is the most typical time series setting, in which the time series consist of a sequence of observations equally spaced in time. Depending on the possible values that the observations can take, time series can be divided in two broad categories: *discrete-valued* and *real-valued*. Both of these settings are of high practical importance, each one having numerous applications in various areas, and are therefore both studied in detail here. As the motivation and the specific tools developed in each case are slightly different, the rest of this thesis is roughly divided in two parts, considering respectively the discrete-valued and the real-valued case.

1.1 Discrete-valued time series

In this setting, we consider time series $\{X_n\}$ taking values in a finite *alphabet*, A, of size m. Without any loss of generality, we take $A = \{0, 1, \ldots, m-1\}$.

1.1.1 Motivation

Higher-order Markov chains are frequently the first and most natural modelling choice for discrete-valued time series with significant – apparent or suspected – temporal structure, especially when no specific underlying data generating mechanism can be assumed. But the description of a full Markov chain of order d taking values in a finite set of size m, requires the specification of $m^d(m-1)$ parameters, which makes the use of full Markov chains problematic in practice. As has been often noted (Bühlmann and Wyner, 1999; Raftery, 1985; Sarkar and Dunson, 2016), the dimension of the parameter space grows exponentially with the memory length, and the resulting model class lacks modelling wealth and flexibility. This lack of flexibility severely hinders, among other things, the important goal of balancing the bias-variance tradeoff between more complex models that fit the data closely, and simpler models that generalise well.

To address these issues and to offer better solutions to a wealth of related scientific and engineering problems that arise in connection with discrete time series, numerous approaches have been developed since the mid-1980s.

The mixture transition distribution (MTD) models introduced by Raftery (1985) and later generalised in Berchtold and Raftery (2002); Raftery and Tavaré (1994), allow for more parsimonious parametrisations of the transition distribution of some dth order Markov chains, as mixtures of first order transition matrices corresponding to different lags. MTD models make it possible to consider longer memory lengths d and to quantify the relative importance of different lags, but the resulting model class is still structurally poor.

A much more flexible class of Markov chain models with variable memory are the *tree sources* introduced in the celebrated work of Rissanen (1983a,b, 1986a). The gist of this approach is that the length of the memory that determines the transition probability of the chain can depend on the exact pattern of the most recently occurring symbols. Initially tree sources received a lot of attention in the information-theoretic literature in connection with data compression (Weinberger et al., 1994; Willems et al., 1995). One of their first applications outside information theory was by Ron et al. (1996), who introduced the notion of a probabilistic suffix tree (PST) as an effective structure for representing variable-memory chains. The PST point of view, along with the associated model selection technique *Learn-PSA*, have been used for bioinformatics problems and other machine learning tasks (Bejerano and Yona, 2001; Gabadinho and Ritschard, 2016).

In the statistics literature, tree-structured models were examined by Bühlmann (2000); Bühlmann and Wyner (1999). Their variable-length Markov chains (VLMCs) and the associated model selection tools are based, in part, on Rissanen's tree sources and his CONTEXT algorithm. The VLMC approach has been successful in applications (Busch et al., 2009; Mächler and Bühlmann, 2004) that include DNA modelling (Ben-Gal et al., 2005; Browning, 2006) and linguistics (Abakuks, 2012; Galves et al., 2012).

Tree sources and VLMC models group together certain patterns of past symbols that lead to the same conditional distribution for the chain, under the constraint that each such group consists of all patterns of length d that share a common suffix. A more general class of parsimonious Markov models, known as Sparse Markov Chains (SMC), arises when this constraint is removed. Originally introduced as "minimal Markov models" by García and González-López (2011), they were later examined in more detail by García and González-López (2017); Jääskinen et al. (2014); Xiong et al. (2016). But the lack of structure of this vast model class makes it difficult to identify appropriate models in practice.

A different class of high-order Markov models was recently introduced by Sarkar and Dunson (2016), who used conditional tensor factorisations (CTF) to give parsimonious representations of the full transition probability distribution (viewed as a high-dimensional tensor) of a Markov chain. These representations are based on earlier ideas on tensor factorisation for categorical regression (Yang and Dunson, 2016). CTF effectively shrinks the high-dimensional transition probability tensor to a lower-dimensional structure that can still capture high-order dependence. Unlike MTD, CTF accommodates complex interactions between the lags, and is accompanied by computational tools for Bayesian inference.

In this work we revisit the class of variable-memory Markov models. We introduce a new Bayesian framework for a version of these models, and we develop algorithmic tools that lead to very effective and efficient *exact* inference.

1.1.2 Overview of contributions

Bayesian Context Trees. In Chapter 2 we define a class of models for variable-memory Markov chains that admit natural representations as *context trees*. Given an alphabet A, and a maximal memory length $D \geq 0$, the class $\mathcal{T}(D)$ contains all variable-memory chains with values in A and memory no longer than D. A new family of prior distributions $\pi_D(T;\beta)$ on models $T \in \mathcal{T}(D)$ is introduced, indexed by a hyperparameter $\beta \in (0,1)$. Roughly speaking, π_D penalises larger and more complex models by an exponential amount. Given a model T, we place independent Dirichlet priors $\pi(\theta|T)$ on the associated parameters θ. We refer to the models in $\mathcal{T}(D)$ equipped with this prior structure as *Bayesian Context Trees* (BCT).

Inference Algorithms. Chapter 2 also contains our core methodological results in this setting, in the form of three exact inference algorithms for BCTs. First we show that a version of the Context Tree Weighting (CTW) algorithm (Willems et al., 1995; Willems, 1998) can be used to not only evaluate the marginal likelihoods $P(x|T) = \int P(x|\theta,T)\pi(\theta|T)d\theta$ of observations x with respect to models T, but also the *prior predictive likelihood* $P_D^*(x)$, averaging over all models and parameters,

$$P_D^*(x) := \sum_{T\in\mathcal{T}(D)} \pi_D(T;\beta)P(x|T) = \sum_{T\in\mathcal{T}(D)} \pi_D(T;\beta) \int P(x|\theta,T)\pi(\theta|T)d\theta. \tag{1.1}$$

The CTW algorithm computes $P_D^*(x)$ exactly, and its complexity is only linear in the length of the time series x. This quantity, also known as the *evidence* (MacKay, 2003), plays a pivotal role in Bayesian inference, as it is the normalising factor of the posterior distribution. In fact, the most basic obstacle to performing effective Bayesian inference is typically the inability to compute this term (Bernardo and Smith, 1994; Gelman et al., 2013). So, it is clear that the exact nature of the results produced by the CTW algorithm should facilitate the development of efficient methods for numerous core statistical tasks and related applications.

Furthermore, we describe the Bayesian context tree (BCT) algorithm and prove that it identifies the maximum *a posteriori* probability (MAP) model. This is a generalisation of the "context tree maximizing" algorithm of Willems and Volf (1994). Finally, we show that a new algorithm, the k-BCT algorithm, can be used to identify the top-k *a posteriori* most likely tree models, for any $k \geq 1$. Despite the fact that the class $\mathcal{T}(D)$ is vast, consisting of doubly-exponentially many models in D, the complexity of both the BCT and k-BCT algorithms is only *linear* in D and in the length of the time series. However, the complexity of k-BCT grows with k, limiting its practical applicability to moderate values of k.

In order to enable broader exploration of the posterior distributions $\pi(T|x)$ and $\pi(\theta,T|x)$, we also develop a new family of variable-dimension Markov chain Monte Carlo (MCMC) algorithms (Robert and Casella, 2004; Tierney, 1994) that obtain samples from $\pi(\theta,T|x)$.

Experimental Results. Closing Chapter 2, we present extensive experimental results illustrating the performance of the BCT framework on both simulated and real-world data from a wide range of applications. We focus on the fundamental statistical tasks of model selection, estimation and sequential prediction, and compare with state-of-the-art approaches. The proposed methods are found to be very effective in practice, outperforming the previous approaches both in terms of efficacy and computational requirements.

The BCT predictor in particular is found to have two significant advantages. The first is that the *posterior predictive distribution* can be computed exactly via the CTW algorithm, as $P_D^*(x_{n+1}|x_1,\ldots,x_n) = P_D^*(x_1,\ldots,x_{n+1})/P_D^*(x_1,\ldots,x_n)$. This way, the induced predictor is obtained by implicitly averaging over all models with respect to their exact posterior probabilities, thus avoiding the need to perform approximate model averaging via simulation or other numerical integration methods. The second advantage is that it can be updated sequentially in a very efficient manner, something that is very important in practice.

Theoretical Results. The Bayesian perspective adopted in this thesis is neither purely subjective nor purely objective. For example, we think of the model posterior distribution as a summary of the most accurate, data-driven representation of the regularities present in a given time series, but we also examine the frequentist properties of the resulting inferential procedures (Bernardo and Smith, 1994; Gelman et al., 2013).

In Chapter 3 we derive an alternative representation of the BCT posterior distribution on model space, $\pi(T|x)$, in a terms of a simple branching process. Apart from providing a practical way of obtaining independent and identically distributed (*i.i.d.*) samples from the posterior, this representations also allows to establish theoretical results that further justify the use of the BCT framework. In particular, we show that the posterior asymptotically almost surely concentrates on the "true" underlying model, and derive an explicit rate for this convergence as a function of the sample size. Analogous results are established in the case of out-of-class modelling, when the data are generated by a model outside the BCT class. Importantly, the limiting model is explicitly identified in this case, and is shown to be the model in $\mathcal{T}(D)$ that is closest in some sense to the true underlying model.

1.1.3 Additional results

Apart from the methodological contributions and the extensive experimental results focusing on the primary tasks of model selection, estimation and prediction, in Chapters 4 and 5 we also consider some more specialised statistical tasks for discrete-valued time series. More specifically, we consider Bayesian entropy estimation and change-point detection, both of which come with important applications in a number of practical settings.

Entropy Estimation. In Chapter 4 we consider the task of estimating the entropy rate from empirical data, an important and challenging problem that dates back to the original work of Shannon on the entropy of English text (Shannon, 1951). Since then, this task has received a lot of attention, perhaps most notably in connection with neuroscience (London et al., 2002; Nemenman et al., 2004; Strong et al., 1998), where the entropy rate plays a crucial role in efforts to describe and quantify the amount of information communicated between neurons. Many different approaches have been developed for this task throughout the years, including Lempel-Ziv (LZ) estimators (Ziv and Lempel, 1977), prediction by partial matching (PPM) (Cleary and Witten, 1984), the CTW algorithm (Gao et al., 2008), and block sorting methods (Cai et al., 2004); for an extensive review of the relevant literature, see Verdú (2019).

In contrast with most earlier work, here we adopt a fully-Bayesian approach. The BCT framework makes it possible to effectively sample from (and hence estimate) the actual posterior distribution of the entropy rate, which of course provides a much richer picture than the simple point estimates employed in most applications. The performance of the BCT entropy estimator is illustrated on both simulated and real-world data, where it is found to outperform several state-of-the-art methods. Also, we establish theoretical results that further justify our methodology: We show that the posterior asymptotically concentrates on the true underlying value of the entropy rate, and that it is asymptotically normal.

Change-Point Detection. In Chapter 5 we consider the tasks of segmentation and change-point detection with discrete-valued time series. Among others, the perhaps more notable application in this setting is the segmentation of genetic data, for which Hidden Markov Models (HMMs) have been used widely and with great success throughout the years (Churchill, 1989, 1992; Durbin et al., 1998; Lander et al., 2001). Here, we take a different, Bayesian approach, modelling each segment of the time series as a variable-memory Markov chain.

In Chapter 5, we introduce a Bayesian modelling framework for piecewise homogeneous time series. A uniform prior is placed on the number of change-points, and an "order statistics" prior is placed on their locations (Fearnhead, 2006; Green, 1995), which penalises short segments to avoid overfitting. Each segment is described by a BCT model, thus defining a new class of *piecewise homogeneous* variable-memory chains.

Having as a starting point the methodology of Chapter 2, we develop a new class of Bayesian methods for inferring the number and location of change-points in empirical data. A collection of MCMC algorithms is introduced for sampling directly and efficiently from the desired posterior distribution of the number and the location of the change-points, hence providing significant additional information on top of point estimates for them. The performance of our methods is illustrated on simulated and real-world data, where they are found to perform at least as well as state-of-the-art approaches.

1.2 Real-valued time series

In this setting we consider time series $\{X_n\}$ taking values in $\mathbb{R}$, which is the most natural setting in numerous real-world problems, opening the door to many more applications.

1.2.1 Motivation

Real-valued time series arise all over the sciences and engineering, in a wide range of applications from many different fields. This explains why numerous approaches have been developed throughout the years for the tasks of modelling, inference and prediction of real-valued time series. These include classical statistical methods (Box et al., 2015; Hamilton, 2020; Tsay, 2005), as well as modern machine learning (ML) techniques, notably matrix factorisations (Faloutsos et al., 2018; Yu et al., 2016), Gaussian processes (GPs) (Frigola, 2015; Rasmussen and Williams, 2006; Roberts et al., 2013), and neural networks (Alexandrov et al., 2020; Benidis et al., 2020; Zhang et al., 1998). Despite their popularity, there has not been conclusive evidence that in general the latter outperform the former in the time series setting (Ahmed et al., 2010; Makridakis et al., 2018a,b).

Motivated in part by the two well-known limitations of neural network models (Benidis et al., 2020), namely, their lack of interpretability and data efficiency, we propose a general class of flexible Bayesian mixture models that are both naturally *interpretable* and suitable for applications with limited training data. Also, we provide computationally efficient – *linear complexity* – algorithms for inference and prediction, offering another practical advantage compared to standard ML methods, which have greater computational requirements due to the heavier training involved (Makridakis et al., 2018b).

Roughly speaking, time series models can be broadly categorised in two classes, depending on the absence or presence of an underlying hidden state process. The first class includes the family of autoregressive (AR) and autoregressive integrated moving average (ARIMA) models, along with their extensions and generalisations, which directly model the mapping from previous to current observations. Besides the classical AR and ARIMA models, nonlinear AR models have been proposed, using both GPs (Candela et al., 2003; Girard et al., 2002; Murray-Smith and Girard, 2001; Roberts et al., 2013) and neural networks to model nonlinear dynamics. Feed-forward neural networks have been used since the 90s in this setting (Zhang et al., 1998), as in the neural network autoregressive (NNAR) model. More recently, recurrent neural networks (RNNs) have also been employed (Graves, 2013; Oreshkin et al., 2019; Salinas et al., 2020), with the DeepAR model of (Salinas et al., 2020) that uses long short-term memory (LSTM) cells (Hochreiter and Schmidhuber, 1997) being one of the most successful approaches.

The second class contains the family of state space models (SSMs) (Durbin and Koopman, 2012), which can describe more complex dynamics by allowing an underlying hidden state process (state transition) together with an observation model (emission). Classical approaches here include the linear Gaussian SSM, for which the Kalman filter (Kalman, 1960) can be used for inference, and exponential smoothing methods; see Hyndman et al. (2008) for a detailed review. Again, various extensions have been proposed by making the transition and/or emission equations nonlinear, using both GPs (Eleftheriadis et al., 2017; Frigola et al., 2013, 2014; Turner et al., 2010; Turner, 2012) and neural networks (Chung et al., 2015; Karl et al., 2016; Krishnan et al., 2017, 2015; Rangapuram et al., 2018; Zheng et al., 2017). However, inference in these settings becomes a quite challenging task, requiring sophisticated and computationally-intense approximate techniques, including particle MCMC (Andrieu et al., 2010; Doucet et al., 2001) and variational methods like stochastic gradient variational Bayes (SGVB) (Kingma and Welling, 2013; Rezende et al., 2014).

In this work, we introduce an intermediate Bayesian modelling approach that combines important features of both the above classes. We first identify meaningful discrete *states*, but these are *observable* rather than hidden, and given by the discretised values of some of the most recent samples. Then we associate a different time series model with each of these discrete *context-based* states, ultimately defining a very general class of hierarchical Bayesian mixture models for real-valued time series.

1.2.2 Overview of contributions

General Bayesian Mixture Models. In line with the above direction, in Chapter 6 we define a hierarchical Bayesian model, which at the top level selects the set of relevant states (that can be viewed as providing an adaptive partition of the state space), and at the bottom level associates an arbitrary time series model to each state. These collections of states (equivalently, the corresponding state space partitions) are naturally represented as discrete context-tree models, which are shown to admit a natural *interpretation* and to enable capturing important aspects of the structure present in the data. We refer to the resulting model class as the *Bayesian Context Trees State Space Model* (BCT-SSM).

Although we refer to the BCT-SSM as a 'model', it is in fact a very general framework for building Bayesian mixture models for time series, that can be used in conjunction with *any* existing model class as a base model. The resulting family of mixture models is much richer, more flexible and general than the class one starts with. For example, using any of the standard linear families (like the classical AR or ARIMA) leads to much more general models that can in fact capture highly nonlinear trends in the data. At the same time, the resulting mixture models are also easily interpretable.

Inference Algorithms. In Chapter 6, we show that in addition to resulting in the construction of flexible model classes, employing this particular type of observable state process (as opposed to a conventional hidden state) also makes it is possible to perform effective Bayesian inference. This is achieved by exploiting the structure of context-tree models and by extending the algorithms of Chapter 2, which were previously used only in the much more restricted setting of discrete-valued time series.

These tools make the BCT-SSM a powerful Bayesian framework that allows for *exact* and computationally very efficient Bayesian inference. In particular, the prior predictive likelihood, also known as the *evidence*, can be computed exactly, with all models and parameters integrated out. Also, the *a posteriori* most likely (MAP) partition (i.e., the MAP set of discrete states) can be identified, along with its exact posterior probability. Following the Bayesian approach here is particularly important, as it means that the data automatically select the set of relevant states. In contrast, maximum likelihood would overfit and select a large number of irrelevant states, with consequences both in the interpretation and the out-of-sample prediction performance of the fitted model.

Importantly, all the associated algorithms still have very low computational complexity, which is linear in the length of the time series. In fact, it is shown that we can perform inference in this much richer class of mixture models at essentially no additional cost compared to fitting a single model from the original base-model class. Also, our algorithms allow for efficient sequential updates, which are ideally suited for online forecasting, and provide an important practical advantage compared to other approaches.

Autoregressive Mixture Models. To illustrate the application of the general framework, we study in detail the case where AR models are used as building blocks for the BCT-SSM, with a different AR model associated to each discrete state. We refer to the resulting model class as the *Bayesian Context Trees Autoregressive* (BCT-AR) model. This is shown to be a flexible nonlinear AR mixture model that generalises popular AR mixtures, including the threshold AR (TAR) models (Tong, 2011; Tong and Lim, 1980) and the mixture AR (MAR) models of Wong and Li (2000). The BCT-AR model is found to outperform several state-of-the-art methods in simulated and real-world applications of nonlinear time series from economics and finance, both in terms of forecasting accuracy and computational requirements.

Volatility Modelling. As a second example, we associate conditional heteroscedastic models to each discrete state, something that is particularly important for financial applications. This results in a flexible mixture model that gives a systematic and powerful way of modelling the well-known asymmetric response in volatility due to positive and negative shocks, which is an important feature of financial time series (Tsay, 2005). Again, our method is found to outperform previous approaches that were developed to capture this effect.

1.3 Outline

In closing this Introduction we give an outline of the rest of this thesis, including a brief summary of the contents in each chapter.

Chapter 2. This chapter contains our core methodological contributions for discrete-valued time series, along with extensive experimental results for the fundamental statistical tasks of model selection, estimation and prediction. Its contents have been published before in:

- I. Kontoyiannis, L. Mertzanis, A. Panotopoulou, I. Papageorgiou, and M. Skoularidou. Bayesian Context Trees: Modelling and exact inference for discrete time series. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(4):1287–1323, 2022.

- I. Papageorgiou, I. Kontoyiannis, L. Mertzanis, A. Panotopoulou, and M. Skoularidou. Revisiting context-tree weighting for Bayesian inference. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 2906–2911, 2021.

Chapter 3. In this chapter, we derive an alternative representation of the BCT posterior in terms of a branching process, providing a practical way of sampling from it and allowing to establish theoretical results that further justify the BCT framework. This work appears in:

- I. Papageorgiou and I. Kontoyiannis. Posterior representations for Bayesian Context Trees: Sampling, estimation and convergence. *Bayesian Analysis*, to appear, 2023.

- I. Papageorgiou and I. Kontoyiannis. The posterior distribution of Bayesian Context-Tree models: Theory and applications. In *2022 IEEE International Symposium on Information Theory (ISIT)*, pages 702–707, 2022.

Chapter 4. In this chapter, we consider the challenging task of entropy estimation. Having as a starting point the BCT framework, we develop a fully-Bayesian methodology for this task, and provide extensive experimental and theoretical results for it. This work is also contained in:

- I. Papageorgiou and I. Kontoyiannis. Posterior representations for Bayesian Context Trees: Sampling, estimation and convergence. *Bayesian Analysis*, to appear, 2023.

- I. Papageorgiou and I. Kontoyiannis. Truly Bayesian entropy estimation. In *2023 IEEE Information Theory Workshop (ITW)*, pages 497–502, 2023.

Chapter 5. In this chapter, we consider the challenging task of change-point detection. We develop a Bayesian modelling framework based on piecewise homogeneous BCTs, along with a collection of tools for inferring the number and location of change-points. These contributions appear in:

- V. Lungu, I. Papageorgiou, and I. Kontoyiannis. Bayesian change-point detection via context tree weighting. In *2022 IEEE Information Theory Workshop (ITW)*, pages 125–130, 2022.
- V. Lungu, I. Papageorgiou, and I. Kontoyiannis. Change-point detection and segmentation of discrete data using Bayesian Context Trees. Submitted for publication, *arXiv preprint arXiv:2203.04341*, 2022.

Chapter 6. In this chapter, we develop a very general Bayesian framework for building flexible mixture models for real-valued time series that are based on context trees. The proposed framework can be combined with any existing model class as a base model, with a number of different examples and applications illustrating its practical utility. This work will appear in the following conference paper and preprint:

- I. Papageorgiou and I. Kontoyiannis. Context-tree weighting for real-valued time series: Bayesian inference with hierarchical mixture models. In *2023 IEEE International Symposium on Information Theory (ISIT)*, 2023.
- I. Papageorgiou and I. Kontoyiannis. The Bayesian Context Trees State Space Model: Interpretable mixture models for time series. *In preparation*, 2023.

Chapter 7. This final chapter contains some concluding remarks, including a brief summary of the contributions of this thesis as well as a discussion of some promising directions for future research.

Chapter 2

Bayesian Context Trees for Discrete Time Series

This chapter contains our core methodological contributions for discrete-valued time series. First we introduce the model class that we will be using, and then we develop an appropriate prior structure for it, together with effective tools for Bayesian inference. Finally, extensive experimental results are presented for the critical tasks of model selection, estimation and prediction, in which our methods are found to outperform the state-of-the-art approaches.

2.1 Variable-memory Markov chains

The distribution of a full dth order Markov chain $\{X_n\}$ with values in the finite state space, or *alphabet*, A, is identified by its conditional distributions,

$$\theta_s(a) := \Pr(X_n = a | X_{n-1} = x_{n-1}, X_{n-2} = x_{n-2}, \ldots, X_{n-d} = x_{n-d}), \qquad a \in A,$$

for every *context* $s = (x_{n-1}, x_{n-2}, \ldots, x_{n-d}) \in A^d$ of length d. If the alphabet A has size m, the description of these conditional distributions requires the specification of $(m-1)m^d$ parameters, which grows quickly with m and d. But suppose, for example, that $A = \{0,1,2\}$, and that when the most recent symbol x_{n-1} is a "1" the distribution of the next symbol is independent of the remaining values $x_{n-2}, ..., x_{n-d}$, that is, the conditional probabilities,

$$\Pr(X_n = a | X_{n-1} = 1, X_{n-2} = x_{n-2}, \ldots, X_{n-d} = x_{n-d}),$$

only depend on a. An example of how the distribution of such a 5th order, *variable-memory* Markov chain may be represented by a labelled tree is shown in Figure 2.1.

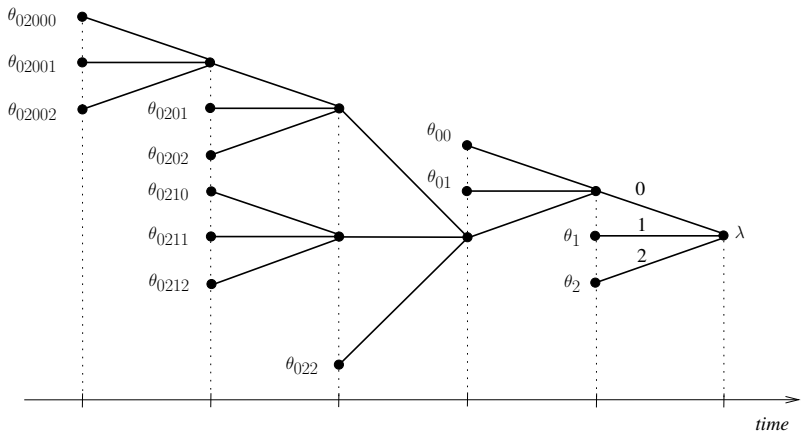


Figure 2.1: The tree model and parameters of a 5th order variable-memory chain.

Each leaf of the tree corresponds to a string s containing between one and five symbols from A, determined by the labels along the path from the root to that leaf. Whenever the string s consists of the same sequence of symbols as the most recent x_i's, the probability that the next value of the chain will be "a" is $\theta_s(a)$, given by the distribution θ_s which is associated to that leaf. For example,

$$\Pr(X_n = 2|X_{n-1} = 0, X_{n-2} = 1, X_{n-3} = 1, \ldots) = \Pr(X_n = 2|X_{n-1} = 0, X_{n-2} = 1) = \theta_{01}(2).$$

Note that, since the tree model for this chain has 13 leaves, instead of the full $2 \times 3^5 = 486$ parameters, it suffices to only specify $2 \times 13 = 26$.

2.1.1 Model description

Let $\{X_n\}$ be a dth order Markov chain with values in an alphabet of size m. Without loss of generality, we take $A = \{0, 1, \ldots, m-1\}$ throughout. The *model* describing $\{X_n\}$ as a variable-memory chain will always be represented by a tree as in the example above.

Let T be an m-ary tree of depth no greater than d, which is *proper*, in that, if a node in T is not a leaf, then it has exactly m children. This means that every length-d context $s := (x_{-1}, x_{-2}, \ldots, x_{-d}) \in A^d$ has a unique suffix $(x_{-1}, \ldots, x_{-j})$ which is a leaf of T, for some $j \leq d$. Let C be the function that maps each context $s \in A^d$ to a leaf of T. Viewing T as the collection of its leaves, it is the range of C,

$$C : A^d \to T \subset \bigcup_{i=0}^{d} A^i,$$

with the convention that $A^0 = \{\lambda\}$ contains only the *empty string* λ.

For indices $i \leq j$, we write X_i^j for the vector of random variables $(X_i, X_{i+1}, \ldots, X_j)$ and similarly $x_i^j \in A^{j-i+1}$ for a string $(x_i, x_{i+1}, \ldots, x_j)$ representing a particular realisation of these random variables. Then the Markov property for a variable-memory chain $\{X_n\}$ with model T takes the form:

$$P(x_1^n | x_{-d+1}^0) := \Pr(X_1^n = x_1^n | X_{-d+1}^0 = x_{-d+1}^0) = \prod_{i=1}^{n} P(x_i | x_{i-d}^{i-1}) = \prod_{i=1}^{n} P(x_i | C(x_{i-d}^{i-1})). \quad (2.1)$$

The complete description of the distribution of $\{X_n\}$, in addition to the model T, requires the specification of a set of *parameters*, $\theta = \{\theta_s \,;\, s \in T\}$. To every $s \in T$ we associate a probability vector,

$$\theta_s := (\theta_s(0), \theta_s(1), \ldots, \theta_s(m-1)),$$

where the $\theta_s(j)$ are nonnegative and sum to one. Then (2.1) can be written as,

$$P(x_1^n | x_{-d+1}^0) = \prod_{i=1}^{n} \theta_{C(x_{i-d}^{i-1})}(x_i) = \prod_{s \in T} \prod_{j \in A} \theta_s(j)^{a_s(j)}, \quad (2.2)$$

where each element $a_s(j)$ of the *count vector* $a_s = (a_s(0), a_s(1), \ldots, a_s(m-1))$ is,

$$a_s(j) \;:=\; \#\text{ times symbol } j \in A \text{ follows context } s \text{ in } x_1^n. \quad (2.3)$$

2.2 Bayesian modelling and inference

Throughout this work, we consider every dth order, variable-memory chain $\{X_n\}$ as described by a (proper) tree model T with associated parameters $\theta = \{\theta_s \,;\, s \in T\}$. Here we define a family of prior distributions on (T, θ).

2.2.1 Prior structure

Model prior. For a maximal depth $D \geq 0$ and an alphabet $A = \{0, 1, \ldots, m-1\}$ of size $m \geq 2$, let $\mathcal{T}(D)$ denote the collection of all (proper) tree models T on A with depth no greater than D. Given an arbitrary $\beta \in (0,1)$, we define the prior distribution,

$$\pi(T) := \pi_D(T; \beta) := \alpha^{|T|-1} \beta^{|T| - L_D(T)}, \qquad T \in \mathcal{T}(D), \quad (2.4)$$

where $\alpha := (1-\beta)^{1/(m-1)}$, $|T|$ denotes the number of leaves of T, and $L_D(T)$ denotes the number of leaves of T at depth D. As it is described below in more detail, the essence of this prior is to penalise the larger trees by an exponential amount.

The following lemma, proved in Appendix B.1, states that (2.4) does indeed define a proper probability distribution.

Lemma 2.1. *For any $D \geq 0$ and any $\beta \in (0,1)$: $\sum_{T \in \mathcal{T}(D)} \pi_D(T;\beta) = 1$.*

Choice of β. The first factor, $\alpha^{|T|-1}$, in the definition of the model prior $\pi_D(T;\beta)$ is the more important and easier one to interpret, showing that larger models are penalised by an exponential amount, while the second factor, $\beta^{|T|-L_D(T)}$, adds a less intuitive penalty only to the leaves at depth strictly smaller than D.

Consider two models $S, T \in \mathcal{T}(D)$ such that S is a subtree of T. If T is produced by adding a single branch of m nodes to one of the leaves, s, say, of S, and s is at depth $D-2$ or smaller, then $|T| = |S| + m - 1$ and $L_D(T) = L_D(S)$, so that,

$$\frac{\pi_D(T;\beta)}{\pi_D(S;\beta)} = (1-\beta)\beta^{m-1},$$

which, as desired, is always less than one. But if T is produced from S by adding a branch of m nodes to a node s at depth $D-1$, then $L_D(T) = L_D(S) + m$, and,

$$\frac{\pi_D(T;\beta)}{\pi_D(S;\beta)} = \frac{1-\beta}{\beta}.$$

This is strictly decreasing in β, and for $\beta < 1/2$ it is greater than one, while for $\beta > 1/2$ it is smaller than 1. Therefore, $\pi_D(T;\beta)$ penalises larger trees by an *exponential* amount as long as $\beta \geq 1/2$, and larger values of β make the penalisation more severe.

Also, for larger alphabet sizes, $\alpha = (1-\beta)^{1/(m-1)}$ becomes very close to 1 and the second factor dominates, an effect which is unintuitive and less desirable. Therefore, in practice we will always take the *default* value to be $\beta \approx 1 - 2^{-m+1}$, so that $\alpha \approx 1/2$, unless there are specific reasons for a different choice.

A more explicit interpretation of β is given in Chapter 3, where an alternative description of the prior $\pi_D(T;\beta)$ is given in terms of a simple branching process.

Prior on θ. Given a model $T \in \mathcal{T}(D)$, we place an independent Dirichlet prior with parameters $(1/2, 1/2, \ldots, 1/2)$ on each θ_s so that, $\pi(\theta|T) = \prod_{s \in T} \pi(\theta_s)$, where,

$$\pi(\theta_s) = \pi(\theta_s(0), \theta_s(1), \ldots, \theta_s(m-1)) = \frac{\Gamma(m/2)}{\pi^{m/2}} \prod_{j=0}^{m-1} \theta_s(j)^{-\frac{1}{2}} \propto \prod_{j=0}^{m-1} \theta_s(j)^{-\frac{1}{2}}. \qquad (2.5)$$

Although we take the parameter vector of the Dirichlet prior for all θ_s to be $(1/2, \ldots, 1/2)$, corresponding to the Jeffreys prior (Jeffreys, 1998; Robert et al., 2009), the extension of all our results to arbitrary Dirichlet parameters is straightforward as outlined in Appendix B.1.

2.2.2 Marginal likelihood and prior predictive likelihood

Likelihood. Given the model T and its associated parameters $\theta = \{\theta_s \,;\, s \in T\}$, from (2.2) it is easy to obtain the *likelihood*,

$$P(x_1^n | x_{-d+1}^0, \theta, T) = \prod_{s \in T} \prod_{j=0}^{m-1} \theta_s(j)^{a_s(j)}, \tag{2.6}$$

where a_s are the count vectors defined in (2.3). By convention, when we write $\sum_{s \in T}$ or $\prod_{s \in T}$, we take the corresponding sum or product over all the *leaves* s of the tree, not all its nodes. Also, in order to avoid cumbersome notation, in what follows we often write x for the entire time series x_1^n and suppress the dependence on its initial context x_{-d+1}^0, so that, for example,

$$P(x|T,\theta) := P(x_1^n | x_{-d+1}^0, \theta, T).$$

Marginal likelihood. A useful property of the BCT framework is that the parameters θ can easily be integrated out, so that the *marginal likelihoods* $P(x|T)$ can be expressed in closed form. This result, stated without proof in Lemma 2.2, is based on a standard computation.

Lemma 2.2. *The marginal likelihood $P(x|T)$ of the observations x given a model T is,*

$$P(x|T) = \int P(x,\theta|T)d\theta = \int P(x|\theta,T)\pi(\theta|T)d\theta = \prod_{s \in T} P_e(a_s),$$

where the count vectors $a_s = (a_s(0), a_s(1), \ldots, a_s(m-1))$ are defined in (2.3) *and the* estimated probabilities *$P_e(a_s)$ are defined by,*

$$P_e(a_s) := \frac{\prod_{j=0}^{m-1} [(1/2)(3/2)\cdots(a_s(j)-1/2)]}{(m/2)(m/2+1)\cdots(m/2+M_s-1)}, \tag{2.7}$$

where $M_s := a_s(0) + a_s(1) + \cdots + a_s(m-1)$, with the convention that any empty product is taken to be equal to 1.

Prior predictive likelihood. In terms of inference, one of the main objects of interest is the model posterior distribution,

$$\pi(T|x) = \frac{P(x|T)\pi(T)}{P(x)},$$

where the denominator $P(x)$ is its normalising constant, also known as the *prior predictive likelihood*, or *evidence* (MacKay, 2003). Recalling (1.1), this term is given by,

$$P(x) = P_D^*(x) := \sum_{T \in \mathcal{T}(D)} \pi_D(T;\beta) P(x|T) = \sum_{T \in \mathcal{T}(D)} \pi_D(T;\beta) \int P(x|\theta,T)\pi(\theta|T)d\theta.$$

The difficulty in computing $P_D^*(x)$ comes, in part, from the fact that the class $\mathcal{T}(D)$ of variable-memory models is enormously rich, even for moderate (or even small) alphabet sizes m and maximal depths D. The size of $\mathcal{T}(D)$ grows doubly exponentially in D; in fact, a simple computation shows that,

$$|\mathcal{T}(D)| \geq \sum_{d=0}^{D-1} 2^{m^d} - D. \tag{2.8}$$

In the next section, we describe three exact inference algorithms for the class of Bayesian context trees. We show that, perhaps somewhat surprisingly, $P_D^*(x)$ can be computed *exactly* and in a very efficient manner, and that the mode of the posterior $\pi(T|x)$ as well as the next few most likely models can be explicitly identified.

2.3 Inference algorithms

This section contains our core methodological contributions, which form the basis for most of the statistical and learning tasks that can be performed within the BCT framework. In particular, we develop a collection of methodological and algorithmic tools which are proven to allow for exact and very efficient Bayesian inference.

The CTW algorithm is first described in Section 2.3.1, and it is proved that it indeed computes the prior predictive likelihood $P_D^*(x)$ of a time series x. Section 2.3.2 introduces the BCT algorithm, which identifies the maximum *a posteriori* probability (MAP) tree model T_1^* given the data x, along with its exact posterior probability $\pi(T_1^*|x)$. A more general algorithm, the k-BCT algorithm, is described in Section 2.3.3. This is proven to identify not just the mode T_1^* of the posterior $\pi(T|x)$ on model space, but the top-k *a posteriori* most likely tree models, $T_1^*, T_2^*, \ldots, T_k^*$, for any $k \geq 1$. All three algorithms are implemented in the publicly available `R` package `BCT` (Papageorgiou et al., 2022).

We also discuss how the posterior probability $\pi(T|x)$ of any model T can be easily computed, and we identify the full conditional density of the parameters, $\pi(\theta|x,T)$. In order to explore the posterior distribution further, a family of variable-dimension MCMC samplers are developed in Section 2.3.4. Finally, in Section 2.3.5 we describe how the CTW, BCT and k-BCT algorithms can all be updated sequentially, and show that their complexity is only linear in the length of the time series x. In particular, the fact that the CTW algorithm can be implemented in a sequential fashion means that it can be effectively used for efficient online prediction, some which is very important in practice; cf. Section 2.6.

2.3.1 The Context Tree Weighting (CTW) algorithm

The CTW algorithm takes as input: The size of the alphabet $m \geq 2$, the maximum context depth $D \geq 0$, a time series of observations x_1^n together with the initial context x_{-D+1}^0, all taking values in the alphabet A, and the value of the prior parameter $\beta \in (0,1)$.

Algorithm description. The CTW algorithm executes the following steps:

(i) Build an m-ary tree, T_{MAX}, whose leaves are all the contexts x_{i-D}^{i-1}, $i = 1, 2 \ldots, n$, that appear in the observations x_{-D+1}^n. If some node s of T_{MAX} is at depth $d < D$ and some but not all of its children are in T_{MAX}, then add all its remaining children as well, so that T_{MAX} is a proper tree.

(ii) Compute the count vector a_s as in (2.3), at *every* node s of the tree T_{MAX} (not only at the leaves), so that a_s will be the all-zero vector for the additional leaves included in the last step of (i).

(iii) Compute the *estimated probability* $P_{e,s} := P_e(a_s)$ given by (2.7), at each node s of the tree T_{MAX}, with the convention that $P_e(a_s) = 1$ when a_s is the all-zero count vector.

(iv) Write sj for the concatenation of context s and symbol j, corresponding to the jth child of node s. Starting at the leaves and proceeding recursively towards the root, at each node s of the tree T_{MAX} compute the *mixture* (or *weighted*) *probabilities*:

$$P_{w,s} := \begin{cases} P_{e,s}, & \text{if } s \text{ is a leaf,} \\ \beta P_{e,s} + (1-\beta) \prod_{j=0}^{m-1} P_{w,sj}, & \text{otherwise.} \end{cases} \tag{2.9}$$

(v) Output the weighted probability $P_{w,\lambda}$ at the root λ.

Theorem 2.1. *The weighted probability $P_{w,\lambda}$ at the root λ computed by the CTW algorithm is exactly the prior predictive likelihood of the observations,*

$$P_{w,\lambda} = P_D^*(x) = \sum_{T \in \mathcal{T}(D)} \pi_D(T;\beta) \int_\theta P(x_1^n | x_{-D+1}^0, \theta, T) \pi(\theta|T) d\theta,$$

where the sum is over all proper context-tree models of depth no greater than D.

Proof. Theorem 2.1 is proved in Appendix B.1. Its proof is by induction and is based on some structural properties of the BCT prior $\pi_D(T;\beta)$. □

2.3.2 The Bayesian Context Tree (BCT) algorithm

The BCT algorithm takes the same input as CTW and executes the following steps:

(i) Build the m-ary tree T_{MAX} and compute the count vectors a_s and the estimated probabilities $P_{e,s} = P_e(a_s)$ at all nodes s of T_{MAX}, as in steps (i)–(iii) of CTW.

(ii) Starting at the leaves and proceeding recursively towards the root, at each node s of the tree T_{MAX} compute the *maximal probabilities*,

$$P_{m,s} := \begin{cases} P_{e,s}, & \text{if } s \text{ is a leaf at depth } D, \\ \beta, & \text{if } s \text{ is a leaf at depth } < D, \\ \max\left\{\beta P_{e,s},\ (1-\beta)\prod_{j=0}^{m-1} P_{m,sj}\right\}, & \text{otherwise.} \end{cases} \tag{2.10}$$

(iii) Starting at the root and proceeding recursively with its descendants, for each node s: If the maximum in (2.10) can be achieved by the first term, then prune all its descendants from the tree T_{MAX}; otherwise, repeat the same process at each of the m children of s.

(iv) After all nodes have been exhausted in (iii), output the resulting tree T_1^* and the maximal probability at the root $P_{m,\lambda}$.

Theorem 2.2. *For all $\beta \geq 1/2$, the tree T_1^* produced by the BCT algorithm is the MAP tree model (or one of the MAP tree models, in case the maximum below is not uniquely achieved),*

$$\pi(T_1^*|x) = \max_{T \in \mathcal{T}(D)} \pi(T|x), \tag{2.11}$$

and the maximal probability at the root satisfies,

$$P_{m,\lambda} = P(x|T_1^*)\pi_D(T_1^*;\beta) = P(x, T_1^*). \tag{2.12}$$

Proof. This theorem is also proved in Appendix B.1. □

2.3.3 The top-k Bayesian Context Trees (k-BCT) algorithm

Although the k-BCT algorithm is conceptually a natural generalisation of BCT, the precise description of its most efficient implementation is quite lengthy. So, for the sake of both clarity and brevity, we first describe here a less practical, idealised version of k-BCT, which is conceptually identical with its the actual practical version and only differs from it in the initialisation step of the leaves at depth $d < D$. The actual practical algorithm is given in Appendix A, along with a discussion of its implementation complexity.

Algorithm description. The idealised k-BCT algorithm takes the same input as CTW, together with the number $k \geq 1$ of the *a posteriori* most likely models to be determined. It executes the following steps:

(i) Let T_{MAX} be the *complete* m-ary tree at depth D (this is the "idealised" part), and compute the count vectors a_s and the estimated probabilities $P_{e,s} = P_e(a_s)$ at all nodes s of T_{MAX} as in steps (ii) and (iii) of CTW.

(ii) Starting at the leaves and proceeding towards the root, at each node s compute a list of k maximal probabilities $P_{m,s}^{(i)}$ and k position vectors $c_s^{(i)} = (c_s^{(i)}(0), c_s^{(i)}(1), \ldots, c_s^{(i)}(m-1))$, for $i = 1, 2, \ldots, k$, where each $c_s^{(i)}(j)$ is an integer between 0 and k, recursively, as follows.

(iia) At each leaf s, we let $P_{m,s}^{(1)} = P_{e,s}$ and $c_s^{(1)} = (0, 0, \ldots, 0)$, where the all-zero vector $c_s^{(1)}$ indicates that $P_{m,s}^{(1)}$ corresponds to the value of $P_{e,s}$ and does not depend on the children of s (since there are none). For $i = 2, 3, \ldots, k$, we leave $P^{(i)}$ and $c_s^{(i)}$ undefined.

(iib) At each node s having only m descendants (which are necessarily leaves), we compute the two probability-position vector pairs $\beta P_{e,s}$, $(0, 0, \ldots, 0)$ and $(1-\beta)\prod_{j=0}^{m-1} P_{m,sj}^{(1)}$, $(1, 1, \ldots, 1)$ (where the all-1 vector indicates that the latter probability only depends on the first maximal probability of each of the children), and sort them as $P_{m,s}^{(1)}$, $c_s^{(1)}$ and $P_{m,s}^{(2)}$, $c_s^{(2)}$ in order of decreasing probability. For $i = 3, 4, \ldots, k$, we leave $P^{(i)}$ and $c_s^{(i)}$ undefined.

(iic) A general internal node s has m children, where each child sj has a list of k_j (for some $1 \leq k_j \leq k$) probability-vector pairs $P_{m,sj}^{(i)}, c_{sj}^{(i)}$, $1 \leq i \leq k_j$. We compute the probability $\beta P_{e,s}$ with associated position vector $(0, 0, \ldots, 0)$, and all possible probability-position vector pairs,

$$(1-\beta)\prod_{j=0}^{m-1} P_{m,sj}^{(i_j)}, \quad (i_0, i_1, \ldots, i_{m-1}), \tag{2.13}$$

for all possible combinations of indices $1 \leq i_j \leq k_j$ for $0 \leq j \leq m-1$. We then sort these $k' = 1 + k_0 \times k_1 \times \cdots \times k_{m-1}$ probabilities in order of decreasing probability, and rename the top-k of them as $P_{m,s}^{(i)}$, for $i = 1, 2, \ldots, k$, together with their associated position vectors $c_s^{(i)}$. [Of course, if $k' < k$, after sorting we leave the remaining $k - k'$ probability-position vector pairs undefined.]

(iii) Having determined all maximal probabilities $P_{m,s}^{(i)}$ for all nodes s and $1 \leq i \leq k$, we now determine the "top-k" trees $T_1^*, T_2^*, \ldots, T_k^*$ from the corresponding position vectors $c_s^{(i)}$.

For each i we repeat the following process, starting at the root and proceeding until all available nodes of the tree T_{MAX} have been exhausted.

(*iiia*) *Depth* $d = 0$. At the root node λ, we examine $c_\lambda^{(i)}$. If it is the all-zero vector, then T_i^* is the tree consisting of the root node only. Otherwise, we add to T_i^* the branch of m children starting at the root, and proceed to examine each of the nodes corresponding to the m children recursively.

(*iiib*) *Depth* $d = 1$. Reaching node $s = j$ corresponding to the jth child of the root, means that $t = c_\lambda^{(i)}(j)$ is nonzero. We examine $c_s^{(t)}$: If it is the all-zero vector, then we prune from T_i^* all the descendants of s and move to the next unexamined node; otherwise, we add to T_i^* the branch of m children starting at s, and proceed to examine each of the nodes corresponding to the m children recursively.

(*iiic*) *General depth* $1 \le d \le D-1$. Reaching a node sj at depth d from its parent node s means that we decided to visit sj because $t = c_s^{(u)}(j)$ is nonzero for the appropriate index u (corresponding to the position vector $c_s^{(u)}$ that was examined at node s). We examine $c_{sj}^{(t)}$: If it is the all-zero vector, then we prune from T_i^* all the descendants of sj and move to the next unexamined node; otherwise, we add to T_i^* the branch of m children starting at sj, and proceed to examine each of the nodes corresponding to the m children recursively.

(*iiid*) *Depth* $d = D$. Reaching a node s at depth D means we have reached a leaf of T_{MAX}, so we simply add s to T_i^* and proceed to the next unexamined node.

(*iv*) Output the k resulting trees T_i^* and the k maximal probabilities at the root, $P_{m,\lambda}^{(i)}$, for $i = 1, 2, \ldots, k$.

Theorem 2.3. *For any* $\beta \ge 1/2$*, the trees* $T_1^*, T_2^*, \ldots, T_k^*$ *produced by the k-BCT algorithm are the k* a posteriori *most likely tree models. For each* $i = 1, 2, \ldots, k$*,*

$$\pi(T_i^*|x) = \max_{T \in \mathcal{T}(D)}{}^{(i)} \pi(T|x), \tag{2.14}$$

where $\max_{t \in \mathcal{T}}^{(i)} f(t)$ *of a function* f *defined on a discrete set of arguments* $t \in \mathcal{T}$ *denotes the ith largest value of* f*. Moreover, the ith maximal probability at the root satisfies,*

$$P_{m,\lambda}^{(i)} = P(x|T_i^*)\pi_D(T_1^*;\beta) = P(x, T_i^*), \quad i = 1, 2, \ldots, k.$$

Proof. Theorem 2.3 is also proved in Appendix B.1. We note that, if some of the maxima in (2.11) are not uniquely achieved, then $T_1^*, \ldots, T_k^*$ are one of the equivalent collections of k *a posteriori* most likely models. □

Computation of posterior probabilities

In addition to the prior predictive likelihood $P_D^*(x) = P_{w,\lambda}$ computed by CTW, and the *a posteriori* most likely models identified by the BCT and k-BCT algorithms, there are several other useful quantities that can easily be obtained through the results of these algorithms.

Model posterior probabilities. For any model $T \in \mathcal{T}(D)$,

$$\pi(T|x) = \frac{P(x|T)\pi(T)}{P_D^*(x)} = \frac{\pi(T)\prod_{s\in T} P_e(a_s)}{P_{w,\lambda}}, \tag{2.15}$$

where $P_D^*(x) = P_{w,\lambda}$ is the prior predictive likelihood computed by CTW, and the numerator is given as in Lemma 2.2. If a model T has a leaf s that is not included in the tree T_{MAX} generated by CTW, we follow the convention of setting $P_e(a_s) = 1$.

Full conditional density of θ. Conditional on a model T and observations x, the density of the parameters $\theta = \{\theta_s \,;\, s \in T\}$ is,

$$\pi(\theta|x,T) \propto P(x|\theta,T)\pi(\theta|T) \propto \left(\prod_{s\in T}\prod_{j=0}^{m-1} \theta_s(j)^{a_s(j)}\right)\left(\prod_{s\in T}\prod_{j=0}^{m-1} \theta_s(j)^{-\frac{1}{2}}\right),$$

where we have used the definition of the prior on θ in (2.5) and the expression for the likelihood (2.6). Therefore, the full conditional density $\pi(\theta|x,T)$ is the product of Dirichlet densities,

$$\pi(\theta|x,T) \sim \prod_{s\in T} \text{Dir}(a_s(0)+1/2, a_s(1)+1/2, \ldots, a_s(m-1)+1/2). \tag{2.16}$$

2.3.4 MCMC samplers

The k-BCT algorithm can identify the k *a posteriori* most likely models, including in cases where the posterior $\pi(T|x)$ is multimodal, but its computational complexity grows with k, limiting its practical applicability to moderate values of k. In this section, we describe a family of effective variable-dimension MCMC samplers (Robert and Casella, 2004) that make it possible to explore $\pi(T|x)$ further, and to sample from the posterior $\pi(\theta,T|x)$ jointly on models and parameters.

The random walk (RW) sampler for $\pi(T|x)$ is a Metropolis-Hastings (MH) algorithm (Hastings, 1970; Metropolis et al., 1953) with a proposal distribution that, at each step, either adds or removes an m-tuple of leaves from the current tree. It takes as input the same parameters as CTW together with: An initial model $T^{(0)} \in \mathcal{T}(D)$ and the tree T_{MAX} along with the estimated probabilities $P_{e,s}$ at all its nodes, as computed by CTW.

RW sampler. It executes the following steps at each MCMC iteration $t = 1, 2, \ldots, N-1$:

(i) Given $T^{(t)}$, propose a new tree T' as follows:

  (a) If $T^{(t)}$ is the empty tree $\Lambda := \{\lambda\}$ consisting only of the the root node, let T' be the complete tree at depth 1, $T_c(1)$.

  (b) If $T^{(t)}$ is the complete m-ary tree at depth D, $T_c(D)$, choose uniformly at random one of the internal nodes s at depth $D-1$ and let T' be the same as $T^{(t)}$ but with the m children of s removed.

  (c) Otherwise, with probability 1/2 decide to propose a larger tree T', formed by choosing uniformly at random one of the leaves of $T^{(t)}$ at depths $\leq D-1$ and adding its m children to form T';

  (d) Or, with probability 1/2 decide to propose a smaller tree T', formed by choosing uniformly at random one of the internal nodes s of $T^{(t)}$ that only have m descendants and removing the m leaves that stem from s, to form T'.

(ii) Either accept T' and set $T^{(t+1)} = T'$, or reject it and set $T^{(t+1)} = T^{(t)}$, with corresponding probabilities $\alpha(T^{(t)}, T') = \min\{1, r(T^{(t)}, T')\}$ and $1 - \alpha(T^{(t)}, T')$, respectively; explicit expressions for the accept-reject ratios $r(T, T')$ follow easily from standard MH methodology (Chib and Greenberg, 1995), and are given in Appendix B.4.

The jump sampler for $\pi(T|X)$ is a modification of the RW sampler, which, in addition to nearest neighbour moves, also allows for jumps to any one of the k most likely models. This way we overcome the common difficulty of RW samplers to move between separated modes of multimodal posterior distributions.

Jump sampler. It takes as input the same parameters as the the RW sampler, and also the value of a jump parameter $p \in (0,1)$ and the collection $\mathcal{T}^* = \{T_1^*, \ldots, T_k^*\}$ of the top-k trees computed by k-BCT. It executes the following steps at each iteration $t = 1, 2, \ldots, N-1$:

(i) Given $T^{(t)}$, propose a new tree T' as follows:

  (a) With probability $(1-p)$, propose a new tree T' as in steps (ia)–(id) of the RW sampler;

  (b) Or, with probability p, propose a jump move: Let T' be one of the top-k trees T_i^*, uniformly chosen from $\mathcal{T}^*$.

(ii) Either accept T' and set $T^{(t+1)} = T'$, or reject it and set $T^{(t+1)} = T^{(t)}$, with corresponding probabilities $\alpha(T^{(t)}, T') = \min\{1, r(T^{(t)}, T')\}$ and $1 - \alpha(T^{(t)}, T')$, where the ratios $r(T, T')$ are given explicitly in Appendix B.4.

Note that a jump move to one of the top-k models T_i^* may be proposed from any state $T^{(t)}$ of the sampler, but it only has a nonzero probability of being accepted if $T^{(t)}$ itself is either one of the T_i^* or a neighbour of one of them. This suggests that the jump parameter should be chosen so that $(1-p)$ is not too small; in all of our experiments below we take $p = 1/2$.

MCMC convergence. The target distribution of both the RW and jump samplers is the posterior $\pi(T|x)$, $T \in \mathcal{T}(D)$, on model space. Unlike with most MCMC samplers used in Bayesian inference, here we can in fact compute the value of the target distribution $\pi(T|x)$ exactly, for any specific $T \in \mathcal{T}(D)$, as noted in (2.15). So, knowing the posterior probabilities $\pi(T|x)$ precisely (not only up to a multiplicative constant) means that it is easy to obtain a good first indication of whether the sampler has converged, or at least whether it has spent the "right" amount of time in the important areas of the support of $\pi(T|x)$ near its mode(s): Simply compute the frequency of each of the top-k models T_i^* in the MCMC sample, and compare it with its actual posterior probability $\pi(T_i^*|x)$.

Although we have not observed convergence issues in any of our experiments, we note that more sophisticated MCMC methods can also be used, e.g., tempering the likelihood (Robert and Casella, 2004) or using a Wang-Landau-style algorithm to force the sampler to spend a specified proportion of time at models of each depth (Atchade and Liu, 2004).

Joint sampler. Being able to obtain MCMC samples $\{T^{(t)}\}$ for $\pi(T|x)$, and knowing the full conditional density $\pi(\theta|x,T)$ of the parameters explicitly as in (2.16), it is simple to obtain a corresponding sequence of samples $\{(\theta^{(t)}, T^{(t)})\}$ for the posterior $\pi(\theta, T|x)$ jointly on models and parameters. This can be done by drawing a conditionally independent sample $\theta^{(t)} \sim \pi(\theta|x,T^{(t)})$ at each MCMC iteration.

2.3.5 Sequential updates and computational complexity

As more observations become available, the results of all the three exact inference algorithms can be updated sequentially, in a very efficient manner. This facilitates their online use in applications where it is important that data be processed sequentially rather than in large blocks, as in the critical task of prediction; cf. Section 2.6.

For CTW, having computed $P_D^*(x_1^n|x_{-D+1}^0)$ and given an additional sample x_{n+1}, the new prior predictive likelihood $P_D^*(x_1^{n+1}|x_{-D+1}^0)$ can be obtained in $O(D)$ operations. In specific, if we denote $s_D, s_{D-1}, \ldots, s_0$ to be the contexts of lengths $D, D-1, \ldots, 0$ immediately preceding x_{n+1}, so that in particular $s_D = x_{n-D+1}^n$ and $s_0 = \lambda$, then updates need to be performed only at these $D+1$ nodes in the tree T_{MAX}. This can be done by executing steps (ii)–(iv) of CTW only at these nodes, as follows.

Sequential updates. If $x_{n+1} = j$, at each of the nodes $s_D, s_{D-1}, \ldots, s_0$ in the tree T_{MAX} already constructed, in that order and *only* there:

(ii') Update $a_s(j)$ and M_s by increasing each of their values by 1.

(iii') Update $P_{e,s} = P_e(a_s)$ by multiplying its earlier value by $(a_s(j) - 1/2)/(m/2 + M_s - 1)$, using the updated values of $a_s(j)$ and M_s.

(iv') Re-compute the probability $P_{w,s}$.

The required result $P_D^*(x_1^{n+1}|x_{-D+1}^0)$ is the (updated) weighted probability $P_{w,\lambda}$ at the root. The corresponding update rules for the BCT and k-BCT algorithms are analogous and easy to determine and implement.

Computational complexity. Next we briefly discuss the implementation complexity of the three algorithms in Sections 2.3.1–2.3.3, as a function of the parameters n, m, D, and k. The complexity of CTW is *linear* in each of n, m and D, and in fact it is of order $O(nmD)$.

To see this, observe that in step (i), for each x_i, $1 \le i \le n$, a new node is created for each of the $(D+1)$ contexts of x_i, of lengths $0, 1, \ldots, D$. This requires $O(nD)$ operations and produces the tree T_{MAX} which has no more than $nD+1$ nodes. The second and third steps, where the count-vectors a_s and the probabilities $P_e(a_s)$ are computed, can be integrated into the first one. For each i, when we visit each of the $(D+1)$ contexts s of x_i, we increase the corresponding counts $a_s(x_i)$ by one and update the values of $P_e(a_s)$, using a constant number of operations. Therefore, the additional complexity of steps (ii) and (iii) is again $O(nD)$. Finally, to compute the weighted probabilities in step (iv), at each node of T_{MAX} (which are no more than $nD+1$ in total), we perform $O(m)$ operations, so that the overall computational complexity is $O(nD) + O((nD+1)m) = O(nmD)$ operations.

Following the same reasoning, it is easy to see that the complexity of the BCT algorithm is also $O(nmD)$. A similar argument shows that, as a function of n and D, the complexity of the k-BCT algorithm is also $O(nD)$. However, in order to determine the exact complexity of k-BCT as a function of all the parameters n, m, D and k, we actually need its most-efficient implementation, as described in Appendix A. As outlined in more details there, by using a priority queue to maintain all possible candidates and efficiently pick the top-k combinations for the lists of the maximal probabilities $P_{m,s}$, the complexity of the k-BCT algorithm in its most-efficient implementation can be reduced to $O(nmD \times km\log(km))$.

From the above discussion and the description of the algorithms it is also easy to see that the memory requirements of the CTW and BCT algorithms are of order $O(nmD)$, while for the k-BCT algorithm they are of order $O(nmDk)$.

2.3.6 Bibliographical remarks

The CTW algorithm was first introduced for data compression in 1993. It was originally described for binary observations ($m = 2$) and only in the special case of $\beta = 1/2$ by Willems et al. (1993a). The general version of CTW for chains on non-binary alphabets and arbitrary β was introduced by Tjalkens et al. (1994), without reference to Bayesian inference. The connection between CTW and the prior predictive likelihood $P_D^*(x)$ established in Theorem 2.1 was given in the restricted setting $m = 2, \beta = 1/2$ in the unpublished work of Willems et al. (1993c), and the outline of a corresponding argument in the case of general β (still only for binary time series) was later described in Willems et al. (2002). Another approach investigating the Bayesian interpretation of CTW in the setting of data compression was studied in Matsushima and Hirasawa (1994, 2009). The result of Theorem 2.1 at the level of generality stated here is new, as is the class of prior distributions $\pi_D(T;\beta)$.

A special case of the BCT algorithm, termed the Context Tree Maximizing (CTM) algorithm, was first introduced, again in the context of data compression, by Willems and Volf (1994, 1995); Willems et al. (1993c), for binary observations ($m = 2$) and only in the special case of $\beta = 1/2$. An extension for arbitrary β (still only for $m = 2$) was later given in Willems et al. (2002); the general version of the BCT algorithm presented here is new. The fact that the BCT algorithm identifies the MAP tree model was established in the restricted setting $m = 2$, $\beta = 1/2$ in Willems et al. (1993c, 2000), and some generalisations (still restricted to $m = 2$) are discussed in Willems et al. (2002). Theorem 2.2 at the level of generality stated here is new.

The k-BCT algorithm, Theorem 2.3, and the MCMC samplers of Section 2.3.4 are new. All the methodological contributions, algorithms and theoretical results presented in the rest of this thesis are also new, broadening the scope of applications of context-tree models by a lot, in a number of different practical settings.

Further connections and comments. Another point of view which naturally relates to the present development is Rissanen's celebrated Minimum Description Length (MDL) principle (Grünwald, 2007; Rissanen, 1987, 1989). The MDL principle provides a broad operational foundation for statistical inference, as well as constructive tools and appealing metaphors for selecting prior distributions (Chipman et al., 2001). In particular, MDL considerations underpin much of the original work on CTW and our own choice of priors. A method commonly used for model comparison is the Bayesian Information Criterion (BIC), derived by Schwarz (1978) as an asymptotic approximation to twice the logarithm of the Bayes factor between two models (Kass and Raftery, 1995). The form of the BIC and its familiar "$(1/2)\log n$-per-degree-of-freedom" log-likelihood penalty also shares deep connections with the MDL principle (Barron et al., 1998; Csiszár and Shields, 2000).

2.4 Model selection

Here, we compare the model selection performance of the BCT algorithms of Section 2.3 with the VLMC and MTD approaches described in the Introduction. In the remaining of this section we give brief descriptions of how these three different techniques will be used, and then present extensive experimental results with simulated and real data.

Bayesian Context Trees. Variable-memory models describe a flexible and rich class of higher-order Markov chains that admit parsimonious parametrisations and allow for natural graphical representations of important structural dependencies. In particular, the shape of the context tree can be easily interpreted and provides useful information about the regularities present in the data (Bejerano and Yona, 2001; Mächler and Bühlmann, 2004). However, since BCTs are a vast model class, global model selection techniques based, e.g., on criteria like AIC and BIC, cannot be applied directly via, say, exhaustive search.

This is why it is so important that the BCT framework provides a consistent foundation for learning and evaluating appropriate models for a given data set x. The BCT and the k-BCT algorithms can be used to identify the top-k *a posteriori* most likely models T_i^*, $i = 1, 2, \ldots, k$, for some reasonable k, and we can further explore the model posterior $\pi(T|x)$ using the MCMC samplers of Section 2.3.4. With the interpretation of $\pi(T|x)$ as a measure of the "truth" of a model T (Chipman et al., 2001), the posterior probability provides a quantitative confidence measure for the resulting models.

Variable-length Markov chains. The VLMC class (Bühlmann, 2000; Bühlmann and Wyner, 1999) consists of tree models very similar to those in the BCT class $\mathcal{T}(D)$, except for the fact that VLMC trees are not necessarily proper. The associated VLMC model selection algorithm also has similarities with the pruning procedure of the BCT algorithm. First, a version of the tree T_{MAX} is constructed and the count vectors a_s are computed similarly with BCT. And then, T_{MAX} is pruned based on a *cut-off parameter* K, which plays a role analogous to β, finally producing the fitted model. Theoretical justifications for the resulting VLMC model are provided in Bühlmann and Wyner (1999), where general conditions for asymptotic consistency are established.

The VLMC results in our experiments below are obtained using the implementation in the `R` package `VLMC` (Mächler, 2022). Since the algorithm that uses the default value of the cut-off parameter K ("default-VLMC") generally gives significantly inferior results, we also examine the results obtained by optimizing the choice of K in order to minimise the BIC or the AIC score ("best-BIC-VLMC" and "best-AIC-VLMC"). But these parameter optimisations are computationally very costly, as we point out in more detail in the discussion at the end of Section 2.4.1.

Mixture transition distribution models. The original 'single-matrix' version of the MTD model (Raftery, 1985) is based on a different way of parsimoniously representing the Dth order transition probabilities of a Markov chain, as a mixture of the form,

$$P(x_0|x_{-D}^{-1}) = \sum_{d=1}^{D} \lambda_d Q(x_0|x_{-d}),$$

where $(\lambda_1, \lambda_2, \ldots, \lambda_D)$ are lag parameters and $Q = (Q(j|i))$ is a stochastic matrix. Numerical methods for fitting this model via approximate maximum likelihood were developed by Raftery and Tavaré (1994), and a generalisation, the *multi-matrix MTD model*, or MTDg, was introduced by Berchtold (1996, 1998). The MTDg model is based on the representation,

$$P(x_0|x_{-D}^{-1}) = \sum_{d=1}^{D} \lambda_d Q^{(d)}(x_0|x_{-d}),$$

where a different stochastic matrix $Q^{(d)} = (Q^{(d)}(j|i))$ is used for each lag $d = 1, 2, \ldots, D$.

In our experiments below we use the MTD and MTDg implementation in the `R` package `march` (Maitre et al., 2022). For each data set we run the MTD and the MTDg algorithms for a range of possible depths D, and choose the value of D that minimises the corresponding BIC or AIC score. We refer to the resulting models as the best-BIC and best-AIC models. Recently, based on the sparsity-inducing prior of Heiner et al. (2019), a Bayesian approach was also developed for model selection and estimation with MTD models (Heiner and Kottas, 2022b), while stationary MTD models have also been studied in Zheng et al. (2022).

Alternative approaches. Closing, we note that there exist a number of alternative approaches to modelling discrete time series apart from those described above and in the Introduction. An important collection of tools is provided by Hidden Markov Models (HMMs), which are a general and very broadly used model class, with a wide range of applications and a variety of associated methodological procedures for learning and inference (Bishop, 2006; Murphy, 2012; Zucchini and MacDonald, 2017). Another interesting class of models is that of *reversible* variable-memory chains (Bacallado, 2011; Bacallado et al., 2009, 2013, 2016).

A more classical approach to discrete time series modelling is via discrete analogs of linear models, often using multinomial logit or probit regression (Yee, 2010). Such a linear-predictor approach is described in Zeger and Liang (1986), and a different parsimonious class of models with an emphasis on binary time series is given in Fahrmeir and Kaufmann (1987). A treatment of partial likelihood inference on generalised discrete linear models is presented in Fokianos and Kedem (2003), and an extension of the traditional ARMA methodology to integer autoregressive models for count time series is developed in Fokianos (2012). For a detailed overview of some of the current and most recent activity in the field of discrete-valued time series, see also Davis et al. (2016).

Model comparison. A natural and logically consistent way to compare different models from $\mathcal{T}(D)$ is to compare their posterior probabilities $\pi(T|x)$, but this is not possible for models produced by different methods. In such cases, we follow the common practice of comparing their corresponding Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) *scores* (Akaike, 1973; Schwarz, 1978). BIC is asymptotically consistent when dealing with a fixed number of finite-dimensional parametric models (Hannan and Quinn, 1979; Wei, 1992) whereas AIC is typically asymptotically efficient and minimax optimal in infinite-dimensional, nonparametric settings (Barron et al., 1999; Shibata, 1980); see also Dziak et al. (2019). This dichotomy suggests that, for our purposes, BIC is the more relevant criterion, as confirmed by our findings in Section 2.4.1 and Appendix C.1.

2.4.1 Simulated data

Here we compare the performance of the BCT, VLMC, and MTD approaches described above on a relatively short synthetic time series. Analogous comparisons on simulated data from two more chains – one from the VLMC paper (Bühlmann, 2000) and one from the MTD paper (Berchtold and Raftery, 2002) – are carried out in Appendix C.1.

A ternary 5th order Markov chain. We revisit the 5th order variable-memory chain $\{X_n\}$ on the alphabet $A = \{0,1,2\}$ of $m = 3$ letters, with model given by the tree T shown in Figure 2.1 of Section 2.1. The associated parameters at its leaves are given by,

$$\theta_1 = (0.4,0.4,0.2),\ \theta_2 = (0.2,0.4,0.4),\ \theta_{00} = (0.4,0.2,0.4),\ \theta_{01} = (0.3,0.6,0.1),$$
$$\theta_{022} = (0.5,0.3,0.2),\ \theta_{0201} = (0.8,0.05,0.15),\ \theta_{0202} = (0.1,0.2,0.7),$$
$$\theta_{0210} = (0.35,0.55,0.1),\ \theta_{0211} = (0.05,0.25,0.7),\ \theta_{0212} = (0.1,0.3,0.6),$$
$$\theta_{02000} = (0.3,0.45,0.25),\ \theta_{02001} = (0.1,0.1,0.8),\ \theta_{02002} = (0.7,0.2,0.1).$$

With a time series x consisting of $n = 10,000$ simulated observations, the MAP tree model T_1^* obtained by the BCT algorithm with $D = 10$ and $\beta = 1 - 2^{-m+1} = 3/4$ is the true underlying tree model. Its posterior probability is $\pi(T_1^*|x) \approx 0.3946$ while its prior probability is $\pi(T_1^*) \approx 5.8 \times 10^{-6}$. Although it may seem quite unremarkable that the "correct" model is identified based on a series of 10,000 samples, it is worth noting that with $D = 10$ there are more than 10^{5900} different models in $\mathcal{T}(D)$. With $n = 1,000$ samples, $D = 10$, and $\beta = 3/4$, the top-5 *a posteriori* most likely models produced by the k-BCT algorithm are shown below in Figure 2.2. The MAP model is a depth-4 subtree of the true underlying model, with prior $\pi(T_1^*) \approx 2.9 \times 10^{-4}$ and posterior $\pi(T_1^*|x) \approx 0.2702$. The true model appears as the fourth most likely tree, T_4^*, with posterior $\pi(T_4^*|x) \approx 0.0213$. The sum of the posterior probabilities of the top-5 models is approximately 0.4737.

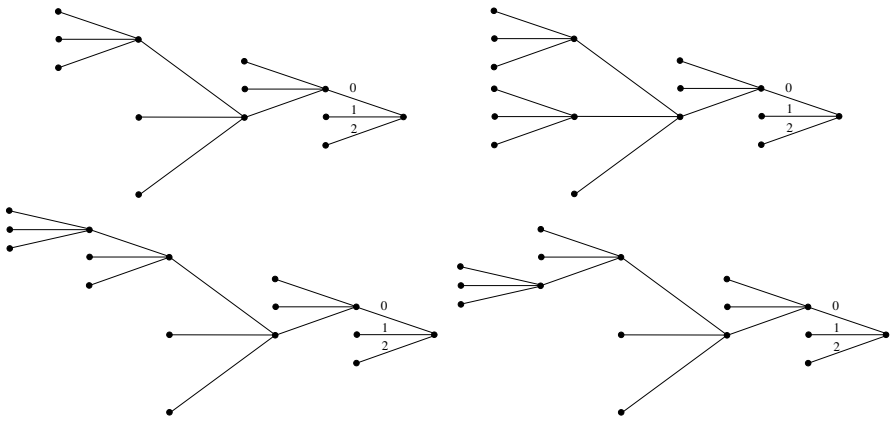


Figure 2.2: The first, second, third, and fifth MAP trees models $T_1^*, T_2^*, T_3^*, T_5^*$ produced by the k-BCT algorithm with $n = 1,000$ samples; the fourth tree T_4^* is the true tree of Figure 2.1. The posterior odds are $\pi(T_1^*|x)/\pi(T_i^*|x) = 2.369, 5.358, 12.69, 15.24$, respectively, for $i = 2,3,4,5$.

The default-VLMC model is the first tree shown in Figure 2.3; only about half of its nodes appear in the true underlying model. It has a worse BIC score and a better AIC score than our MAP model. The best-BIC-VLMC produces the small tree of depth 3 shown second in Figure 2.3, which is a subtree of the true model; it has a good BIC score and a poor AIC score. In sharp contrast, the best-AIC-VLMC produces a clearly overfitted model of depth 6, shown third in Figure 2.3, which has a very a poor BIC score, but a good AIC score. Finally, the best-BIC-MTD and the best-BIC-MTDg both give $D = 0$, whereas the the best-AIC-MTD gives $D = 3$ and the best-AIC-MTDg gives $D = 2$. Their scores (both BIC and AIC) are generally quite worse than those of the BCT and VLMC models.

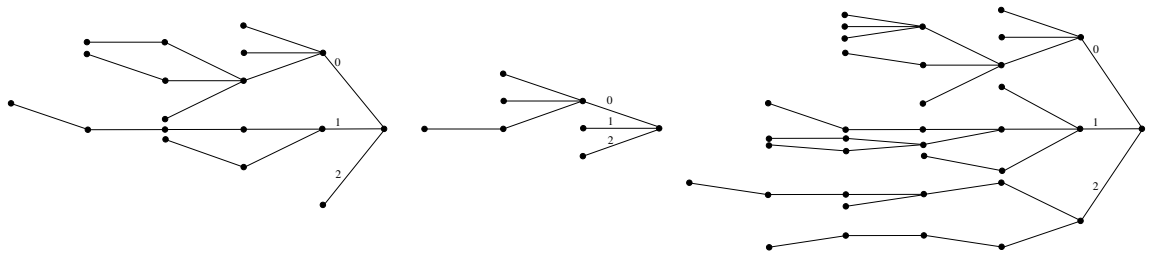


Figure 2.3: The models produced by the default-VLMC, the best-BIC-VLMC and the best-AIC-VLMC methods, with $n = 1,000$ samples.

Overall, the BCT and the best-BIC-VLMC algorithms achieve the best performance in terms of scores, and they learn part of the true underlying tree. VLMC has a marginally better BIC score whereas BCT has a slightly better AIC score (by approximately 1% in both cases). More importantly, our MAP model T_1^* has an additional full branch at depth 4 that reveals more of the true underlying structure, and k-BCT also identifies the true model as T_4^*.

Discussion. In the results on simulated data above and in the two examples in Appendix C.1, the BCT and k-BCT algorithms consistently give the most accurate model fit, with the best-BIC version of VLMC often giving similar results. The BCT framework has the advantage of identifying not just a single model, but the top-k *a posteriori* most likely models, together with their exact prior and posterior probabilities. And, importantly, in terms of complexity, the BCT algorithm is more efficient than either VLMC or MTD, typically by at least two orders of magnitude, since it does not require any tuning.

VLMC. The default and best-AIC versions of VLMC generally gave similar or identical models, usually much larger than the true underlying model, in rather typical examples of overfitting. The best-BIC-VLMC was found to be much more accurate in revealing significant parts of the true model, as expected in view of the earlier AIC-vs-BIC discussion. The resulting models were smaller, which is consistent with the observation that optimizing the BIC score led to larger values for the cut-off parameter K and more frequent pruning. For the best-AIC and best-BIC versions, we executed VLMC approximately 500 times with different values of $K \in [1, 10]$. Although a smaller range ($2.8 \leq K \leq 6$) was used by Mächler and Bühlmann (2004), we found it often necessary to look further; e.g., for the financial data in Appendix C.1 the best value was found to be close to 10, and for the genetic data in Section 2.4.2 it was larger than 20. In most cases, using fewer runs resulted in different models giving poor fits.

BCT. The MAP models produced by the BCT algorithm were usually similar to those produced by the best-BIC-VLMC, they had good AIC and BIC scores, and they generally learned the most accurate approximations of the underlying model among all methods considered. The additional models obtained by k-BCT offered further indications of the type of structure present in the data, and they were accompanied by posterior probabilities, indicating the level of "posterior confidence" we may have in these models. Also, BCT and k-BCT only require a single run with the default value of the hyperparameter β.

MTD. None of the MTD methods performed as well as the BCT or VLMC algorithms, partly because they gave (by design) less flexible models. As their only output in terms of the model is the Markov order D and the number of nonzero lag parameters λ_d, which can only take a few discrete values, the best-BIC and best-AIC results were often same. MTDg in most cases had worse scores than MTD, but even the MTD's overall performance was not competitive with that of BCT and VLMC. Finally, MTD had very high complexity, since the implicit maximum likelihood computation is over a non-convex space defined by a large number of non-linear constraints (Raftery and Tavaré, 1994). For this reason, it appears infeasible in practice to use values for D significantly larger that $D = 10$.

2.4.2 Real data

In view of the above discussion, for the comparisons in the real-world data examples we only consider the best-BIC versions of VLMC, MTD and MTDg. One more example with a financial time series is given in Appendix C.1.

SARS-CoV-2 genome. The SARS-CoV-2 virus – standing for Severe Acute Respiratory Syndrome Coronavirus 2 – is the virus responsible for the Covid-19 global pandemic. Here we examine its genome, available in the GenBank database (Clark et al., 2016) as the sequence MN908947.3 identified in Wu et al. (2020). It consists of $n = 29,903$ base pairs, and we translate the four-letter DNA alphabet to $\{0,1,2,3\}$ via the map (A,C,G,T) $\mapsto (0,1,2,3)$.

The top 3 models obtained by the k-BCT algorithm with $\beta = 1-2^{-m+1} = 7/8$, $D = 10$, and $k = 3$ are shown as the first three trees in Figure 2.4. The MAP model has posterior probability $\pi(T_1^*|x) \approx 0.963$ and prior $\pi(T_1^*) \approx 4.3 \times 10^{-5}$. The sum of the posterior probabilities of these three models is ≈ 0.9994. The VLMC model is the depth-2 subtree of T_1^* shown last in Figure 2.4. The optimisation of the cut-off parameter K required to find the best-BIC model, resulted in the choice $K = 22.1$. Both MTD and MTDg give $D = 1$ as the optimal depth, corresponding to a simple first order Markov chain.

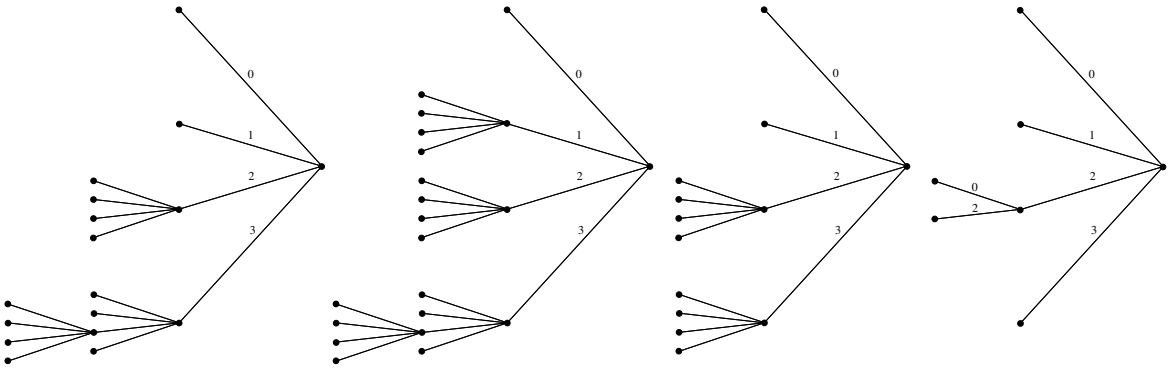


Figure 2.4: First three trees: The top-3 MAP tree models T_1^*, T_2^*, T_3^* obtained by the k-BCT algorithm on the SARS-CoV-2 genome. The posterior odds $\pi(T_1^*|x)/\pi(T_2^*|x)$ and $\pi(T_1^*|x)/\pi(T_3^*|x)$ are 35.75 and 101.4, respectively. Last tree: The best-BIC-VLMC model; its AIC and BIC scores are both within 0.1% of the corresponding scores of the MAP tree model T_1^*.

The AIC and BIC scores of all models generated by the three approaches are within 0.3% of each other. It is interesting that our MAP model has such high posterior probability, and that k-BCT gives models of depth 3 with very high confidence. Although it is not possible to otherwise verify the significance of the bigger depth of the BCT models compared to those of the other methods, it may be that they find evidence of the fact that DNA naturally gets encoded into triplets of bases to form codons that specify particular amino acids.

Pewee birdsong. The twilight song of the wood pewee bird can be described as a sequence consisting of an arrangement of musical phrases taken from an alphabet of three specific, distinct phrases. The data here consist of a single contiguous song by a wood pewee, of length $n = 1327$ phrases, first recorded and studeid extensively by Craig (1943). It was analysed by Berchtold (2001); Raftery and Tavaré (1994) using MTD models, by Kharin and Piatlitski (2011) using a different MTD-type representation, and by Sarkar and Dunson (2016) using CTF models; it is also contained in the `R` package `march`. It is well known that the birdsong contains significant variability but it is also fairly structured, with specific repeating patterns. The most common of these, described by Saunders (1944) as "the commonest and most pleasing sentence," is the string $s^* = 0201$, which dominates the data set, appearing 266 times and occupying 1064 positions or a little over 80% of the data.

The MAP tree T_1^* obtained by BCT with $m = 3$, $\beta = 1 - 2^{-m+1} = 3/4$, and $D = 10$, is shown in Figure 2.5. Its prior probability is $\pi(T_1^*) \approx 4.1 \times 10^{-5}$, and its posterior is $\pi(T_1^*|x) \approx 0.1244$. Clearly the importance of $s^* = 0201$ is captured in the MAP tree T_1^*, which includes the entire prefix 020 necessary to predict the appearance of $s^* = 0201$.

The VLMC produces an interesting model, also shown in Figure 2.5, which is quite similar to T_1^*. Although it does not include the 020 context, it does include the context 02, which contains most of the predictive power regarding $s^* = 0201$: While s^* appears 266 times in the data, the context $s = 201$ appears 267 times, therefore, we can "statistically" almost identify s^* with s. Compared to the VLMC model, the MAP tree T_1^* has a marginally better AIC score and a slightly worse BIC score, but it is clear that the two methods learn much of the same structure from the data. The MTD gives $D = 6$ as the optimal depth, while MTDg gives $D = 3$. The resulting MTDg model has better AIC and BIC scores than the MTD model, but they are still much worse than those of our MAP model. This confirms the findings of both Raftery and Tavaré (1994) and Berchtold (2001), where it was noted that the MTD family of models is not appropriate for this data set, in large part due to the high significance of individual patterns like s^*.

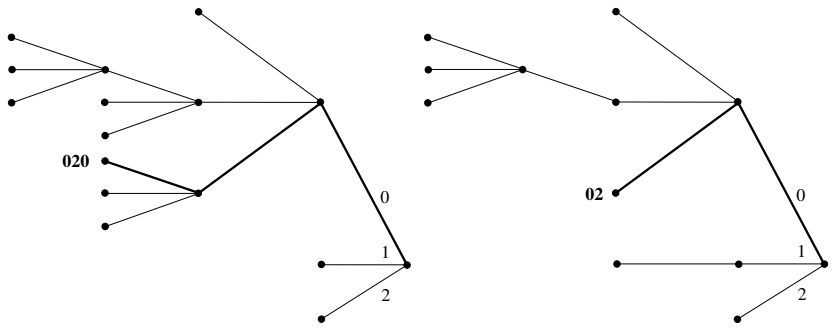


Figure 2.5: The MAP tree model (left) and VLMC model (right) for the pewee birdsong data.

The four *a posteriori* most likely tree models after T_1^*, as identified using by k-BCT algorithm with $k = 5$, are shown in Figure 2.6. They are all quite similar to the MAP tree, each one differing from T_1^* in only a single branch.

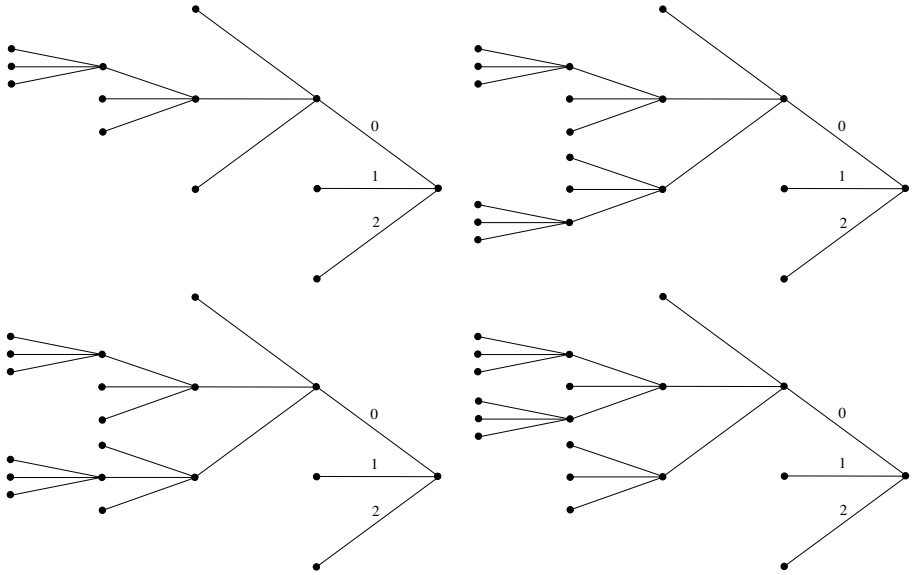


Figure 2.6: The four *a posteriori* most likely models T_i^*, $i = 2,3,4,5$, after T_1^*, obtained by the k-BCT algorithm from the pewee birdsong data. The corresponding posterior odds are $\pi(T_1^*|x)/\pi(T_2^*|x) \approx 5.727$ and $\pi(T_1^*|x)/\pi(T_i^*|x) \approx 7.111$, for $i = 3,4,5$.

The sum of the posterior probabilities of the top-5 models here is 20%. Therefore, in order to get a better sense of the posterior distribution on model space, we employed the RW sampler to produce $N = 10^6$ samples from $\pi(T|x)$. The acceptance rate was 57.8%, a total of 274,721 unique trees were visited, and the sum of their posterior probabilities was 61.2%. The MCMC frequency of T_1^* in Figure 2.7 indicates that the sampler converged quite quickly. The 100 most visited models all have depths between 4 and 7, and their total posterior probability is 38.3%. Together with the results of k-BCT and VLMC, these suggest that there is at least fourth-order structure present in the data.

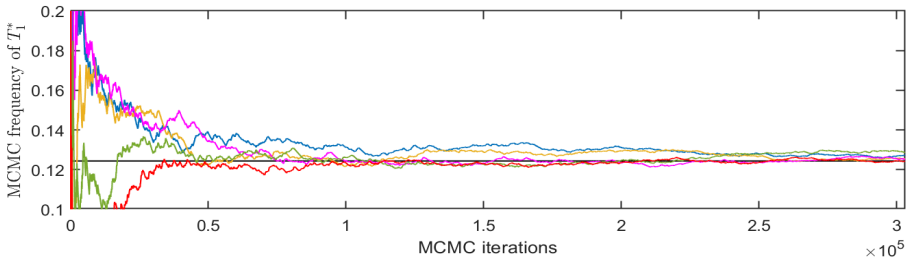


Figure 2.7: MCMC frequency of T_1^*. The five graphs correspond to five independent repetitions of the experiment with $N = 3 \times 10^5$ iterations; the horizontal line is the limiting frequency, $\pi(T_1^*|x)$.

Neural spike trains. Next we consider a long binary time series describing the spike train recorded from a single neuron in the V4 region of a monkey's brain; it consists of $n = 3,919,361$ bits. The recording was made during the experiment described in Gregoriou et al. (2009, 2012), while the monkey was performing an attention task. The recorded signal was discretised into one-millisecond bins (with $x_i = 1$ if there was a spike in bin i and $x_i = 0$ otherwise), corresponding to a trial lasting a little over 65 minutes. As we have not been able to find implementations of VLMC or MTD that can operate on data sets of this length, in this section we only present the results obtained by the BCT and k-BCT algorithms.

With $D = 100$, and $\beta = 1 - 2^{-m+1} = 1/2$, the MAP tree model T_1^* is shown in Figure 2.8. It has depth 98 and, with the exception of two additional branches at depths 9 and 90, it is very similar to a renewal model, qualitatively similar to the first chain in Appendix C.1. Its prior probability is $\pi(T_1^*) \approx 3.1 \times 10^{-61}$ and its posterior $\pi(T_1^*|x) \approx 2.1 \times 10^{-8}$. Although this probability is small, we note that there are more than $79^{10^{29}}$ possible models of depth no more than 100, cf. (2.8), and that this posterior is still larger than the corresponding prior probability by more than 50 orders of magnitude. Therefore, the observations still offer significant support for this model.

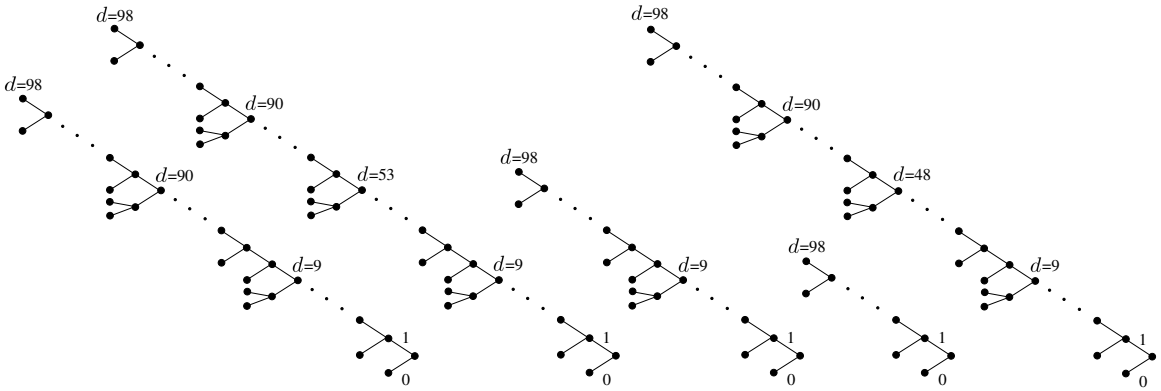


Figure 2.8: The top $k = 5$ models obtained by k-BCT on the spike train data. The posterior odds $\pi(T_1^*|x)/\pi(T_i^*|x)$ for $i = 2,3,4,5$, are 1.2, 1.43, 1.52 and 1.53, respectively.

All the next four *a posteriori* most likely models shown in Figure 2.8 are also very similar to simple renewal models – and in fact the fourth model T_4^* is exactly the renewal tree of depth 98. These observations offer a partial justification to the biological intuition behind the elementary Poisson/renewal models commonly used in neuroscience (Dayan and Abbott, 2001; Rieke et al., 1999), and confirm relevant earlier findings (Gao et al., 2006, 2008). In other words, we have learned from the data that the statistically most significant factor in determining whether a neuron will fire, given its past history, is the time when it most recently fired before.

Although the sum of the posterior probabilities of the top five models is less than 10^{-7}, in this case it would not make sense to employ an MCMC sampler to explore the posterior further. The reason is that, given the very small values of the posterior probabilities of the top-5 models, even with 10^6 MCMC samples visiting 10^6 distinct models, we would still only visit around 1% of the support of the posterior, at best. On the other hand, increasing the value of k can give a better idea of the shape of the posterior near its mode T_1^*. With $k = 50$, k-BCT produced 50 trees, all of depth 98, and all of them being small variations of the renewal model T_4^*: All the resulting models T_i^* had between one and five additional branches at various depths. The sum of their posterior probabilities is 4.1×10^{-7}.

As a final test of the scalability of the BCT algorithm on a large data set, we obtained the MAP tree with maximum depth $D = 1500$. Interestingly, it is the same as the MAP tree for $D = 100$, and with only a slightly smaller posterior probability $\pi(T_1^*|x) \approx 1.6 \times 10^{-8}$.

2.5 Estimation tasks

In this section we present two examples that illustrate the utility of the MCMC samplers of Section 2.3.4 for posterior exploration, and the application of the BCT methodology to Markov order estimation and parameter estimation.

Example 2.1 (Daily changes in S&P 500). We consider the daily changes in Standard & Poor's index, from January 2, 1928 until October 7, 2016 (available at https://finance.yahoo.com/quote/^GSPC/), quantised to $m = 7$ values: If the change between two successive trading days, day $(i-1)$ and day i, is smaller than -3%, x_i is set equal to 0; for changes in the intervals $(-3\%, -2\%]$, $(-2\%, -1\%]$, $(-1\%, +1\%]$, $(1\%, 2\%]$, and $(2\%, 3\%]$, x_i is set equal to $1, 2, 3, 4$ and 5, respectively; and for changes greater than 3%, $x_i = 6$.

Based on the resulting $n = 22,900$ points x_i, the top $k = 5$ *a posteriori* most likely models obtained by the k-BCT algorithm with maximum tree depth $D = 260$ (corresponding to approximately one calendar year's trading days), are described in Figure 2.9.

Interpretation. The shape of the MAP model T_1^* contains significant information. Since its maximal depth is 5, in order to determine the distribution of the next sample we never have to look more than five days back – corresponding to a week of trading days. The smaller the changes in the most recent S&P values, the further back we need to look in order to predict tomorrow's value. For example, if the difference between yesterday and today is larger than $\pm 2\%$, we need look no further than yesterday; if it is between 1% and 2%, we need to look at the day before yesterday as well; and if it is smaller than $\pm 1\%$, we need to look even further back, but no more than a week.

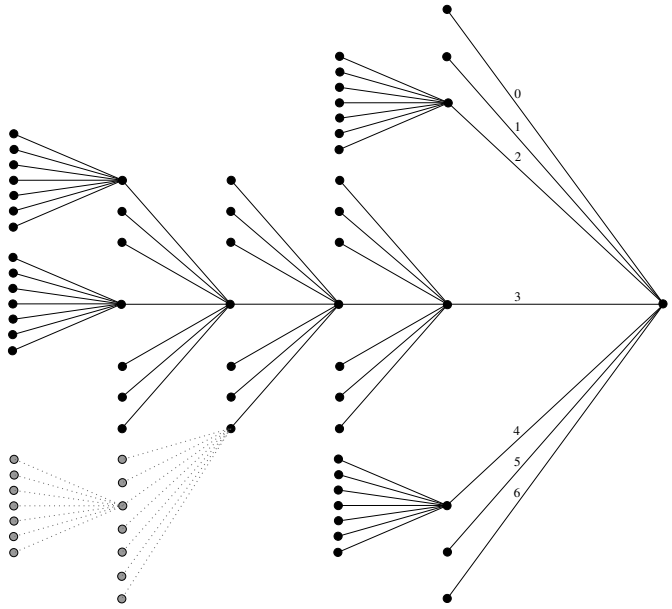


Figure 2.9: The tree shown *without* the two dotted branches is the MAP model T_1^* obtained from $n = 22,900$ observations, with $D = 260$. Its posterior probability is $\pi(T_1^*|x) \approx 0.0174$ and its prior $\pi(T_1^*) \approx 5.7 \times 10^{-11}$. The whole tree shown is the fifth *a posteriori* most likely model T_5^*, and T_2^*, T_3^* and T_4^* were found to be small variations around T_1^* and T_5^*, all with depth 5. The corresponding posterior odds $\pi(T_1^*|x)/\pi(T_i^*|x)$, for $i = 2,3,4,5$, are 1.094, 1.367, 1.496 and 2.467, respectively.

The sum of the posterior probabilities of the top-5 MAP models is less than 6.5%. But with $n = 22,900$ data points, $m = 7$ and $D = 260$, the complexity of k-BCT becomes prohibitive for large values of k. In order to explore $\pi(T|x)$ further, we ran the RW sampler with $T^{(0)} = T_1^*$ for $N = 10^6$ MCMC iterations.

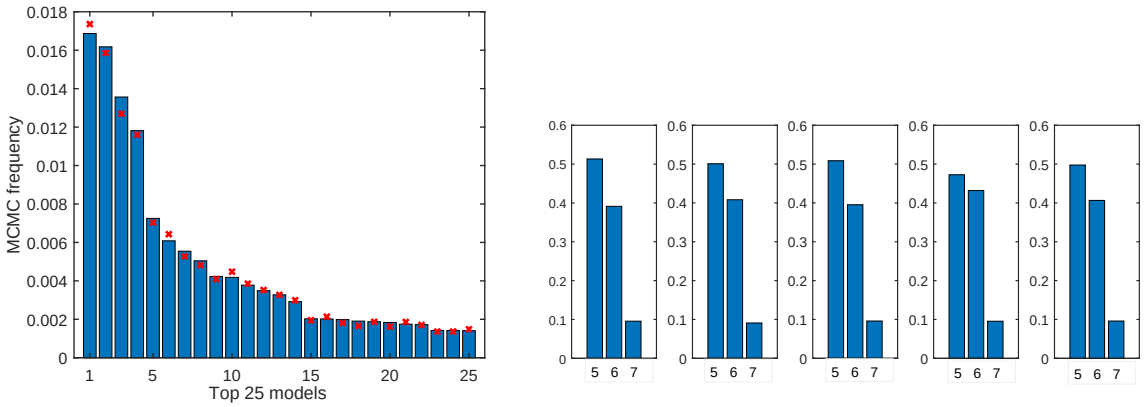


Figure 2.10: Left: MCMC empirical frequencies of the 25 most frequently visited models, after $N = 10^6$ iterations. The corresponding true posterior probabilities are marked with a red 'x'. Right: Markov order estimation. The five histograms show the empirical frequencies of the depths of the 1,000 most visited models, in five independent repetitions of the same experiment.

Markov order estimation. The acceptance rate in the MCMC sample was 44.9%, and a total of 356,531 different models were visited. The MCMC frequencies of the 25 most visited models shown in Figure 2.10 indicate that the sampler has converged after $N = 10^6$ iterations. The MCMC output can also be used for Markov order estimation, by providing an approximation to the posterior distribution on model depth. The empirical distributions of the model depths obtained in five repetitions of the same experiment are shown in Figure 2.10.

Example 2.2 (A bimodal posterior). Consider a 3rd order chain $\{X_n\}$ on the alphabet $A = \{0,1,2,3,4,5\}$ with the property that each X_n depends on the past $(X_{n-1}, X_{n-2}, X_{n-3})$ only via X_{n-3}. Specifically, suppose that, for $i, j, a, b \in A$,

$$\Pr(X_n = j | X_{n-1} = a, X_{n-2} = b, X_{n-3} = i) = Q_{ij},$$

where the transition matrix $Q = (Q_{ij})$ is given by,

$$\begin{pmatrix} 0.5 & 0.2 & 0.1 & 0 & 0.05 & 0.15 \\ 0.4 & 0 & 0.4 & 0.2 & 0 & 0 \\ 0.3 & 0.1 & 0.23 & 0.12 & 0.05 & 0.2 \\ 0.05 & 0.1 & 0.05 & 0.05 & 0.03 & 0.72 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0.1 & 0.2 & 0.3 & 0.2 & 0.05 & 0.15 \end{pmatrix}$$

The model of $\{X_n\}$ viewed as a variable-memory chain is clearly the complete tree of depth 3, but the dependence of each X_n on its past is only meaningful if it can extend at least three time steps back: The first and second most recent symbols are independent of X_n.

Therefore, it is not surprising that the MAP model identified by the k-BCT algorithm based on $n = 1850$ samples is simply the root, $T_1^* = \Lambda = \{\lambda\}$, with T_2^* a complex tree of depth 3 (with 54 leaves at depth 3) being a close second; their posterior probabilities are $\pi(T_1^*|x) \approx 0.3512$ and $\pi(T_1^*|x) \approx 0.2373$, respectively. The next three *a posteriori* most likely models T_3^*, T_4^*, T_5^* were found to be small variations of T_2^*, also of depth 3.

Running the RW sampler with $T^{(0)} = T_1^*$, we found that it never visited any of the other top-5 models after 10^6 iterations, and similarly starting at T_2^* it never visited T_1^*. Although T_2^* can theoretically be reached from T_1^* in just 12 MCMC steps, most models between them have extremely small posterior probabilities; e.g., the complete tree of depth one, $T_c(1)$, which must necessarily be visited in order to move between T_1^* and T_2^*, has posterior $\pi(T_c(1)|x) \approx 3.2 \times 10^{-19}$.

In contrast, the jump sampler with jump parameter $p = 1/2$, made frequent jumps between T_1^* and $\{T_2^*, T_3^*, T_4^*, T_5^*\}$. Starting with $T^{(0)} = T_1^*$, after $N = 10^5$ MCMC iterations

it appears to have explored the bulk of the posterior distribution, having visited all of the significant parts of its support. The empirical frequencies of the top-5 models were very close to their actual posterior probabilities (Figure 2.11), the total number of unique models visited was 39, and the sum of their posterior probabilities was 99.8%.

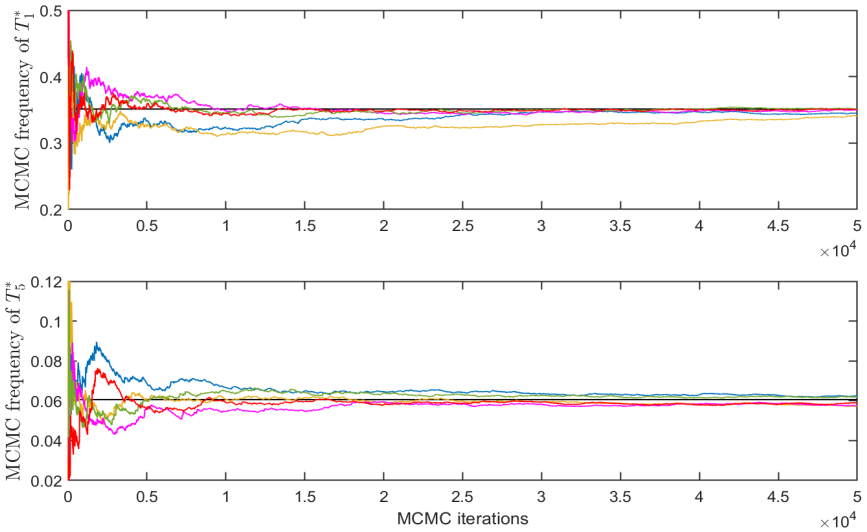


Figure 2.11: Results of the jump sampler. The top plot shows the empirical frequency of T_1^* as a function of the number of MCMC iterations. The five graphs plotted correspond to five independent repetitions of the experiment with 50,000 iterations; the horizontal line is the limiting frequency, $\pi(T_1^*|x)$. The bottom plot shows corresponding results for T_5^*.

Markov order estimation. In terms of Markov order estimation, although the MAP tree T_1^* has depth 0, the mode of the posterior distribution of the tree-depth is 3: The $N = 10^5$ MCMC samples obtained above consist entirely of models having depths either 0 or 3, with corresponding MCMC frequencies of approximately 35.13% and 64.87%, respectively.

Estimation of general functionals. More generally, for any statistic $F(\theta, T)$, we can obtain MCMC samples $\{(\theta^{(t)}, T^{(t)})\}$ using the joint sampler of Section 2.3.4, and use the values $\{F(\theta^{(t)}, T^{(t)})\}$ to provide an approximation to the posterior $\pi(F|x)$. Standard Bayesian methods (Bernardo and Smith, 1994; Geisser, 1993; Gelman et al., 2013) can then be applied to provide point estimates, credible sets, and other relevant information.

Parameter estimation. For example, suppose we wish to estimate the parameter $\theta_s(j) = \Pr(X_0 = j|X_{-D}^{-1} = s)$, for the specific context $s = 020$ and $j = 5$. The maximum likelihood estimate (MLE) is $\hat{\theta}_s^{\mathrm{MLE}}(j) = a_s(j)/M_s$, and a commonly used Bayesian counterpart to the MLE, $\hat{\theta}_s^{\mathrm{MAP}}(j)$, is the mode of the full conditional density $\pi(\theta_s(j)|x, T)$ in (2.16) of $\theta_s(j)$ given the data x and the MAP model $T = T_1^*$.

Another Bayesian alternative to the MLE is the posterior mean, which can be approximated using the MCMC samples $\{(\theta^{(t)}, T^{(t)})\}$, as,

$$\hat{\theta}_s^{\text{MCMC}}(j) := \frac{1}{N}\sum_{t=1}^{N} \theta_s^{(t)}(j),$$

and an estimator with smaller variance can also be obtained via Rao-Blackwellization (Blackwell, 1947; Gelfand and Smith, 1990), using (2.16):

$$\hat{\theta}_s^{\text{RB}}(j) := \frac{1}{N}\sum_{t=1}^{N} E\big(\theta_s(j)\big|x, T^{(t)}\big).$$

In this example, we obtain the values:

$\hat{\theta}_s^{\text{MLE}}(j)$	$\hat{\theta}_s^{\text{MAP}}(j)$	$\hat{\theta}_s^{\text{MCMC}}(j)$	$\hat{\theta}_s^{\text{RB}}(j)$	true $\theta_s(j)$
0.1290	0.1871	0.1512	0.1516	0.15

The estimates $\hat{\theta}_s^{\text{MCMC}}(j)$ and $\hat{\theta}_s^{\text{RB}}(j)$ are based on $N = 10^5$ MCMC samples obtained by the jump version of the joint sampler, with jump parameter $p = 1/2$. From these samples we also get an estimate of the entire posterior distribution of $\theta_{020}(5)$, shown in the histogram of Figure 2.12. Finally, in Figure 2.12 we also show the results of five independent MCMC experiments, indicating that the ergodic averages in the estimators $\hat{\theta}_s^{\text{MCMC}}(j)$ and $\hat{\theta}_s^{\text{RB}}(j)$ converge quickly, and that the variance of $\hat{\theta}_s^{\text{RB}}(j)$ appears to be smaller, as expected.

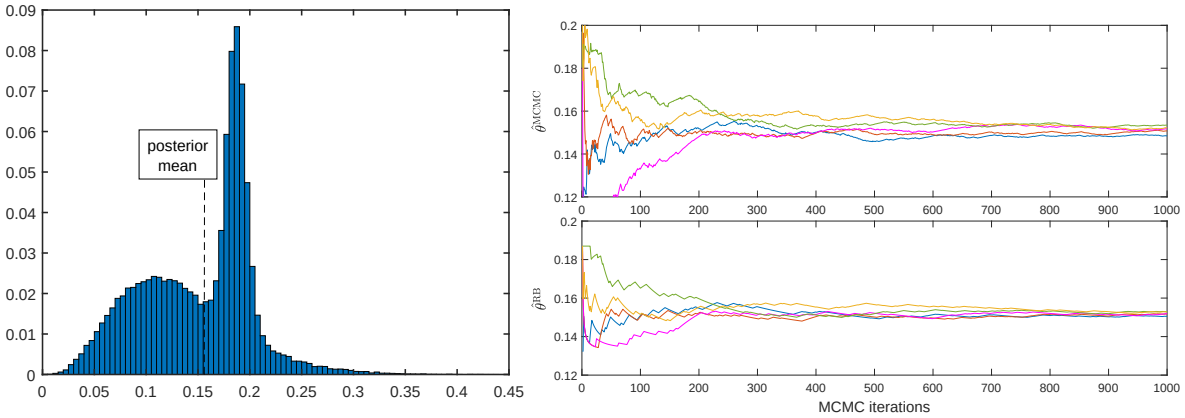


Figure 2.12: Left: Histogram of the $N = 10^5$ MCMC samples $\theta_s^{(t)}(j)$, with $D = 3$, $\beta = 0.95$, $s = 020$ and $j = 5$. Right: Corresponding MCMC estimates. The top plot shows the ergodic averages $\hat{\theta}_s^{\text{MCMC}}(j)$ as a function of the number of MCMC iterations. The five graphs plotted correspond to five independent repetitions of the experiment with 1,000 MCMC iterations each. The bottom plot shows corresponding results for $\hat{\theta}_s^{\text{RB}}(j)$.

2.6 Prediction

Given a *training set* $x_1^t = (x_1, x_2, \ldots, x_t)$ of length t from a discrete time series x, with values in the alphabet $A = \{0, 1, \ldots, m-1\}$, we wish to sequentially predict the next n values in the *test set* $x_{t+1}^{t+n} = (x_{t+1}, x_{t+2}, \ldots, x_{t+n})$. At each step $i = 1, 2, \ldots, n$ in the test set, given the past samples x_1^{t+i-1}, the prediction of the next sample x_{t+i} is expressed as a conditional distribution $\widehat{P}_i(z|x_1^{t+i-1})$, $z \in A$, and the performance of the resulting predictor is evaluated by the normalised, cumulative log-loss,

$$\mathcal{L} = -\frac{1}{n} \sum_{i=1}^{n} \log \widehat{P}_i(x_{t+i}|x_1^{t+i-1}).$$

In the remainder of this section we first describe the natural predictor induced by the BCT framework, together with the alternative prediction methods that will be used below. Then, extensive experimental results are presented, illustrating the efficacy of the BCT predictor in a number of simulated and real-world data sets.

BCT predictor. The ability to compute the prior predictive likelihood sequentially, as described in Section 2.3.5, makes it easy to perform online prediction in a very efficient manner within the BCT framework, something which is very important in practice. The canonical Bayesian rule for predicting the next observation x_{n+1} given the past x_1^n is given by the *posterior predictive distribution* (Gelman et al., 2013),

$$P_D^*(x_{n+1}|x_1^n) = \sum_{T \in \mathcal{T}(D)} \pi(T|x_1^n) \int_\theta P(x_{n+1}|x_1^n, T, \theta) \pi(\theta|T, x_1^n) \, d\theta. \tag{2.17}$$

An important advantage of the Bayesian predictor is that it incorporates model uncertainty and avoids the problem of model selection by replacing it with model averaging. Although it is common that the posterior predictive distribution can only be estimated (e.g., by using MCMC sampling or approximate model averaging), here its value can be computed exactly and sequentially by CTW, via,

$$P_D^*(x_{n+1}|x_1^n) = \frac{P_D^*(x_1^{n+1})}{P_D^*(x_1^n)}. \tag{2.18}$$

Apart from following the well-justified canonical Bayesian methodology, as the following discussion indicates, the BCT predictor can be shown to essentially achieve the optimal minimax regret with respect to the log-loss. Theorem 2.4 gives a nonasymptotic lower bound, for the special case of binary chains and $\beta = 1/2$; it was established by Willems et al. (1993b, 1995) in the setting of data compression.

Theorem 2.4. *For any binary, variable-memory chain model $T \in \mathcal{T}(D)$ and associated parameters $\theta = \{\theta_s\,;\, s \in T\}$, for any data sequence x_1^n of arbitrary length n, and any initial context x_{-D+1}^0, the prior predictive likelihood for $\beta = 1/2$ satisfies,*

$$\log P_D^*(x_1^n | x_{-D+1}^0) \geq \log P(x_1^n | x_{-D+1}^0, \theta, T) - \frac{|T|}{2} \log n + C, \tag{2.19}$$

with an explicit constant $C = C(T)$, independent of n and θ.

The lower bound (2.19) is asymptotically tight up to and including the $\log n$ term, both in expectation and for individual sequences: Corresponding upper bounds follow from the fundamental "converse" theorem by Rissanen (1984, 1986b), and from the general results by Weinberger et al. (1994). These results clearly indicate that the BCT predictor indeed essentially achieves the optimal minimax regret (Takeuchi and Barron, 2014).

VLMC predictor. For prediction using variable-length Markov chains we use the prediction function in the `R` package `VLMC`. This selects a VLMC model based on the training data, and then predicts future samples by the maximum likelihood parameters associated with this model. We only show results for the best-BIC-VLMC model, as it was found to be the most effective choice in practice; see also the relevant comments in Section 2.4.1. When the data are generated by a stationary and ergodic variable-memory chain, the results of Bühlmann and Wyner (1999) imply that the VLMC predictor is asymptotically consistent, but for consistency it is the size of the training data that needs to grow to infinity. Further methodology in connection with VLMC prediction is developed in Bühlmann (2000).

MTD predictor. The MTD approach to prediction is again based on first selecting a model and then performing prediction using an approximation to the maximum likelihood parameters associated with that model. Among the four MTD versions discussed in Section 2.4, we only report results obtained by the best-BIC-MTDg, since the performance of the other three methods was either identical or inferior in practice.

SMC predictor. As described in the Introduction, SMCs are a generalisation of the BCT models in $\mathcal{T}(D)$: Each SMC model consists of a partition of the set of all contexts of depth D into different states, such that all contexts in the same state induce the same conditional distribution on the next symbol of the underlying chain. In Jääskinen et al. (2014), a class of priors for SMCs are defined, and algorithms for selecting approximate MAP versions of models and parameters are developed in the subsequent work by Xiong et al. (2016). In our experiments, prediction is performed based on the resulting model and parameters obtained using the code publicly available at `https://www.helsinki.fi/bsg/filer/SMCD.zip`.

CTF predictor. The CTF models of Sarkar and Dunson (2016), described in the Introduction, use conditional tensor factorisations to represent the full transition probability tensor of a higher-order Markov chain. In our experiments we use the CTF software publicly available at `https://github.com/david-dunson/bnphomc`. Prediction is again performed in two steps. First a model is selected via the results of an MCMC sampler on model space, and appropriate parameters are chosen by an approximation to their posterior mean via Gibbs sampling. The induced predictive distributions are then obtained from the selected model and parameters.

Throughout our experiments, we take the maximal depth to be $D = 10$ for BCT, MTD, SMC and CTF. Following standard practice (Begleiter et al., 2004) in most cases we split the data 50-50 into a training set and a test set. One difficulty that occasionally arises with the VLMC and MTD predictors is that they use maximum likelihood parameter estimates for their models, which in some cases means that they assign zero conditional probability to certain symbols, and which in turn results in poor performance and an infinite log-loss. Finally we note that different versions of the CTW algorithm have been used for prediction in earlier work, including Begleiter et al. (2004); Dimitrakakis (2010); Ron et al. (1996).

2.6.1 Simulated data

The experiments here are based on 1,000 samples simulated from the 5th order chain with alphabet size $m = 3$ in Section 2.4.1. Figure 2.13 shows the log-loss achieved by all five predictors as a function of the number of predicted samples in the test set.

The results in the first plot are based on $t = 100$ training samples and $n = 900$ test samples. The BCT predictor is seen to perform consistently better than the other four methods. The difference from the second best method, CTF, increases as more data get predicted, reaching a significant $\approx 3.7\%$ by the end. This is in part due to the fact that the BCT predictor continues to get updated past the training stage, whereas the models employed by the other four methods based on only 100 training samples are too simple to be effective. SMC, VLMC and MTD all produce the empty model corresponding to i.i.d. data, with VLMC and MTD also having the same parameters, inducing the exact same predictors (hence shown as a single graph). CTF uses a first order Markov model, leading to slightly better performance.

The second plot shows the log-loss obtained with $t = 500$ training samples and $n = 500$ test samples. The results are similar, however, the difference between BCT and the other methods is now smaller. Since the training set is larger, the other four methods have a better chance to capture more of the structure present in the data. The VLMC model is a simple tree of depth two, and the CTF model also corresponds to a second order chain. MTD again

produces an i.i.d. model, and SMC identifies a second order model with five states, which is the closest to BCT with a log-loss difference of 2.5%.

With $t = 900$ training and $n = 100$ test samples, BCT again appears to have the best performance, although there are larger fluctuations due to the smaller test data size. The results of SMC, VLMC and CTF are almost identical, and significantly better than MTD. SMC uses a second order model with 4 states, the CTF model also has order 2, MTD selects the i.i.d. model, and VLMC, which is closest to BCT by the end (by a 2.2% difference), produces a tree model of depth 3 with 5 leaves.

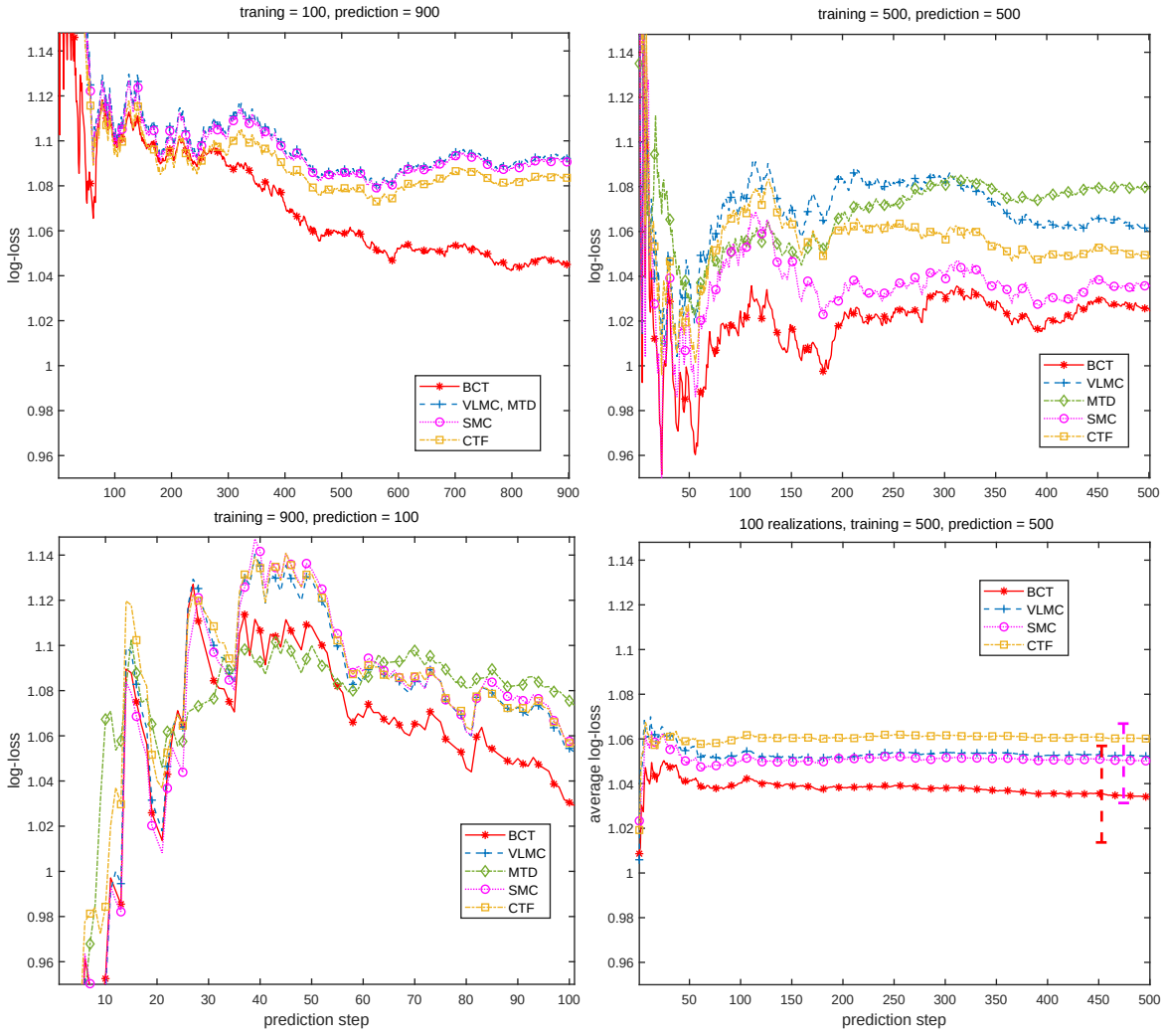


Figure 2.13: Prediction results on simulated data. First three plots: Log-loss achieved by each of the five methods, as a function of the size of the test data. Bottom right: Log-loss achieved by BCT, VLMC, SMC and CTF on $n = 500$ test samples with $t = 500$ training samples, averaged over 100 independent repetitions of the experiment. One-standard-deviation error bars are also plotted for BCT and SMC near the end of the test data. The vertical (log-loss) axis has the same range in all plots.

Finally, we examine the log-loss of these predictors averaged over 100 independent realisations simulated from this chain, each time with a 50-50 split between training and test data. The last plot in Figure 2.13 shows the results obtained by all methods except MTD, which was based on an i.i.d. model every time. The average performance of BCT is better than that of the other methods; after the first 10 samples or so, the BCT log-loss stays consistently approximately 1.6% lower than that of the second best method, SMC.

2.6.2 Real data

SARS-CoV-2 gene. Here we examine the spike (S) gene, in positions 21,563–25,384 of the SARS-CoV-2 genome (Wu et al., 2020) described in Section 2.4.2. The importance of this gene is that it codes for the surface glycoprotein whose function was identified in Lan et al. (2020); Yan et al. (2020) as critical, in that it binds onto the Angiotensin Converting Enzyme 2 (ACE2) receptor on human epithelial cells, giving the virus access to the cell and thus facilitating the Covid-19 disease. The data, consisting of a 3,822 base-pair-long gene sequence, was again split 50-50 into a training set and test set.

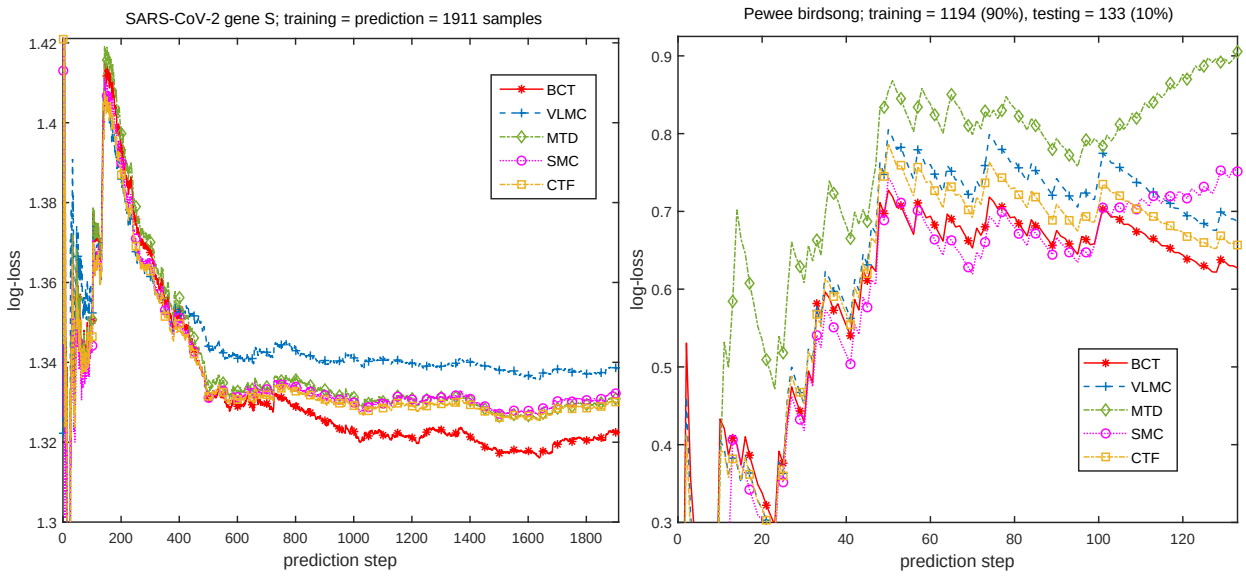


Figure 2.14: Prediction results on real data. Left: SARS-CoV-2 gene (S), with $t = 1911$ training samples and $n = 1911$ test samples. Right: Pewee birdsong data, with $t = 1194$ training samples and $n = 133$ test samples, corresponding to a 90%-training/10%-testing split.

Figure 2.14 shows the prediction results obtained by all methods, with BCT consistently achieving the smallest log-loss. The MAP tree model T_1^* has posterior $\pi(T_1^*|x) \approx 49\%$ and depth 2, while the first order chain appears as T_2^* with $\pi(T_2^*|x) \approx 48\%$, meaning that these two models essentially have the same weight in the Bayesian predictor; cf. (2.17). It is possibly this model averaging which explains, in part, the superior performance of BCT here.

The second best method, CTF, has nearly identical performance with both MTD and SMC, all of them selecting a first order Markov model. The VLMC produces a tree of depth 1 grouping $\{A, G, T\}$ into a single state, which performs worse than all the other methods.

Pewee birdsong. The data, discussed in Section 2.4.2, consists of 1327 samples in a ternary alphabet. When half of it was used for training and the other half for testing, VLMC and MTD both had an infinite log-loss after the 420th symbol in the test sequence; this was mentioned earlier as a possible drawback of methods that use maximum likelihood estimates for the parameters. Among the methods that give a finite log-loss, BCT was found to have the best performance. SMC produced a second order model with five states, and by the end of the test data its log-loss was larger than that of the BCT by 13.2%. CTF used a fourth order model for prediction, resulting in a log-loss larger than that of BCT by 1.2%.

In order to illustrate the relative performance of all five methods, in Figure 2.14 we report the prediction results on the last 10% of the samples, after the first 90% have been used for training. The BCT has the smallest log-loss overall, approximately 4.8% lower than that of the second best method, CTF, at the end of the experiment. The CTF and MTD predictors both use a fourth order model, SMC finds a second order model with 5 states, and VLMC uses a simple tree model of depth 4.

2.6.3 Summary

The BCT predictor was found to have consistently better performance than the other four methods considered, achieving a log-loss that was between 1.2% and 4.8% better than that of the second-best method in each case. This is due, in part, to the fact that the BCT predictor is based on averaging over all models and parameters with respect to their posterior distribution, and that it allows for efficient sequential updates beyond the training stage.

The method that performed the closest to BCT in most cases was CTF, which usually identified the same Markov order as the other methods. The VLMC and MTD predictors were found to be consistently and significantly less effective than BCT, and as noted earlier they sometimes assigned zero probability to the occurrence of certain symbols in the data, resulting in an infinite log-loss. Another difficulty with CTF, VLMC and MTD is that they required much more computational effort in the training stage.

The SMC predictor was found in most cases to have performance similar to VLMC. As the context-tree models in $\mathcal{T}(D)$ are a subset of all SMCs, the fact that BCT performs better than SMC highlights the power of the full Bayesian BCT framework and the importance of being able to compute the exact posterior predictive distribution. In contrast, SMC uses a single (and often poor) estimate of its MAP model, leading to much less satisfactory results.

Chapter 3

Posterior Representations for Bayesian Context Trees

In this chapter we derive explicit representations of the BCT prior and posterior on model space, in terms of simple branching processes. These representations offer a very natural intuition about the underlying modelling that takes place within the BCT framework, from a Bayesian point of view: The BCT prior $\pi_D(T;\beta)$ acts in some sense as a conjugate prior for the class $\mathcal{T}(D)$ of variable-memory chains.

Apart from this appealing interpretation, the branching process representations are also of high practical importance as they are employed to:

(i) Develop a Monte Carlo (MC) sampler that allows to obtain i.i.d. samples from the joint posterior on models and parameters.

(ii) Establish theoretical results that provide some further justifications for the use of the BCT framework.

3.1 Branching process representations

In this section we establish the branching process representations for both the BCT prior $\pi(T) = \pi_D(T;\beta)$ of (2.4) and the resulting posterior on model space, $\pi(T|x)$. We also discuss how the posterior representation leads to efficient samplers that can be used for model selection and estimation.

3.1.1 The prior branching process

Given $D \geq 0$ and $\beta \in (0,1)$, let $T = \{\lambda\}$ consist of only the root node λ and consider the following procedure:

- If $D = 0$, stop.
- If $D > 0$, then, with probability β, mark the root as a leaf and stop, or, with probability $(1-\beta)$, add all m children of λ at depth 1 to T. If $D = 1$, stop.
- If $D > 1$, examine each of the m new nodes, and either mark a node as a leaf with probability β, or add all m of its children to T with probability $(1-\beta)$, independently from node to node.
- Continue recursively, at each step examining all non-leaf nodes at depths strictly smaller than D, until no more eligible nodes remain to be examined.
- Output the resulting tree $T \in \mathcal{T}(D)$.

The above construction is a simple Galton-Watson process (Athreya and Ney, 2004; Harris, 1963) with offspring distribution $(\beta, (1-\beta))$ on $\{0, m\}$, stopped at generation D. The following proposition states that the distribution of a tree T generated by this process is exactly the prior $\pi_D(T;\beta)$. Note that this also implies that the expression for $\pi_D(T;\beta)$ indeed defines a probability distribution on $\mathcal{T}(D)$, giving an alternative proof of Lemma 2.1.

Proposition 3.1. *For any $D \geq 0$ and any $\beta \in (0,1)$, the probability that the above branching process produces any particular tree $T \in \mathcal{T}(D)$ is given by $\pi_D(T;\beta)$ as in* (2.4).

Proof. When $D = 0$, $\mathcal{T}(D)$ consists of a single tree, $T = \{\lambda\}$, which has probability 1 under both π_D and the branching process construction. Assume $D \geq 1$. Note that every tree $T \in \mathcal{T}(D)$ can be viewed as a collection of a number, k, say, of m-branches, since every node in T has either zero or m children. The proposition is proven by induction on k. The result is trivial for $k = 0$, since the only tree with no m-branches is $T = \{\lambda\}$ and its probability under both π_D and the branching process construction is equal to β.

For the inductive step, suppose that the claim of the proposition is true for all the trees with k m-branches, and let $T' \in \mathcal{T}(D)$ consist of $(k+1)$ m-branches. Then T' can be obtained from some $T \in \mathcal{T}(D)$ that has k m-branches, by adding a single m-branch to one of its leaves, s, say. Two cases are considered.

(i) If s is at depth $D-2$ or smaller, then the probability $\pi_b(T')$ of T' under the branching process construction is,

$$\pi_b(T') = \frac{\pi_b(T)}{\beta}(1-\beta)\beta^m = \frac{\pi_D(T;\beta)}{\beta}(1-\beta)\beta^m = \frac{\alpha^{|T|-1}\beta^{|T|-L_D(T)}}{\beta}\alpha^{m-1}\beta^m,$$

where the second equality follows from the inductive hypothesis and the third from the definition of $\pi_D(T;\beta)$. Therefore, since $|T'| = |T| + m - 1$ and no leaves are added at depth D, so that $L_D(T') = L_D(T)$,

$$\pi_b(T') = \alpha^{[|T|+m-1]-1}\beta^{|T|+m-1-L_D(T)} = \alpha^{|T'|-1}\beta^{|T'|-L_D(T')} = \pi_D(T';\beta),$$

as required.

(ii) If s is at depth $D-1$, we similarly find that,

$$\pi_b(T') = \frac{\pi_b(T)}{\beta}(1-\beta) = \frac{\pi_D(T;\beta)}{\beta}(1-\beta) = \frac{\alpha^{|T|-1}\beta^{|T|-L_D(T)}}{\beta}\alpha^{m-1},$$

and since $|T'| = |T| + m - 1$, but now, $L_D(T') = L_D(T) + m$,

$$\pi_b(T') = \alpha^{[|T|+m-1]-1}\beta^{[|T|+m-1]-[L_D(T)+m]} = \alpha^{|T'|-1}\beta^{|T'|-L_D(T')} = \pi_D(T';\beta),$$

completing the proof. □

Apart from being aesthetically appealing, this representation also offers a simple and practical way of sampling trees from $\mathcal{T}(D)$ according to $\pi_D(T;\beta)$. Moreover, using well-known properties of the Galton-Watson process we can perform some direct computations that offer better insight into the nature and specific properties of the BCT prior.

Interpretation and choice of β. The branching process description of $\pi_D(T,\beta)$ further clarifies the role of the hyperparameter β: It is exactly the probability that, when a node is added to the tree T, it is marked as a leaf and its children are not included in T.

In terms of choosing the value of the hyperparameter β appropriately, recall that, for a Galton-Watson process, the expected number of children of each node, in this case $\rho = m(1-\beta)$, governs the probability of extinction P_e: If $\rho \leq 1$ we have $P_e = 1$, whereas if $\rho > 1$, P_e is strictly less than one. Therefore, in the binary case $m = 2$, the original choice $\beta = 1/2$ used in the CTW algorithm gives an expected number of children equal to the critical value $\rho = 1$. This suggests that another reasonable choice for general alphabets (apart from our default $\beta = 1 - 2^{-m+1}$) could be $\beta = 1 - 1/m$, which keeps $\rho = 1$, so that the resulting prior would have similar qualitative characteristics with the well-studied binary case.

Now suppose T is a random model generated by the prior and let $L_d(T)$ denote the number of nodes at depth $d = 0, 1, \ldots, D$. Then, standard Galton-Watson theory (Athreya and Ney, 2004; Harris, 1963) provides the useful expressions,

$$\mathbb{E}\left[L_d(T)\right] = \rho^d, \qquad \mathrm{Var}\left[L_d(T)\right] = \begin{cases} \sigma^2 d, & \text{if } \rho = 1, \\ \sigma^2 \rho^{d-1} \frac{1-\rho^d}{1-\rho}, & \text{if } \rho \neq 1, \end{cases}$$

where $\sigma^2 = m^2\beta(1-\beta)$.

3.1.2 The posterior branching process

Given a time series $x = x_{-D+1}^n$, a maximum depth D, and $\beta \in (0,1)$, for any context s with length strictly smaller than D we define the *branching probabilities* $P_{b,s}$ as,

$$P_{b,s} := \frac{\beta P_{e,s}}{P_{w,s}}, \tag{3.1}$$

where the estimated and weighted probabilities, $P_{e,s}$ and $P_{w,s}$, are defined in (2.7)-(2.9), and with the convention that $P_{b,s} = \beta$ for all contexts s that do not appear in x.

Starting with $T = \{\lambda\}$, the following construction produces a sample model $T \in \mathcal{T}(D)$ from the model posterior $\pi(T|x)$:

- If $D = 0$, stop.
- If $D > 0$, then, with probability $P_{b,\lambda}$, mark the root as a leaf and stop, or, with probability $(1 - P_{b,\lambda})$, add all m children of λ at depth 1 to T. If $D = 1$, stop.
- If $D > 1$, examine each of the m new nodes and either mark a node s as a leaf with probability $P_{b,s}$, or add all m of its children to T with probability $(1 - P_{b,s})$, independently from node to node.
- Continue recursively, at each step examining all non-leaf nodes at depths strictly smaller than D, until no more eligible nodes remain.
- Output the resulting tree $T \in \mathcal{T}(D)$.

Proposition 3.2. *For any $D \geq 0$ and any $\beta \in (0,1)$, the probability that the above branching process produces any particular tree $T \in \mathcal{T}(D)$ is given by $\pi(T|x)$.*

Proof. The proof follows along the same lines as that of Proposition 3.1; it is again by induction on the number of m-branches. For the base case $k = 0$ we also need the result of Theorem 2.1 for the prior predictive likelihood, $P(x) = P_{w,\lambda}$. The complete proof of the proposition is given in Appendix B.2. □

Conjugate prior. It is perhaps somewhat remarkable that the posterior $\pi(T|x)$ on the vast model space $\mathcal{T}(D)$ admits such a simple explicit description. Indeed, the posterior branching process is of exactly the same form as that of the prior, which can then naturally be viewed as a conjugate prior on $\mathcal{T}(D)$.

Model posterior probabilities. Proposition 3.2 allows us to write an exact expression for the posterior of any model $T \in \mathcal{T}(D)$ in terms of the branching probabilities $P_{b,s}$,

$$\pi(T|x) = \prod_{s \in T_o} \left(1 - P_{b,s}\right) \prod_{s \in T} P_{b,s}, \tag{3.2}$$

where T_o denotes the set of all *internal* nodes of T, and with the convention that $P_{b,s} = 1$ for all leaves of T at depth $d = D$. This expression will be the starting point in the proofs of the asymptotic results of Section 3.2 for $\pi(T|x)$.

3.1.3 Sampling from the posterior

In terms of inference, the main utility of Proposition 3.2 is that it offers a practical way of obtaining exact i.i.d. samples $\{T^{(i)}\}$ from the model posterior $\pi(T|x)$.

And since the full conditional density of the parameters $\pi(\theta|T,x)$ is explicitly identified in (2.16) as a product of Dirichlet densities,

$$\pi(\theta|T,x) = \prod_{s \in T} \mathrm{Dir}\left(1/2 + a_s(0), 1/2 + a_s(1), \ldots, 1/2 + a_s(m-1)\right),$$

for each $T^{(i)}$ we can draw a conditionally independent sample $\theta^{(i)} \sim \pi(\theta|T^{(i)},x)$, producing a sequence of exact i.i.d. samples $\{(T^{(i)}, \theta^{(i)})\}$ from the joint posterior $\pi(T,\theta|x)$.

This facilitates numerous applications. For example, effective parameter estimation can be performed by simply keeping the samples $\{\theta^{(i)}\}$, which come from the marginal posterior distribution $\pi(\theta|x)$. Similarly, Markov order estimation can be performed by collecting the sequence of maximum depths of the models $\{T^{(i)}\}$. And in model selection tasks, the model posterior can be extensively explored, offering better insight and deeper understanding of the underlying structure and dependencies present in the data.

Although a family of MCMC samplers was introduced and successfully used for the same tasks in Section 2.5, MCMC sampling has well-known limitations and drawbacks, including potentially slow mixing, high correlation between samples, and the need for convergence diagnostics (Cowles and Carlin, 1996; Gelman and Rubin, 1992; Robert and Casella, 2004). Partly for these reasons, being able to obtain i.i.d. samples from the posterior is generally much more desirable, as illustrated in Section 3.3.

Estimation of general functionals. Consider the general Bayesian estimation problem, where the goal is to estimate an arbitrary functional $F = F(T, \theta)$ of the underlying variable-memory chain, based on data x. Using the above sampler, the entire posterior distribution of the statistic F can be explored, by considering the i.i.d. samples $F^{(i)} = F(T^{(i)}, \theta^{(i)})$, distributed according to the desired posterior $\pi(F|x)$.

In connection with classical estimation techniques, and in order to evaluate estimation performance more easily in practice, several reasonable point estimates can also be obtained. The most common choices are the empirical average approximation to the posterior mean,

$$\widehat{F}_{\mathrm{MC}} = \frac{1}{N}\sum_{i=1}^{N} F^{(i)} = \frac{1}{N}\sum_{i=1}^{N} F(T^{(i)}, \theta^{(i)}), \tag{3.3}$$

or the posterior mode, i.e., the maximum *a posteriori* probability (MAP) estimate, $\widehat{F}_{\mathrm{MAP}}$. In cases where the conditional mean $\bar{F}(T) = E\big(F(T,\theta)|x,T\big)$ can be computed for any model T (as e.g. in the case of parameter estimation), a lower-variance Rao-Blackwellised estimate for the posterior mean can also be obtained as,

$$\widehat{F}_{\mathrm{RB}} = \frac{1}{N}\sum_{i=1}^{N} \bar{F}(T^{(i)}).$$

Importantly, as this posterior sampler provides access to the entire posterior distribution $\pi(F|x)$ of the statistic of interest F, standard Bayesian methodology can be applied to quantify the resulting uncertainty of any estimator $\hat{F}$, for example by obtaining credible intervals in terms of the posterior $\pi(F|x)$. An interesting special case of the above methodology is the estimation of the entropy rate of the underlying process, $F(T,\theta) = H(T,\theta)$, which will be studied in detail in Chapter 4.

3.2 Theoretical results

In this section, using the branching process representation in Proposition 3.2, we show how to derive precise results on the asymptotic behaviour of the BCT posterior $\pi(T|x)$ on model space, and provide an explicit expression for the posterior predictive distribution.

Let $\{X_n\}$ be a variable-memory chain with model $T \in \mathcal{T}(D)$. The specific model T that describes the chain is typically not unique, for the same reason, e.g., that every i.i.d. process can also trivially be described as a first-order Markov chain: Adding m children to any leaf of T which is not at maximal depth, and giving each of them the same parameters as their parent, leaves the distribution of the chain unchanged.

The natural main goal in model selection is to identify the "minimal" model, i.e., the smallest model that can fully describe the distribution of the chain. A model $T \in \mathcal{T}(D)$ is called *minimal* if every m-tuple of leaves $\{sj\,;\ j = 0,1,\ldots,m-1\}$ in T contains at least two with non-identical parameters, i.e., there are $j \neq j'$ such that $\theta_{sj} \neq \theta_{sj'}$. It is easy to see that every Dth order Markov chain $\{X_n\}$ has a unique minimal model $T^* \in \mathcal{T}(D)$.

A variable-memory chain $\{X_n\}$ with model $T \in \mathcal{T}(D)$ and with associated parameters $\theta = \{\theta_s; s \in T\}$ is *ergodic*, if the corresponding first-order chain $\{Z_n := X_{n-D+1}^n; n \geq 1\}$ taking values in A^D is irreducible and aperiodic. In order to avoid uninteresting technicalities, in most of our results we will assume that the data are generated by a *positive-ergodic* chain $\{X_n\}$, namely that all its parameters $\theta_s(j)$ are nonzero, so that its unique stationary distribution π gives strictly positive probability to all finite contexts s.

3.2.1 Posterior consistency and concentration

Our first theorem is a strong consistency result, which states that, if the data $x = x_{-D+1}^n$ are generated by an ergodic chain with minimal model $T^* \in \mathcal{T}(D)$, then the model posterior asymptotically almost surely (*a.s.*) concentrates on T^*. Theorem 3.1 both strengthens and generalises a weaker result on the asymptotic behaviour of the MAP model established in (Willems et al., 1993c, Theorem 8).

Theorem 3.1. *Let $X_{-D+1}^n = (X_{-D+1},\ldots,X_0,X_1,\ldots,X_n)$ be a time series generated by a positive-ergodic, variable-memory chain $\{X_n\}$ with minimal model $T^* \in \mathcal{T}(D)$. For any value of $\beta \in (0,1)$, the posterior distribution over models concentrates on T^*, i.e.,*

$$\pi(T^*|X_{-D+1}^n) \to 1, \quad \text{a.s., as } n \to \infty.$$

Proof. Recalling the posterior representation in (3.2), it suffices to show that, as $n \to \infty$, $P_{b,s} \to 0$ a.s. for all internal nodes of T^*, and $P_{b,s} \to 1$ a.s. for all leaves of T^*. These two claims are established in Lemmas 3.2 and 3.3 below. □

In Lemma 3.1, we first recall simple bounds on the estimated probabilities $P_{e,s}$; see Krichevsky and Trofimov (1981); Xie and Barron (2000) and (Catoni, 2004, Ch. 1). Then, in Lemmas 3.2 and 3.3 we establish the asymptotic behaviour of $P_{b,s}$, considering separately the cases of internal nodes and leaves of T^*; Lemma 3.2 is a generalisation of (Jiao et al., 2013, Lemma 12).

Lemma 3.1. *For every node s with count vector $a_s = (a_s(0), a_s(1), \ldots, a_s(m-1))$ and $M_s = a_s(0) + \cdots + a_s(m-1)$, the estimated probabilities $P_{e,s} = P_e(a_s)$ of (2.7) satisfy:*

$$\log P_{e,s} \geq \sum_{j=0}^{m-1} a_s(j) \log \frac{a_s(j)}{M_s} - \frac{m-1}{2} \log M_s - \log m; \tag{3.4}$$

$$\log P_{e,s} \leq \sum_{j=0}^{m-1} a_s(j) \log \frac{a_s(j)}{M_s} - \frac{m-1}{2} \log \frac{M_s}{2\pi} - \log \frac{\pi^{m/2}}{\Gamma(m/2)}. \tag{3.5}$$

Lemma 3.2. *Under the assumptions of Theorem 3.1, for every internal node s of T^*, the branching probability $P_{b,s} \to 0$, a.s., as $n \to \infty$.*

Proof. As the complete proof is quite involved, only the main and more interesting part of the argument is given here; the remaining details are given in Appendix B.2. We begin by observing that,

$$P_{b,s} = \frac{\beta P_{e,s}}{P_{w,s}} = \frac{\beta P_{e,s}}{\beta P_{e,s} + (1-\beta) \prod_j P_{w,sj}} = \frac{1}{1 + (1-\beta)/\beta \prod_j P_{w,sj}/P_{e,s}} \tag{3.6}$$

$$\leq \frac{1}{1 + c_0 \prod_j P_{e,sj}/P_{e,s}}, \tag{3.7}$$

for some constant c_0, where in the last step we used that either $P_{w,sj} \geq \beta P_{e,sj}$ or $P_{w,sj} = P_{e,sj}$. Therefore, it suffices to show that $P_{e,s} \big/ \prod_j P_{e,sj} \to 0$, a.s., as $n \to \infty$.

Let s be a fixed finite context, and let X and J denote the random variables corresponding to the symbols that follow and precede s, respectively, under the stationary distribution of $\{X_n\}$. Using Lemma 3.1 and the ergodic theorem for Markov chains (Chung, 1967), it is shown in Appendix B.2 that,

$$\log P_{e,s} - \sum_j \log P_{e,sj} = -nI(X;J|s)\pi(s) + o(n), \quad \text{a.s.}, \tag{3.8}$$

where $I(X;J|s)$ is the conditional mutual information between X and J given s (Cover and Thomas, 1999). This mutual information is always nonnegative, and it is zero if and only if X and J are conditionally independent given s.

For any internal node s that is a parent of leaves of T^*, the minimality of T^* implies that X and J are not conditionally independent given s, as there exist $j \neq j'$ such that $\theta_{sj} \neq \theta_{sj'}$, so that θ_{sj} depends on j. Therefore, $I(X;J|s) > 0$ and $\pi(s) > 0$ by assumption, so (3.8) implies that $\log P_{e,s} - \sum_j \log P_{e,sj} \to -\infty$, a.s., as required.

For the general case of internal nodes that may not be parents of leaves, a simple iterative argument is given in Appendix B.2 establishing the same result in that case as well, and completing the proof of the lemma. □

Lemma 3.3. *Under the assumptions of Theorem 3.1, for every leaf s of T^*, the branching probability $P_{b,s} \to 1$, a.s., as $n \to \infty$.*

Proof. As with the previous lemma, we only give an outline of the main interesting steps in the proof here; complete details are provided in Appendix B.2.

By definition, for any leaf s of T^* (and also for any 'external' node s, that is, any context s not in T^*), we have $I(X;J|s) = 0$ because of conditional independence. Therefore, we need to consider the higher-order terms in the asymptotic expansion of (3.8). Using Lemma 3.1, the ergodic theorem, and the law of the iterated logarithm (LIL) for Markov chains (Chung, 1967), it is shown in Appendix B.2 that here,

$$\sum_j \log P_{e,sj} - \log P_{e,s} \leq -\frac{(m-1)^2}{2} \log n + O(\log\log n), \quad \text{a.s.} \tag{3.9}$$

This implies that $\sum_j \log P_{e,sj} - \log P_{e,s} \to -\infty$, so that $\prod_j P_{e,sj}/P_{e,s} \to 0$, a.s.

The last step of the proof, namely that for any leaf s of T^*, $\prod_j P_{e,sj}/P_{e,s} \to 0$ also implies that $\prod_j P_{w,sj}/P_{e,s} \to 0$, a.s., so that, by (3.6), $P_{b,s} \to 1$, a.s., is given in Appendix B.2. □

Our next result is a refinement of Theorem 3.1, which characterises the rate at which the posterior probability of T^* converges to 1.

Theorem 3.2. *Let X_{-D+1}^n be a time series generated by a positive-ergodic variable-memory chain $\{X_n\}$ with minimal model $T^* \in \mathcal{T}(D)$. For any value of the prior hyperparameter $\beta \in (0,1)$ and any $\varepsilon > 0$, we have, as $n \to \infty$:*

$$\pi(T^*|X_{-D+1}^n) = 1 - O\left(n^{-\frac{(m-1)^2}{2}+\varepsilon}\right), \quad \text{a.s.}$$

Proof. The proof of Theorem 3.2 is given in Appendix B.2. It follows from considering the rate at which $P_{b,s}$ converges to either 0 or 1, depending on whether the node s is an internal node or a leaf of T^*. □

In fact, as discussed in Appendix B.2, the proof of Theorem 3.2 also reveals that a stronger statement can be made for the convergence rate in the case of full Dth order Markov chains, as in the corollary below.

Corollary 3.1. *If $\{X_n\}$ is a genuinely Dth order chain in that its minimal model T^* is the complete tree of depth D, then its posterior probability $\pi(T^*|X_{-D+1}^n)$ almost surely converges to 1 at an exponential rate.*

3.2.2 Out-of-class modelling

In this section we consider the behaviour of the posterior distribution $\pi(T|x)$ on models $T \in \mathcal{T}(D)$ when the time series x is *not* generated by a chain from the model class $\mathcal{T}(D)$, but from a general *stationary* and *ergodic* process $\{X_n\}$ with possibly infinite memory. We first give an explicit description of the "limiting" model $T_\infty \in \mathcal{T}(D)$ on which the posterior $\pi(T|x)$ concentrates when the observations are generated by a general process outside $\mathcal{T}(D)$, and then we give conditions under which T_∞ is structurally "as close as possible" to the true underlying model.

Description of T_∞. Recall that, for any context s, we write X and J for the random variables corresponding to the symbols that follow and precede s, respectively. The limiting tree $T_\infty \in \mathcal{T}(D)$ corresponding to a general stationary process $\{X_n\}$ with values in A can be constructed via the following procedure:

- Take T_∞ to be the empty tree.
- Starting with the nodes at depth $d = D-1$, for each such s, if $I(X;J|s) > 0$, then add s to T_∞ along with all its children and all its ancestors; that is, add the complete path from the root λ to the children of s.
- After all nodes s at depth $d = D-1$ have been examined, examine all possible nodes s at depth $d = D-2$ that are not already included in T_∞, and repeat the same process.
- Continue recursively towards the root, until all nodes at all depths $0 \leq d \leq D-1$ have been examined.
- For any node already in T_∞ at depth $d \leq D-1$, such that only some but not all m of its children are included in T_∞, add the missing children to T_∞ so that it becomes proper.
- Output T_∞.

In order to state our results we need to impose two additional conditions on the underlying data-generating process. Suppose $\{X_n\}$ is stationary. Without loss of generality (by Kolmogorov's extension theorem) we may consider the two-sided version of the process, $\{X_n\,;\, n \in \mathbb{Z}\}$. Its α-mixing coefficients (Ibragimov, 1962; Philipp and Stout, 1975) are defined as,

$$\alpha_n = \sup_{A \in \mathcal{A},\, B \in \mathcal{B}} |\mathbb{P}(A \cap B) - \mathbb{P}(A)\mathbb{P}(B)|, \tag{3.10}$$

where $\mathcal{A}$ and $\mathcal{B}$ denote the σ-algebras, $\sigma(\ldots, X_{-1}, X_0)$ and $\sigma(X_n, X_{n+1}, \ldots)$, respectively.

We will need a mixing condition and a positivity condition for our results:

$$\sum_{n=1}^{\infty} \alpha_n^{\delta/(2+\delta)} < \infty \text{ for some } \delta > 0, \quad \text{and} \quad \mathbb{P}(X_0^D = x_0^D) > 0 \text{ for all } x_0^D \in A^{D+1}. \tag{3.11}$$

Theorem 3.3. *Let X_{-D+1}^n be a time series generated by a stationary ergodic process $\{X_n\}$ satisfying the assumptions* (3.11)*, and let $T_\infty \in \mathcal{T}(D)$ be given by the above construction. Then, for any value of $\beta \in (0,1)$ we have:*

$$\pi(T_\infty | X_{-D+1}^n) \to 1, \quad \textit{a.s., as } n \to \infty.$$

Proof. The proof of Theorem 3.3 follows along exactly the same lines as the earlier proof of Theorem 3.1. Instead of the ergodic theorem for Markov chains (Chung, 1967) we now use Birkhoff's ergodic theorem (Breiman, 1992), and instead of the LIL for Markov chains (Chung, 1967) we apply the general LIL for functions of blocks of an ergodic process, which follows, as usual, from the almost-sure invariance principle (Philipp and Stout, 1975; Rio, 1995; Zhao and Woodroofe, 2008). The mixing condition in (3.11) was chosen as one of the simplest ones that guarantee this general version of the LIL.

Regarding the overall structure of the proof, all the earlier asymptotic expansions of the branching probabilities still remain valid, including (3.8) and (3.9). The only possible difference might be at the boundary conditions that are required as a starting point for the iterative argument in the proof of Lemma 3.2, since the actual "leaves" of the true underlying model of $\{X_n\}$ are not necessarily at depth $d \leq D$ here. But, as before, all branching probabilities $P_{b,s}$ tend either to 0 or to 1, depending on whether the mutual information condition that appears in the description of T_∞ holds or not. Specifically, starting from nodes s at depth $d = D-1$, if $I(X;J|s) = 0$ then we are in the same situation as in Lemma 3.3, so that $P_{b,s} \to 1$, and all children of s are pruned. On the other hand, if $I(X;J|s) > 0$, then $P_{b,s} \to 0$, and by the same iterative argument, $P_{b,u} \to 0$ for all ancestors u of node s as well. □

Analogous comments apply to the proof of Theorem 3.4, which is again a refinement characterising the rate at which the posterior probability of T_∞ converges to 1.

Theorem 3.4. *Let X_{-D+1}^n be a time series generated by a stationary ergodic process $\{X_n\}$ satisfying the assumptions* (3.11)*, and let T_∞ be its limiting model in $\mathcal{T}(D)$. Then, for any $\beta \in (0,1)$ and any $\varepsilon > 0$, as $n \to \infty$ we have:*

$$\pi(T_\infty | X_{-D+1}^n) = 1 - O\left(n^{-\frac{(m-1)^2}{2}+\varepsilon}\right), \quad \textit{a.s.}$$

In general, it is natural to expect that T_∞ should be "as close as possible" in some sense to the true underlying model T^*, and this is indeed what is most often observed in applications: i.e., T_∞ being the same as T^* truncated at depth D.

But this is not always the case. For instance, recall the 3rd order chain $\{X_n\}$ studied in Example 2.2, also described as having a "bimodal posterior". There, T^* is the complete m-ary tree of depth 3 (with $m = 6$), but X_n depends on $(X_{n-1}, X_{n-2}, X_{n-3})$ only via X_{n-3}. For that reason, the limiting model T_∞ in $\mathcal{T}(D)$ with $D = 1$ or $D = 2$ is not T^* truncated at depth D, but rather the empty tree $\{\lambda\}$ consisting of only the root node λ.

Fortunately, we can read a simple necessary and sufficient condition for the "expected" behaviour to occur from the definition of T_∞ itself. Let $\{X_n\}$ be a stationary and ergodic process on the finite alphabet A. From the description of Csiszár and Talata (2006), it is easy to see that there is a unique minimal context-tree model T^* for $\{X_n\}$ of possibly infinite depth. Let $T^*_{|D}$ be T^* truncated at depth D, and write $\mathcal{N}_{D-1}(T^*)$ for the set of all internal nodes of T^* at depth $d = D-1$ whose children exist in T^* (at depth D) but are not leaves of T^*.

Corollary 3.2. *The limiting tree model is $T_\infty = T^*_{|D}$ if and only if $I(X;J|s) > 0$ for all the nodes $s \in \mathcal{N}_{D-1}(T^*)$.*

Proof. The result follows directly from the definition of T_∞ together with the observation that the condition $I(X;J|s) > 0$ is already satisfied for all nodes s of T^* at depth $d = D-1$ whose children are leaves of T^*. □

3.2.3 The posterior predictive distribution

The branching process representation of the posterior can also be used to facilitate practically useful computations. In Section 2.6, the CTW algorithm was employed to compute the posterior predictive distribution $P^*_D(x_{n+1}|x_1^n)$. In Proposition 3.3, we also give an explicit expression for $P^*_D(x_{n+1}|x_1^n)$ in terms of the branching probabilities $P_{b,s}$.

Proposition 3.3. *The posterior predictive distribution $P^*_D(x_{n+1}|x_1^n)$ of (2.17) satisfies,*

$$P^*_D(x_{n+1}|x_1^n) = \sum_{i=0}^{D} \left(\frac{a_{s^{(i)}}(x_{n+1}) + 1/2}{M_{s^{(i)}} + m/2} \right) \gamma_i, \tag{3.12}$$

where, for $0 \le i \le D$, the string $s^{(i)}$ is the context of length i preceding x_{n+1}, and γ_i is the posterior probability that node $s^{(i)}$ is a leaf, given by,

$$\gamma_i = \begin{cases} P_{b,\lambda}, & i = 0, \\ \prod_{k=0}^{i-1} \left(1 - P_{b,s^{(k)}}\right) P_{b,s^{(i)}}, & 1 \le i \le D-1, \\ \prod_{k=0}^{D-1} \left(1 - P_{b,s^{(k)}}\right), & i = D. \end{cases} \tag{3.13}$$

Proof. Writing $x = x_1^n$, the posterior predictive distribution can be expressed as,

$$P_D^*(x_{n+1}|x) = \sum_{T \in \mathcal{T}(D)} P(x_{n+1}|T,x)\,\pi(T|x). \tag{3.14}$$

For any tree $T \in \mathcal{T}(D)$, exactly one of the contexts $s^{(i)}$, $0 \le i \le D$, is a leaf of the tree. For every $0 \le i \le D$, define the subset $\mathcal{T}_i(D) \subset \mathcal{T}(D)$ to be the collection of trees $T \in \mathcal{T}(D)$ such that the context of x_{n+1} that is a leaf of T is $s^{(i)}$; these $\mathcal{T}_i(D)$ are disjoint and their union is $\mathcal{T}(D)$. The key observation here is that $P(x_{n+1}|T,x)$ is the same for all trees $T \in \mathcal{T}_i(D)$, since,

$$\begin{aligned} P_D^*(x_{n+1}|x) &= \int_\theta P(x_{n+1}|T,\theta,x)\pi(\theta|T,x)\,d\theta \\ &= \int_{\theta_{s^{(i)}}} P(x_{n+1}|T,\theta_{s^{(i)}},x)\pi(\theta_{s^{(i)}}|T,x)\,d\theta_{s^{(i)}} \\ &= \int_{\theta_{s^{(i)}}} \theta_{s^{(i)}}(x_{n+1})\,\pi(\theta_{s^{(i)}}|T,x)\,d\theta_{s^{(i)}} = \frac{a_{s^{(i)}}(x_{n+1}) + 1/2}{M_{s^{(i)}} + m/2}, \end{aligned} \tag{3.15}$$

where we used the full conditional density of the parameters in (2.16). So, from (3.14),

$$P_D^*(x_{n+1}|x) = \sum_{i=0}^{D} \sum_{T \in \mathcal{T}_i(D)} P(x_{n+1}|T,x)\,\pi(T|x) = \sum_{i=0}^{D} \frac{a_{s^{(i)}}(x_{n+1}) + 1/2}{M_{s^{(i)}} + m/2} \sum_{T \in \mathcal{T}_i(D)} \pi(T|x),$$

which completes the proof upon noticing that the last sum $\sum_{T \in \mathcal{T}_i(D)} \pi(T|x)$ is exactly the posterior probability that node $s^{(i)}$ is a leaf, namely, γ_i as in (3.13). □

3.3 Experimental results

Being able to obtain exact i.i.d. samples from the posterior is generally more desirable and typically leads to more efficient estimation than using approximate MCMC samples. In this section we offer empirical evidence justifying this statement in the present setting through a simple simulation example.

In particular, we will compare the performance of the i.i.d. sampler of Section 3.1.3 with the MCMC samplers of Section 2.3.4. The simulated data set that we consider consists of $n = 1{,}000$ samples generated from the ternary 5th order chain of Section 2.4.1. A simple and effective convergence diagnostic here (which can also be viewed as an example of an estimation problem) is the examination of the frequency with which the MAP model, T_1^*, appears in the i.i.d. or the MCMC sample trajectory. Of course, the MAP model T_1^* and its exact posterior probability $\pi(T_1^*|x)$ are obtained using the BCT algorithm.

As shown in Figure 3.1, the estimates based on both the i.i.d. sampler of Section 3.1.3 and the RW sampler of Section 2.3.4 appear to converge, with the corresponding i.i.d. estimates converging significantly faster. In 50 independent repetitions of the same experiment (with $N = 1,000$ simulated samples in each run), the estimated variance of the MCMC estimates was found to be much larger than that for the i.i.d. estimates, by a factor of around 60.

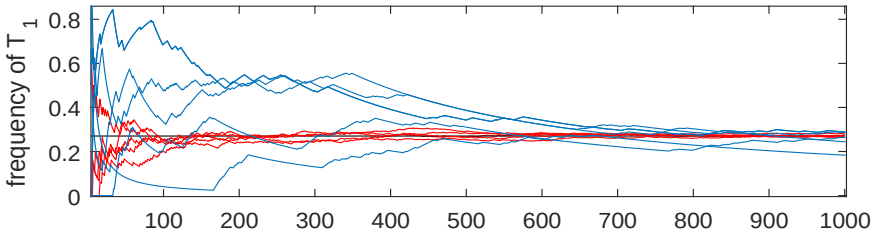


Figure 3.1: Frequency of T_1^*. Blue: MCMC estimates. Red: i.i.d. estimates. In each case, the five graphs correspond to five independent repetitions of the experiment with $N = 1,000$ simulated samples. The horizontal line is the limiting frequency, $\pi(T_1^*|x)$.

Figure 3.2 shows the trace plots (Roy, 2020) obtained from $N = 10,000$ simulated samples from the MCMC and i.i.d. samplers, which can be used to monitor the log-posterior in each case. It is immediately evident that the i.i.d. sampler is more efficient in exploring the effective support of the posterior.

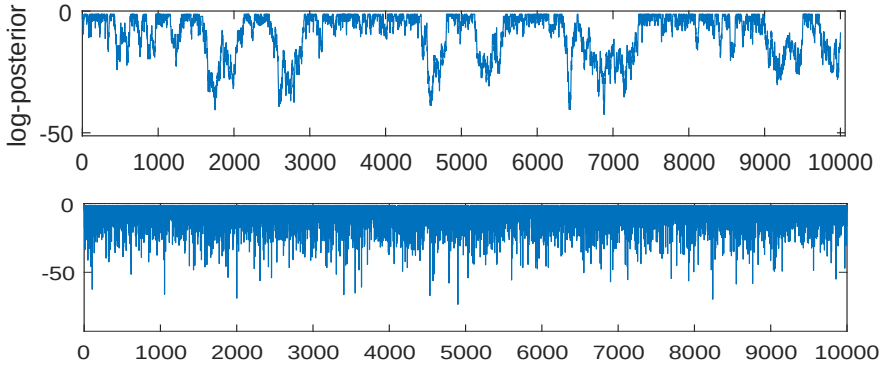


Figure 3.2: Trace plots showing the log-posterior, $\log \pi(T^{(i)}|x)$, at each iteration. Top: MCMC samples. Bottom: i.i.d. samples.

As expected, the i.i.d. sampler has superior performance compared to the MCMC sampler, both in terms of estimation and in terms of mixing. Also, although the two types of samplers have comparable complexity in terms of computation time and memory requirements, the structure of the i.i.d. sampler is much simpler, giving a much easier implementation. In view of these observations, in the following chapter we only employ the i.i.d. sampler for the purposes of entropy estimation.

Chapter 4

Truly Bayesian Entropy Estimation

The *entropy rate* $\bar{H}$ of a stochastic process $\{X_n\}$ on a finite alphabet is defined as,

$$\bar{H} = \lim_{n\to\infty} \frac{1}{n} H(X_1^n), \tag{4.1}$$

whenever the limit exists, where $H(X_1^n)$ denotes the usual Shannon entropy of the discrete random vector $X_1^n = (X_1, \ldots, X_n)$.

In this chapter we consider the challenging task of estimating the entropy rate from empirical data. This is an important and timely problem that has received a lot of attention in the recent literature, perhaps most notably in connection with neuroscience (Archer et al., 2013; Paninski, 2003; Strong et al., 1998; Timme and Lapish, 2018), where the entropy rate plays a crucial role in efforts to describe and quantify the amount of information communicated between neurons. Other important areas of applications include natural language modelling (Willems et al., 2016), animal communication (Kershenbaum, 2014), cryptography (Simion, 2020), genomics (Schmitt and Herzel, 1997), image processing (Hero and Michel, 1999), and web navigation (Levene and Loizou, 2003); see also the recent literature reviews by Verdú (2019) and Feutrill and Roughan (2021).

Building on the BCT framework, here we develop a fully-Bayesian approach for entropy estimation. Since the entropy rate is a functional of the model and the parameters, the branching process sampler of Section 3.1.3 makes it possible to effectively sample from the induced posterior on the entropy rate. This can be used to estimate the entire posterior distribution $\pi(\bar{H}|x)$, hence providing much richer information than simple point estimates. The practical utility of the BCT entropy estimator is illustrated on both simulated and real-world data, where it is found to outperform state-of-the-art alternatives. We also establish theoretical results for the entropy-rate posterior $\pi(\bar{H}|x)$, including proofs of consistency and asymptotic normality, which further justify our methodology.

4.1 Background

Throughout the years, a very wide variety of approaches have been developed for the task of estimating the entropy rate. A recent extensive review of the relevant literature can be found in Verdú (2019) and Feutrill and Roughan (2021). In this section, we briefly describe the most successful earlier approaches among them.

Plug-in estimator. Motivated by the definition of the entropy rate, the simplest and one of the most commonly used estimators is the per-sample entropy of the empirical distribution of k-blocks. Letting $\widehat{p}_k(y_1^k)$, $y_1^k \in A^k$, denote the empirical distribution of k-blocks induced by the data on A^k, the *plug-in* or *maximum-likelihood* estimator is simply,

$$\widehat{H}_k = \frac{1}{k} H(\widehat{p}_k).$$

The main advantage of this estimator is its simplicity. However, this approach is well-known to perform poorly in practice, mainly because of the undersampling problem, i.e., the difficulty in obtaining accurate probability estimates for large block-lengths (Nemenman et al., 2004; Paninski, 2004; Verdú, 2019). Well-known drawbacks include its high variance due to undersampling, and the difficulty in choosing appropriate block-lengths k effectively. Another limitation is its always negative bias, with a number of different approaches developed to correct for it; see Paninski (2003); Verdú (2019) for more details.

Universal compression. In view of the Shannon-McMillan-Breiman theorem (Cover and Thomas, 1999), it is easy to see that, for a string $x_1^n = (x_1, \ldots, x_n)$, any consistent probability estimate $\widehat{p}_n(x_1^n)$ leads to an estimate for the entropy rate as,

$$\widehat{H} = -\frac{1}{n} \log \widehat{p}_n(x_1^n).$$

In this setting, the adaptive probability assignments of universal compressors like the CTW algorithm and prediction by partial matching (PPM) (Cleary and Witten, 1984) are natural choices for the probability estimates to be employed.

Naive CTW estimator. This uses as a probability estimate the prior predictive likelihood $P_D^*(x_1^n)$ computed by the CTW algorithm, to define,

$$\widehat{H}_{\mathrm{CTW}} = -\frac{1}{n} \log P_D^*(x_1^n).$$

This estimator was found to achieve good performance in practice with binary data (Gao et al., 2008; Verdú, 2019). Its consistency and asymptotic normality follow easily from standard results, and its (always positive) bias is of $O((\log n)/n)$.

PPM estimator. Using a different adaptive probability assignment, $Q(x_1^n)$, this method forms the entropy estimate $\widehat{H}_{\mathrm{PPM}} = -(1/n)\log Q(x_1^n)$, where prediction by partial matching is used to fit the model that leads to $Q(x_1^n)$. In our experiments we use the interpolated smoothing variant of PPM introduced by Bunton (1996).

Lempel-Ziv estimators. A different family of estimators is based on string-matching and the fundamental results of Lempel-Ziv (LZ) (Wyner and Ziv, 1989; Ziv and Lempel, 1977) for the relation of match-lengths with the entropy rate; the most effective such estimators are described by Kontoyiannis et al. (1998) and Gao et al. (2008). Among the numerous estimators that have been derived from the LZ family of compression algorithms, the increasing-window estimator of Gao et al. (2008), has been identified as the most effective one, and it will be used in our comparisons: For every position i in the observed data, let ℓ_i denote the length of the longest segment $x_i^{i+\ell_i-1}$ starting at i which also appears somewhere in the window x_0^{i-1} preceding i. Writing $L_i = 1+\ell_i$ for each i, the relevant estimator is,

$$\widehat{H}_{\mathrm{LZ}} = \frac{1}{n}\sum_{i=2}^{n}\frac{\log i}{L_i}.$$

4.2 The BCT entropy estimator

Our starting point here is the fact that for any ergodic variable-memory chain with model $T \in \mathcal{T}(D)$ and parameters θ, the entropy rate can be expressed as an explicit function, $\bar{H} = H(T,\theta)$. This observation follows from the standard expression of the entropy rate for an ergodic, first-order Markov chain $\{X_n\}$, namely (Cover and Thomas, 1999),

$$\bar{H} = -\sum_{i,j\in S}\pi(i)P_{ij}\log P_{ij}, \tag{4.2}$$

where S is the state space of $\{X_n\}$, and (P_{ij}) and $(\pi(i))$ denote its transition matrix and its stationary distribution, respectively. An analogous formula can be written for the entropy rate of an ergodic variable-memory chain $T \in \mathcal{T}(D)$, by viewing it as a full Dth order chain and considering blocks of length $(D+1)$, as usual; cf. Cover and Thomas (1999).

The resulting expression allowing us to compute the entropy rate as a function of (T,θ) opens the door to performing fully-Bayesian entropy estimation. In particular, given a time series x, using the sampling procedure of Section 3.1.3 to produce i.i.d. samples $(T^{(i)},\theta^{(i)})$ from $\pi(T,\theta|x)$, we can simply obtain i.i.d. samples $H^{(i)} = H(T^{(i)},\theta^{(i)})$ from the posterior $\pi(\bar{H}|x)$ of the entropy rate. The calculation of each $H^{(i)} = H(T^{(i)},\theta^{(i)})$ is straightforward and only requires the computation of the stationary distribution π of the induced first-order chain that corresponds to taking blocks of size $[\mathrm{depth}(T^{(i)})+1]$.

The only potential difficulty is if either the depth of $T^{(i)}$ or the alphabet size m are so large that the computation of π becomes computationally expensive. In such cases, $H^{(i)}$ can be computed approximately by including an additional Monte Carlo step: Generate a sufficiently long random sample Y_{-D+1}^{M} from the chain $(T^{(i)}, \theta^{(i)})$, and calculate,

$$H^{(i)} \approx -\frac{1}{M} \log P(Y_1^M | Y_{-D+1}^0, T^{(i)}, \theta^{(i)}). \tag{4.3}$$

The ergodic theorem and the central limit theorem for Markov chains (Chung, 1967; Meyn and Tweedie, 2012) then guarantee the accuracy of (4.3).

The fact that we can easily obtain i.i.d. samples from the entropy-rate posterior is very important, because it means that we can effectively estimate the entire posterior distribution $\pi(\bar{H}|x)$, providing much richer information than point estimates. This is an important advantage of the Bayesian methodology followed here.

4.3 Theoretical results

In this section we give theoretical results for the asymptotic behaviour of the BCT entropy posterior $\pi(\bar{H}|x)$, providing some justifications for its use in estimation.

Our first theoretical result shows that the posterior of the entropy rate asymptotically concentrates on the true underlying value. Theorem 4.2 shows that it is asymptotically normal. In order to avoid inessential technicalities, we assume throughout that the data-generating process $\{X_n\}$ is a *positive ergodic* variable-memory chain with memory no greater than D, i.e., that all the parameters $\theta_s(j)$ are strictly positive.

Theorem 4.1. *Let X_{-D+1}^n be generated by a positive-ergodic, variable-memory chain $\{X_n\}$ with minimal model $T^* \in \mathcal{T}(D)$, parameters θ^*, and entropy rate $H^* = H(T^*, \theta^*)$; let β be arbitrary. The posterior distribution of the entropy rate $\pi(\bar{H}|X_{-D+1}^n)$ concentrates around the true value H^*, i.e.,*

$$\pi(\cdot|X_{-D+1}^n) \xrightarrow{\mathcal{D}} \delta_{H^*}, \quad a.s., \text{ as } n \to \infty, \tag{4.4}$$

where $\xrightarrow{\mathcal{D}}$ denotes weak convergence of probability measures, and δ_{H^*} is the unit mass at H^*.

Proof. This is an immediate consequence of the concentration of the BCT joint posterior $\pi(T, \theta|X_{-D+1}^n)$ around the true values (T^*, θ^*), which was recently proven as Theorem 3.7 in Kontoyiannis (2023). This result for the joint posterior actually follows easily from the concentration of the model posterior $\pi(T|X_{-D+1}^n)$ established in Theorem 3.1. □

Theorem 4.2. *Under the assumptions of Theorem 4.1, let $H^{(n)}$ be distributed according to the posterior distribution $\pi(\bar{H}|X_{-D+1}^n)$, and let $\bar{\theta}_{T^*}^{(n)}$ denote the mean of the posterior of the parameters $\pi(\theta|T^*,X_{-D+1}^n)$. Then, as $n\to\infty$,*

$$\sqrt{n}\big(H^{(n)}-\widetilde{H}^{(n)}\big)\xrightarrow{\mathcal{D}} Z\sim\mathcal{N}(0,\sigma_H^2),\quad a.s., \tag{4.5}$$

where $\widetilde{H}^{(n)}=H(T^*,\bar{\theta}_{T^*}^{(n)})\to H^*$, a.s. as $n\to\infty$.

Proof. First, denoting $x=X_{-D+1}^n$ for simplicity, for the entropy rate posterior density $\pi(\bar{H}|x)$ we can write,

$$\pi(\bar{H}|x)=\sum_{T\in\mathcal{T}(D)}\pi(\bar{H}|T,x)\pi(T|x), \tag{4.6}$$

where for every given T, the conditional density $\pi(\bar{H}|T,x)$ arises from the conditional density of the parameters $\pi(\theta|T,x)$ through the transformation of variables $\bar{H}=H(T,\theta)$.

Denoting the posterior density $\pi_{\bar{H}|x}(h)$ from (4.6) as $f_{\bar{H},n}(h)$, in order to prove the theorem it suffices to show that,

$$\frac{1}{\sqrt{n}}f_{\bar{H},n}\left(\frac{h}{\sqrt{n}}+\widetilde{H}^{(n)}\right)\to\phi_{\sigma_H}(h),\ \text{ as } n\to\infty, \tag{4.7}$$

uniformly on compact sets in $\mathbb{R}$, where $\phi_\sigma(z)$ denotes the density of a zero mean Gaussian with variance σ^2.

From the central limit theorem (CLT) on the parameters, proven as Theorem 3.8 in Kontoyiannis (2023), we have that, for $\theta_{T^*}^{(n)}$ distributed according to the posterior $\pi(\theta|T^*,x)$, as $n\to\infty$,

$$\sqrt{n}\big(\theta_{T^*}^{(n)}-\bar{\theta}_{T^*}^{(n)}\big)\xrightarrow{\mathcal{D}} Z\sim\mathcal{N}(0,J),\quad a.s., \tag{4.8}$$

where the covariance matrix J is given in Kontoyiannis (2023). Similarly, we can get for every T that,

$$\sqrt{n}\big(\theta_{T}^{(n)}-\bar{\theta}_{T}^{(n)}\big)\xrightarrow{\mathcal{D}} Z\sim\mathcal{N}(0,J_T),\quad a.s., \tag{4.9}$$

where $\theta_T^{(n)}$ is distributed according to the posterior $\pi(\theta|T,x)$ with mean $\bar{\theta}_T^{(n)}$.

Now, given T, $\bar{H}$ is just a function of the parameters θ, so applying the delta method (Van der Vaart, 2000) we get that,

$$\sqrt{n}\big(H_{T}^{(n)}-\widetilde{H}_{T}^{(n)}\big)\xrightarrow{\mathcal{D}} Z\sim\mathcal{N}(0,\sigma_T^2),\quad a.s., \tag{4.10}$$

where $\widetilde{H}_T^{(n)}=H(T,\bar{\theta}_T^{(n)})$, and $H_T^{(n)}$ is distributed according to $\pi(\bar{H}|T,x)$.

We note that in order to use the delta method, implicitly we are using the differentiability of H, which reduces to the differentiability of the stationary distribution π as a function of θ; this condition is satisfied for a positive-ergodic chain, see, e.g. Funderlic and Meyer (1986); Golub and Meyer (1986); Meyer (1975); Schweitzer (1968) and the references therein for more details on that.

Considering the conditional density $f_{\bar{H}|T,n}$ of the random variables in (4.10), we get that, uniformly on compact sets,

$$\frac{1}{\sqrt{n}} f_{\bar{H}|T,n}\left(\frac{h}{\sqrt{n}} + \widetilde{H}_T^{(n)}\right) \to \phi_{\sigma_T}(h), \text{ as } n \to \infty. \tag{4.11}$$

Finally, from Theorem 3.1, the model posterior concentrates on T^*, i.e., $\pi(T^*|x) \to 1$ and $\pi(T|x) \to 0$ for $T \neq T^*$, a.s. as $n \to \infty$. Combining this with (4.11) and (4.6) gives (4.7) with $\sigma_H = \sigma_{T^*}$ and $\widetilde{H}^{(n)} = \widetilde{H}_{T^*}^{(n)}$, completing the proof. □

Remarks:

1. The delta method also gives an expression for the asymptotic variance, namely, $\sigma_H^2 = \nabla_\theta H^{\mathrm{T}} J \nabla_\theta H$. This involves the covariance matrix J (Kontoyiannis, 2023) and the partial derivatives $\partial H/\partial\theta$, which in turn involve the partial derivatives of the stationary distribution with respect to θ (Golub and Meyer, 1986; Schweitzer, 1968).

2. Theorem 4.2 shows that the posterior $\pi(\bar{H},x)$ is asymptotically normal around $\widetilde{H}^{(n)}$. A CLT also holds for the convergence of $\widetilde{H}^{(n)}$ to the true value H^*, so it is easy to see that $H^{(n)} = H^* + O_p(1/\sqrt{n})$.

Finally, we give some theoretical results for the naive CTW entropy estimator $\hat{H}_{\mathrm{CTW}}$ described in Section 4.1. Although this was used in Gao et al. (2008) only for binary data, here we prove its consistency and asymptotic normality for finite alphabets.

Theorem 4.3. *Under the assumptions of Theorem 4.1, the naive CTW entropy estimator is consistent, i.e.,*

$$\hat{H}_{\mathrm{CTW}} = -\frac{1}{n}\log P_D^*(x_1^n) \to H^*, \text{ a.s., as } n \to \infty, \tag{4.12}$$

where $P_D^*(x_1^n)$ is the prior predictive likelihood in (1.1).

Theorem 4.4. *Under the assumptions of Theorem 4.1, the naive CTW entropy estimator is asymptotically normal:*

$$\sqrt{n}\left(\hat{H}_{\mathrm{CTW}} - H^*\right) \to Z \sim \mathcal{N}(0, \sigma_{\mathrm{CTW}}^2), \text{ a.s., as } n \to \infty. \tag{4.13}$$

Proof. Denoting the likelihood of the data $P(x_1^n|T^*,\theta^*)$, both theorems follow immediately from the observation that,

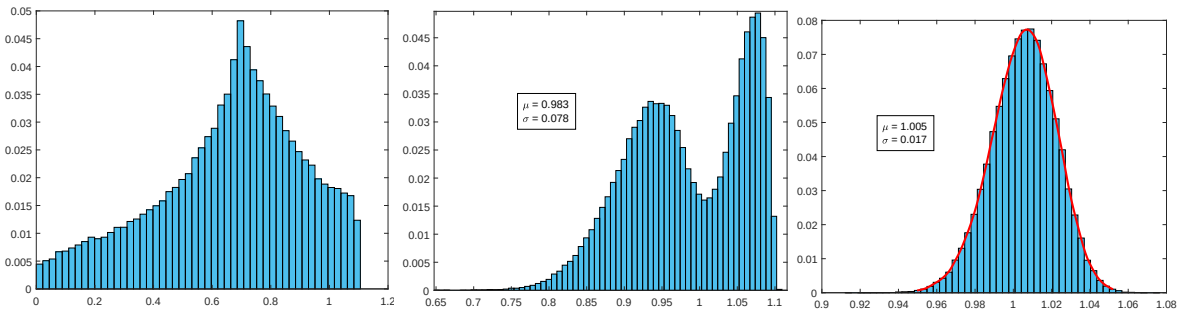
$$\log P_D^*(x_1^n) = \log P(x_1^n|T^*,\theta^*) + O(\log n), \text{ a.s.}, \tag{4.14}$$

which follows directly from the explicit upper and lower bounds on $P_D^*(x_1^n)$ given as Theorem 3.1 in Kontoyiannis (2023) and "Barron's lemma" in Kontoyiannis (1997). The ergodic theorem and the CLT for Markov chains Chung (1967); Meyn and Tweedie (2012) then directly give Theorems 4.3 and 4.4, respectively. Note that *positive* ergodicity is not strictly needed for these results, we just need the ergodic theorem and CLT to hold respectively; see Chung (1967); Meyn and Tweedie (2012) for general conditions. □

4.4 Experimental results

In this section, the BCT entropy estimator (with $D = 10$) is compared with state-of-the-art approaches as identified by Verdú (2019) and summarised in Section 4.1. The BCT estimator is found to give the most reliable estimates on a variety of simulated and real-world data. Throughout this section, all entropies are expressed in *nats*.

A ternary 5th order chain. We consider $n = 1,000$ observations generated from the 5th order, ternary chain of Section 2.4.1. The entropy rate of this chain is $\bar{H} = 1.02$. In Figure 4.1 we show estimates of the prior distribution $\pi(\bar{H})$, and of the posterior $\pi(\bar{H}|x)$ based on $n = 100$ and $n = 1,000$ observations. With $n = 1,000$, the posterior is close to a Gaussian with mean $\mu = 1.005$ and standard deviation $\sigma = 0.017$. For each histogram $N = 10^5$ i.i.d. samples were used, and in each case (and in all subsequent examples), the vertical axis of the histograms shows the frequency of the bins in the Monte Carlo sample.



(a) prior **(b)** $n = 100$ **(c)** $n = 1000$

Figure 4.1: Prior $\pi(\bar{H})$ and posterior $\pi(\bar{H}|x)$ of the entropy rate $\bar{H}$ with $n = 100$ and $n = 1,000$ observations x.

Figure 4.2 shows the performance of the BCT estimator compared with the other estimators described above, as a function of the length n of the available observations x. For BCT we plot the posterior mean. For the plug-in we plot estimates with block-lengths $k = 5, 6, 7$. It is easily observed that the BCT estimator outperforms all the alternatives, and converges faster and closer to the true value of $\bar{H}$.

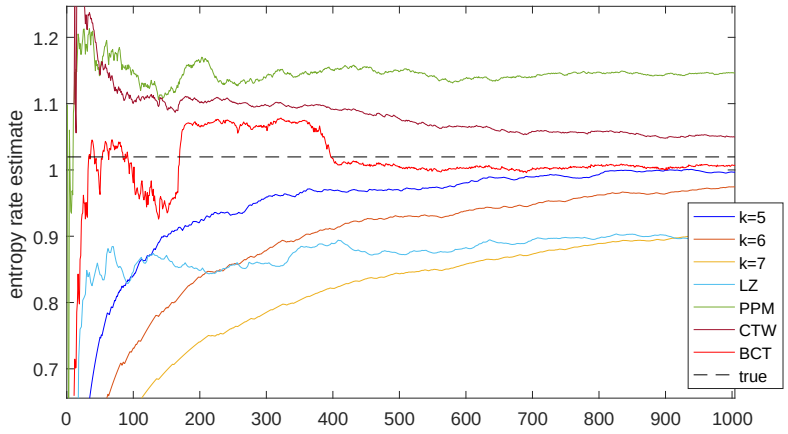


Figure 4.2: Entropy rate estimates for the ternary chain, as the number of observations increases.

A third order binary chain. We consider $n = 1,000$ observations generated from a third order binary chain from Berchtold and Raftery (2002). The underlying model is the complete binary tree of depth 3 pruned at node $s = 11$; the tree model T and parameters θ are given in Appendix C.1. The entropy rate of this chain is $\bar{H} = 0.4815$. Figure 4.3 shows the performance of all estimators, where BCT (using the posterior mean) is found to have the best performance. The histogram of the BCT posterior after $n = 1,000$ observations, is close to a Gaussian with mean $\mu = 0.4806$ and standard deviation $\sigma = 0.0405$.

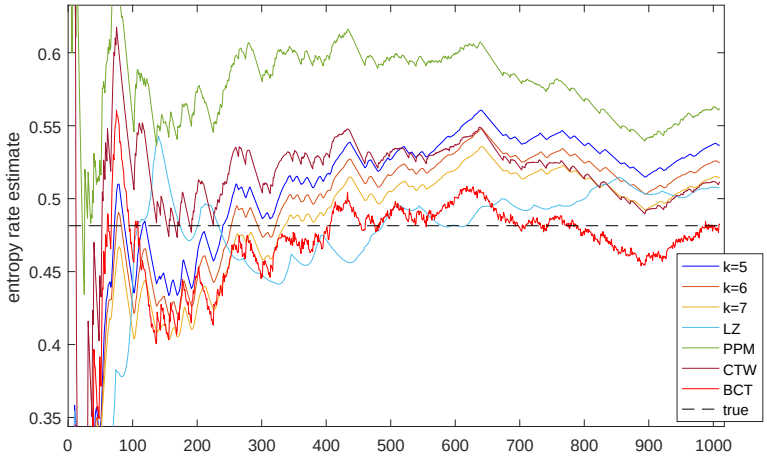
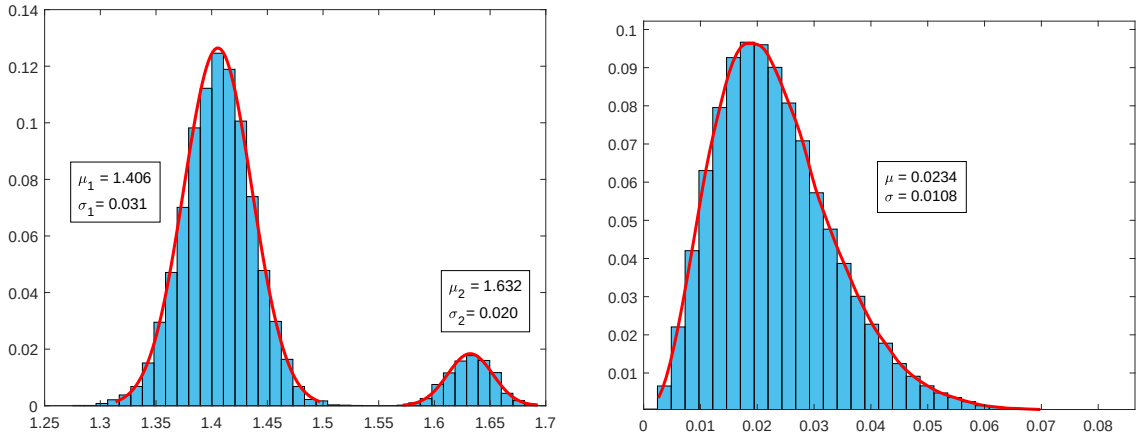


Figure 4.3: Entropy rate estimates for the binary chain, as the number of observations increases.

A bimodal posterior. We re-examine the "bimodal posterior" example (Example 2.2) from Section 2.5 for the purposes of entropy estimation; the true value of the entropy rate is $\bar{H} = 1.355$. Recalling our findings in Section 2.5, the model posterior was found to be bimodal, with one mode corresponding to the empty tree (describing i.i.d. observations) and the other consisting of tree models with depth 3.

As shown in Figure 4.4a, the posterior of the entropy rate is also bimodal here, with two approximately-Gaussian modes corresponding to the two model posterior modes. The dominant mode is the one corresponding to models of depth 3; it has mean $\mu_1 = 1.406$, standard deviation $\sigma_1 = 0.031$, and relative weight $w_1 = 0.91$. The second mode corresponding to the empty tree has mean $\mu_2 = 1.632$, standard deviation $\sigma_2 = 0.020$, and a much smaller weight $w_2 = 1 - w_1 = 0.09$. In this case, because of the bimodality of the posterior, the mode of $\pi(\bar{H}|x)$ gives a more reasonable point estimate than the posterior mean.



(a) bimodal example **(b)** spike train

Figure 4.4: Histograms of the posterior distribution $\pi(\bar{H}|x)$ of the entropy rate, constructed from $N = 10^5$ i.i.d. samples.

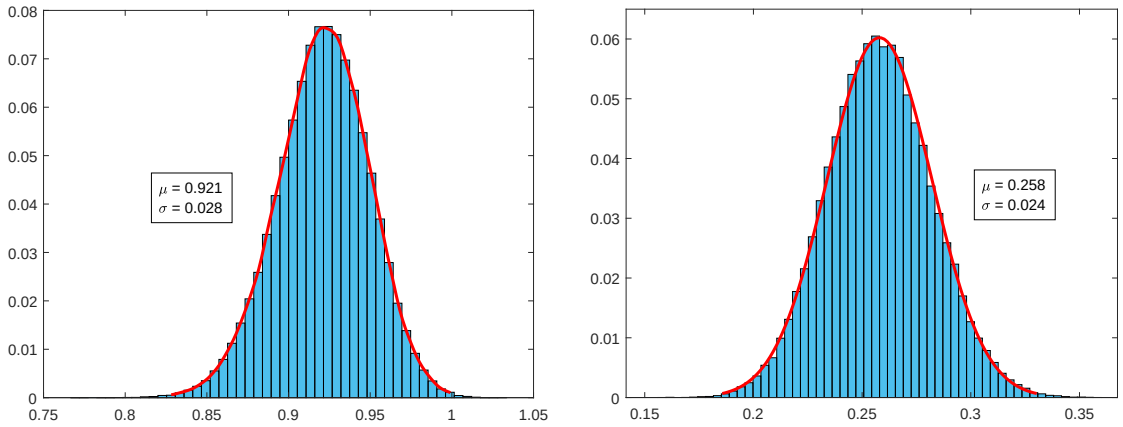
Neural spike train. We consider $n = 1{,}000$ binary observations from the spike train of Section 2.4.2. recorded from a single neuron in region V4 of a monkey's brain. The entropy-rate posterior is shown in Figure 4.4b: Its mean is $\mu = 0.0234$, its standard deviation is $\sigma = 0.0108$, and is slightly skewed to the right.

This dataset is the first part of a long spike train of length $n = 3{,}919{,}361$ bits from Section 2.4.2. Although there is no "true" value of the entropy rate here, for the purposes of comparison we use the estimate obtained by the naive CTW estimator (identified as the most effective method by Verdú (2019)) when all $n = 3{,}919{,}361$ samples are used, giving $\bar{H} = 0.0241$. The resulting estimates for all methods (with the posterior mean used for BCT) are summarised in Table 4.1, verifying again that BCT outperforms all the other methods. For the plug-in estimator, we show a number of different block-lengths k.

Table 4.1: Entropy rate estimates for the neural spike train.

	"True"	BCT	CTW	PPM	LZ	$k=2$	$k=5$	$k=10$	$k=15$
$\widehat{H}$	0.0241	**0.0234**	0.0249	0.0360	0.0559	0.0204	0.0204	0.0198	0.0187

Financial data. Here, we consider $n = 2,000$ observations from the financial dataset in Appendix C.1. This consists of tick-by-tick price changes of the Facebook stock price, quantised to three values: $x_i = 0$ if the price goes down, $x_i = 1$ if it stays the same, and $x_i = 2$ if it goes up. The BCT entropy-rate posterior is shown in Figure 4.5a: It has mean $\mu = 0.921$, and standard deviation $\sigma = 0.028$.



(a) financial dataset **(b)** pewee birdsong

Figure 4.5: Histograms of the posterior distribution $\pi(\bar{H}|x)$ of the entropy rate, constructed from $N = 10^5$ i.i.d. samples.

Once again, as the "true" value of the entropy rate we take the estimate produced by the naive CTW estimator on a longer sequence with $n = 10^4$ observations, giving $\bar{H} = 0.916$. The results of all five estimators are summarised in Table 4.2, where the BCT estimator is again found to outperform the alternatives (again using the posterior mean).

Table 4.2: Entropy rate estimates for the financial dataset.

	"True"	BCT	CTW	PPM	LZ	$k=5$	$k=6$	$k=7$	$k=10$
$\widehat{H}$	0.916	**0.921**	0.939	1.049	0.846	0.930	0.907	0.870	0.713

Pewee birdsong. The last dataset is the twilight song of the wood pewee bird from Section 2.4.2, which consists of $n = 1327$ observations from an alphabet of size $m = 3$. The BCT posterior is shown in Figure 4.5b: It is approximately Gaussian with mean $\mu = 0.258$ and standard deviation $\sigma = 0.024$.

The fact that the standard deviation is small suggests "confidence" in the resulting estimates, which is important because here (as in most real applications) there is no knowledge of a "true" underlying value. Table 4.3 shows all the resulting estimates; the posterior mean is shown for the BCT estimator.

Table 4.3: Entropy rate estimates for the birdsong dataset.

	BCT	CTW	PPM	LZ	$k=2$	$k=5$	$k=10$	$k=15$
$\widehat{H}$	0.258	0.278	0.318	0.275	0.776	0.467	0.336	0.272

Summary. The main conclusion from the results on the six datasets examined in this section is that the BCT entropy estimator gives the most accurate and reliable results among the five estimators considered, outperforming the state-of-the-art approaches. In addition to the fact that the BCT point estimates typically outperform those produced by other methods, the BCT estimator is accompanied by the entire posterior distribution $\pi(\bar{H}|x)$ of the entropy rate, induced by the observations x. As usual, this distribution can be used to quantify the uncertainty in estimating $\bar{H}$, and it contains significantly more information than simple point estimates and their associated confidence intervals.

In closing, it is noted that there exist interesting directions for further research, extending the methodology developed in this chapter. In particular, building on the proposed methodology for entropy estimation, fully-Bayesian approaches could be developed for estimating the mutual information and directed information rates between processes. As measures of the mutual dependence and the flow of information between stochastic processes, these quantities arise in a number of important applications, as for example in testing for causality (Kontoyiannis and Skoularidou, 2016; Massey, 1990).

Chapter 5

Bayesian Change-Point Detection for Discrete Data

The challenging tasks of *segmentation* and *change-point* detection arise in numerous applications with time series data, in various different settings; see, e.g., the recent reviews by Aminikhanghahi and Cook (2017); Truong et al. (2020); Van den Burg and Williams (2020). In the case of discrete-valued data, important application areas include for example genetics, biomedicine, neuroscience, finance and the social sciences (Chandola et al., 2012). The perhaps most critical application in this setting is the segmentation of genetic data, for which the most commonly used tools are based on Hidden Markov Models. Since their introduction for modelling heterogeneous DNA sequences by Churchill (1989, 1992), HMMs have become quite popular for segmentation tasks in a wide range of disciplines (Kehagias, 2004). More recently, Bayesian HMM approaches have also been proposed and used in practice (Boys and Henderson, 2004; Totterdell et al., 2017).

Motivated in part by the demonstrated efficacy of our methods in a number of statistical tasks, in this chapter we build on the BCT framework and develop a Bayesian modelling framework for *piecewise homogeneous* variable-memory chains. An appropriate prior is placed on the number and the locations of the change-points, and each segment is described by a BCT model as before. A collection of MCMC samplers are developed for inference in this setting, allowing to sample directly from the desired posterior distribution of the number and the location of the change-points, and hence to perform effective Bayesian change-point detection. The performance of our methods is illustrated in a number of simulated experiments and real-world applications.

5.1 Piecewise homogeneous BCT models

In this section, we describe the proposed Bayesian modelling framework and the associated inference methodology for discrete-valued time series with change-points.

Each segment is modelled by a homogeneous variable-memory chain as in Chapter 2, and two different cases are considered: When the number of change-points is known *a priori*, and when it is unknown and needs to be inferred as well. Let ℓ denote the number of change-points, and let $1 = p_0 < p_1 < p_2 < \cdots < p_l < p_{\ell+1} = n$ denote their locations, where we include the end-points $p_0 = 1$ and $p_{\ell+1} = n$ for convenience, and write, $\mathbf{p} = (p_0, p_1, \ldots, p_{\ell+1})$.

5.1.1 Known number of change-points

Model description. Suppose the maximum memory length D is fixed. Given a time series x_{-D+1}^n, the number of change-points ℓ, and their locations $\mathbf{p} = (p_0, p_1, \ldots, p_{\ell+1})$, the observations x_1^n are partitioned into $(\ell+1)$ segments,

$$\begin{aligned} x(1;\mathbf{p}) &= x_1^{p_1-1}, \\ x(j;\mathbf{p}) &= x_{p_{j-1}}^{p_j-1}, \text{ for } j = 2,3,\ldots,\ell, \\ x(\ell+1;\mathbf{p}) &= x_{p_\ell}^n. \end{aligned}$$

Each segment $x(j;\mathbf{p})$, $1 \le j \le \ell+1$, is assumed to be distributed as a variable-memory chain with model $T^{(j)} \in \mathcal{T}(D)$, parameter vector $\theta^{(j)}$, and initial context given by the D symbols preceding $x_{p_{j-1}}$ (i.e., the last D symbols of the previous segment). The resulting likelihood is,

$$P(x|\mathbf{p}, \{\theta^{(j)}\}, \{T^{(j)}\}) = \prod_{j=1}^{\ell+1} P(x_{p_{j-1}}^{p_j-1} | x_{p_{j-1}-D}^{p_{j-1}-1}, \theta^{(j)}, T^{(j)}),$$

where each term in the product is given by the standard BCT likelihood as in (2.6).

Prior structure. Given the number ℓ of change-points, following Green (1995) and Fearnhead (2006), we place a prior on their locations $\mathbf{p}$ specified by the even-order statistics of $(2\ell+1)$ uniform draws from $\{2,3,\ldots,n-1\}$ without replacement,

$$\pi(\mathbf{p}|\ell) = K_\ell^{-1} \prod_{j=0}^{\ell} (p_{j+1} - p_j - 1), \tag{5.1}$$

where $K_\ell = \binom{n-2}{2\ell+1}$. This prior penalises short segments to avoid overfitting: The probability of $\mathbf{p}$ is proportional to the product of the lengths of the segments $x(j;\mathbf{p})$. For example, the prior gives zero probability to adjacent change-points, i.e., to segments of zero length.

Finally, given ℓ and $\mathbf{p}$, an independent BCT prior $\pi(T^{(j)})\pi(\theta^{(j)}|T^{(j)})$ is placed on the model and parameters of each segment j, as in (2.4) and (2.5).

Posterior distribution. The posterior distribution of the change-point locations $\mathbf{p}$ is,

$$\pi(\mathbf{p}|x) \propto P(x|\mathbf{p})\pi(\mathbf{p}|\ell),$$

with $\pi(\mathbf{p}|\ell)$ as in (5.1). To compute the term $P(x|\mathbf{p})$, all models $T^{(j)}$ and parameters $\theta^{(j)}$ in $P(x, \{\theta^{(j)}\}, \{T^{(j)}\}|\mathbf{p})$ need to be integrated out. Since independent priors are placed on different segments, $P(x|\mathbf{p})$ reduces to the product of the prior predictive likelihoods,

$$P(x|\mathbf{p}) = \prod_{j=1}^{\ell+1} P_D^*(x(j;\mathbf{p})), \tag{5.2}$$

where we suppress the dependence on the initial context for each segment to simplify notation.

Importantly, each term in the product (5.2) can be computed efficiently using the CTW algorithm as usual (Theorem 2.1). This means that the models and parameters in each segment can be marginalised out, making it possible to efficiently compute the unnormalised posterior $\pi(\mathbf{p}|x)$ for any $\mathbf{p}$. This is critical for effective inference on $\mathbf{p}$, as it means that there is no need to estimate or sample the variables $\{T^{(j)}\}$ and $\{\theta^{(j)}\}$ in order to sample directly from $\pi(\mathbf{p}|x)$, as described next.

MCMC sampler. The MCMC sampler for $\pi(\mathbf{p}|x)$ is a Metropolis-Hastings (MH) algorithm. Apart from the standard inputs of the time series $x = x_{-D+1}^n$, the alphabet size $m \geq 2$, the maximum memory length $D \geq 0$ and the prior hyperparameter β, this algorithm also requires the number of change-points $\ell \geq 1$ and an initial MCMC state $\mathbf{p}^{(0)}$to be specified. The proposal distribution is described below.

Given the current state $\mathbf{p}^{(t)} = \mathbf{p}$, a new state $\mathbf{p}'$ is proposed as follows: One of the change-points p_i is chosen at random from $\mathbf{p}$ (excluding the edges $p_0 = 1$ and $p_{\ell+1} = n$), and it is replaced either by a uniformly chosen position p_i' from the $(n-\ell-2)$ remaining available positions, with probability 1/2, or by one of its two neighbours, again with probability 1/2. [Note that the neighbours of a change-point are always available, as the prior places zero probability to adjacent change-points.]

The proposed $\mathbf{p}'$ is either accepted and $\mathbf{p}^{(t+1)} = \mathbf{p}'$, or rejected and $\mathbf{p}^{(t+1)} = \mathbf{p}$, with acceptance probability given by $\alpha(\mathbf{p}, \mathbf{p}') = \min\{1, r(\mathbf{p}, \mathbf{p}')\}$, where,

$$r(\mathbf{p}, \mathbf{p}') = \frac{P(x|\mathbf{p}')}{P(x|\mathbf{p})} \times \prod_{j=0}^{\ell} \frac{(p'_{j+1} - p'_j - 1)}{(p_{j+1} - p_j - 1)}.$$

Importantly, the fact that the terms $P(x|\mathbf{p})$ and $P(x|\mathbf{p}')$ can be computed exactly and efficiently using the CTW algorithm is what enables this sampler to work effectively.

5.1.2 Unknown number of change-points

In most applications the number of change-points ℓ is not known *a priori*, and needs to be treated as an additional parameter to be inferred. As before, given the number and location of the change-points, each segment is modelled by a variable-memory chain.

Prior structure. Given the maximum possible number of change-points $\ell_{\max} \geq 1$, we place a uniform prior for ℓ on $\{0, 1, \ldots, \ell_{\max}\}$, and, given ℓ, the rest of the priors on $\mathbf{p}$, $\{T^{(j)}\}$ and $\{\theta^{(j)}\}$ remain the same as in Section 5.1.1. The uniform prior is a common choice for ℓ in such problems, as the Bayesian approach implicitly penalises the more complex models by averaging over a larger number of parameters, resulting in what is sometimes referred to as "automatic Occam's Razor" (MacKay, 2003; Rasmussen and Ghahramani, 2000).

Posterior distribution. The joint posterior distribution on the number and the locations of the change-points is,

$$\pi(\mathbf{p}, \ell | x) \propto P(x|\mathbf{p}, \ell)\pi(\mathbf{p}|\ell)\pi(\ell), \tag{5.3}$$

where $\pi(\mathbf{p}|\ell)$ is given in (5.1), $\pi(\ell) = 1/(1+\ell_{\max})$, and the term $P(x|\mathbf{p}, \ell)$ is identical to the term $P(x|\mathbf{p})$ in (5.2).

MCMC sampler. The MCMC sampler for $\pi(\mathbf{p}, \ell|x)$ is again a Metropolis-Hastings algorithm. Given the current state $(\ell^{(t)}, \mathbf{p}^{(t)}) = (\ell, \mathbf{p})$, propose a new state $(\ell', \mathbf{p}')$ as follows:

(i) If $\ell = 0$, set $\ell' = 1$, choose p'_1 uniformly among the $n-2$ available positions, and form $\mathbf{p}' = (1, p'_1, n)$.

(ii) If $1 \leq \ell < \ell_{\max}$, then select one of the following three options with probability 1/3 each:

(a) Set $\ell' = \ell - 1$ and form $\mathbf{p}'$ by deleting a uniformly chosen change-point from $\mathbf{p}^{(t)}$;

(b) Set $\ell' = \ell + 1$, choose a new change-point uniformly from the $(n - \ell - 2)$ available positions, and let $\mathbf{p}'$ be the same as $\mathbf{p}^{(t)}$ with the new point added;

(c) Set $\ell' = \ell$ and propose $\mathbf{p}'$ as in the sampler of Section 5.1.1.

(iii) If $\ell = \ell_{\max}$, then select one of the following two options with probability $1/2$ each:

(a) Set $\ell' = \ell - 1$ and form $\mathbf{p}'$ by deleting a uniformly chosen change-point from $\mathbf{p}^{(t)}$.

(b) Set $\ell' = \ell$ and propose $\mathbf{p}'$ as in the sampler of Section 5.1.1.

Finally, decide to accept the proposed $(\ell', \mathbf{p}')$ and set $(\ell^{(t+1)}, \mathbf{p}^{(t+1)}) = (\ell', \mathbf{p}')$ with acceptance probability $\alpha((\ell, \mathbf{p}), (\ell', \mathbf{p}')) = \min\{1, r((\ell, \mathbf{p}), (\ell', \mathbf{p}'))\}$, where in view of (5.3) the ratio $r((\ell, \mathbf{p}), (\ell', \mathbf{p}'))$ can again be computed easily via the CTW algorithm. The exact form of $r((\ell, \mathbf{p}), (\ell', \mathbf{p}'))$ follows from standard MH methodology and is given in Appendix B.4.

5.1.3 Further connections and comments

Unlike many of the relevant existing methods, which mainly only obtain point estimates and may rely in part on *ad hoc* considerations, the present methodology comes from a principled Bayesian approach that provides access to the entire posterior distribution of the parameters of interest, i.e., the number and the locations of the change-points. As usual, this offers a natural quantitative measure of uncertainty in the resulting estimates, and provides much richer information than simple point estimates.

The most closely related prior work is that of Gwadera et al. (2008), where VLMC models are used in conjunction with the BIC criterion or with a variant of the Minimum Description Length (MDL) principle to estimate variable-memory models and perform segmentation by solving an associated Bellman equation (Bellman, 1966).

There has also been a long line of works on change-point detection in the information-theoretic literature. There, piecewise homogeneous models were first considered by Merhav (1993) and Merhav and Feder (1995), who determined the optimal cumulative log-loss (or 'redundancy', in the language of data compression). Starting with Willems (1996), a series of papers examined sequential change-point detection, typically (but not exclusively) for independent and piecewise identically distributed models. These works (Shamir, 2003; Shamir and Costello, 2000, 2001; Shamir and Merhav, 1999) are primarily concerned with deriving theoretical bounds on the best achievable performance by on-line methods, and also propose sequential algorithms for change-point detection, mostly for piecewise i.i.d. (independent and identically distributed) data.

In a related but different direction, Jacob and Bansal (2008) and subsequently Juvvadi and Bansal (2013); Verma and Bansal (2019); Yamanishi and Fukushima (2018), frame the change-point detection problem as a series of hypothesis tests, performed using statistics based on data compression or entropy estimation algorithms. And an approach combining HMMs with information-theoretic ideas is adopted in Koolen and de Rooij (2013), where prediction is performed on piecewise stationary data.

Finally, in the context of generalisations of the original CTW framework, change-point models have also been examined by Veness et al. (2012) and Veness et al. (2013), and more recently by Shimada et al. (2021), who consider optimal and practical 'Bayes codes' for data compression in connection with these models.

5.2 Experimental results

In this section, we illustrate the performance of the proposed BCT-based methods for segmentation and change-point detection in a number of simulated experiments and real-world applications from genetics and meteorology.

5.2.1 Known number of change-points

The Simian virus 40 (SV40) is one of the most intensely studied animal viruses. Its genome is a circular double-stranded DNA molecule of 5,243 base-pairs (Reddy et al., 1978), available as sequence NC_001669.1 at the GenBank database (Clark et al., 2016).

The expression of SV40 genes is regulated by two major transcripts (early and late), suggesting the presence of a single major change-point in the entire genome (Churchill, 1992; Rotondo et al., 2019; Totterdell et al., 2017). One transcript is responsible for producing structural virus proteins while the other one is responsible for producing two T-antigens (a large and a small one). Hence, we take the start of the gene producing the large T-antigen to be the "true" change-point of the data set, corresponding to position $p_1^* = 2691$.

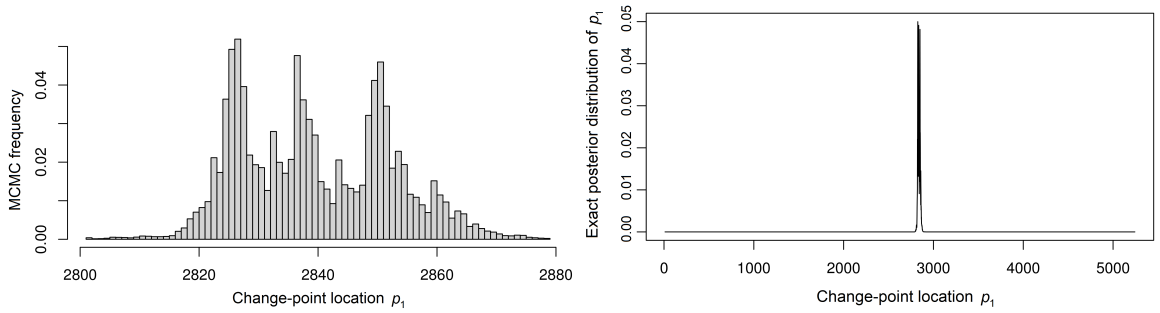


Figure 5.1: Posterior distribution $\pi(p_1|x)$ of the change-point location for the SV40 genome. Left: MCMC histogram of $\pi(p_1|x)$ near its mode, constructed from $N = 300,000$ MCMC samples, with the first 30,000 discarded as burn-in. Right: Entire exact posterior, $\pi(p_1|x)$.

The MCMC sampler of Section 5.1.1 was run with $\ell = 1$ and $D = 10$. The left plot in Figure 5.1 shows the resulting histogram of the change-point p_1, between locations 2800 and 2880; there were essentially no MCMC samples outside that interval. Viewing this as an approximation to the posterior $\pi(p_1|x)$, the approximate MAP location $\widehat{p}_1 = 2827$ is obtained, which is indeed close to $p_1^* = 2691$.

This result is particularly encouraging, especially since our method is a general method that is agnostic with respect to the nature of the observations – unlike some earlier approaches, it does not utilise any biological information about the structure of the data. Moreover, the value $\widehat{p}_1$ is slightly closer to p_1^* than estimates produced by some of the standard change-point detection methods; e.g., the HMM-based technique in Churchill (1992) gives $\widehat{p}_1 \approx 2859$, and the Bayesian HMM approach of Totterdell et al. (2017) gives $\widehat{p}_1 \approx 2854$.

Since this data set is relatively short and contains a single change-point, it is actually possible to compute the *entire* posterior distribution of the location p_1 exactly:

$$\pi(p_1|x) = \frac{P(x|p_1)\pi(p_1|\ell = 1)}{\sum_{p=2}^{n-1} P(x|p)\pi(p|\ell = 1)},$$

where $\pi(p_1|\ell = 1)$ is given by (5.1), and the terms $P(x|p_1)$ and $P(x|p)$ in the numerator and denominator can be computed using the CTW algorithm.

The resulting exact posterior distribution is shown in the right plot in Figure 5.1 and in more detail in Figure 5.2. Even though the shape of the posterior around the mode is quite irregular, the MCMC estimate is almost identical to the true distribution. This indicates that the MCMC sampler converges quite fast and explores all of the support of $\pi(p_1|x)$ effectively: The uniform jumps identify the approximate position of the change-point, and the random-walk moves explore the high-probability region around it.

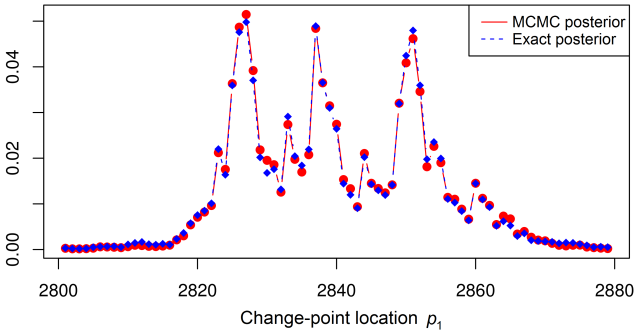


Figure 5.2: Exact distribution vs. MCMC estimate of $\pi(p_1|x)$ for the SV40 genome.

Note that the posterior distribution does not just provide a point estimate for p_1, but it identifies an interval in which it likely lies with high probability, illustrating a common advantage of the Bayesian approach. Also, we should point out that the fact that the posterior here can be computed without resorting to MCMC does not simply indicate that MCMC sampling is sometimes unnecessary. It actually highlights the power of the marginalisation provided by the CTW algorithm in that, for relatively short data sets with few change-points, it is possible to get access to the entire *exact* posterior distribution of interest.

5.2.2 Unknown number of change-points

Simulated data. We first examine a synthetic dataset that consists of $n = 4,300$ observations generated from four variable-memory chains with values in $A = \{0,1,2\}$. Their models (shown in Figure 5.3) and associated parameters (given in Appendix C.2) are chosen to be quite similar, so that the resulting segmentation problem is nontrivial. The locations of the three change-points are at positions $p_1^* = 2500$, $p_2^* = 3500$, $p_3^* = 4000$.

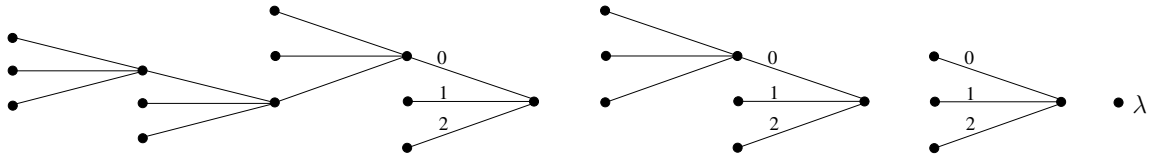


Figure 5.3: The four tree models used for generating the synthetic data set. The last model is the empty tree consisting of only the root node λ, corresponding to independent data.

The MCMC sampler of Section 5.1.2 was run with $\ell_{\max} = 10$ and $D = 10$. The resulting histogram approximation to the posterior of the number of change-points (left plot in Figure 5.4) shows that the algorithm identifies the correct value $\ell = 3$ with overwhelming confidence. Similarly, the posterior of the change-point locations (right plot in Figure 5.4) consists of three narrow peaks centered at the true locations. The actual MAP estimates for the change-point locations are $\widehat{p}_1 = 2497$, $\widehat{p}_2 = 3500$, and $\widehat{p}_3 = 3999$. Zoomed-in versions, showing the posterior of the location of each of the three change-points separately are shown in Figure C.9 in Appendix C.2.

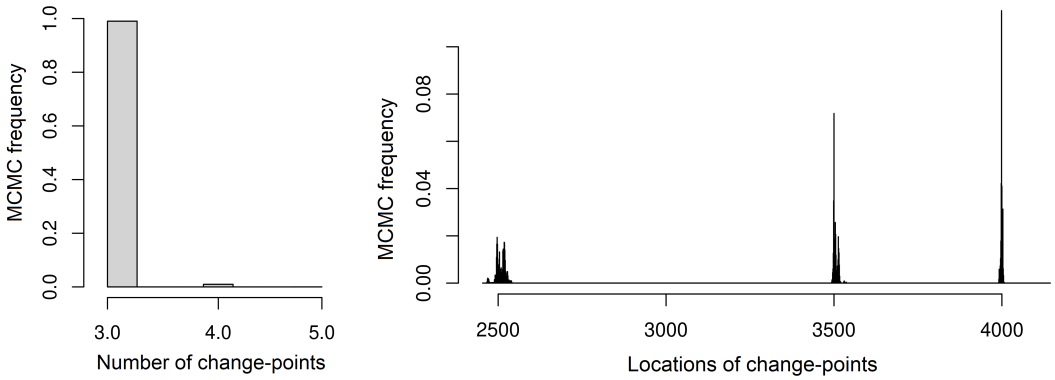


Figure 5.4: Simulated data. Left: MCMC histogram of the posterior of the number of change-points. Right: MCMC histogram of the posterior of the change-point locations. In both cases, $N = 10^5$ MCMC samples were used, with the first 10,000 discarded as burn-in.

Bacteriophage lambda genome. Here we study the 48,502 base-pair-long genome of the bacteriophage lambda virus (Sanger et al., 1982), which is available as sequence NC_001416.1 at the GenBank database. This dataset is often used as a benchmark for comparing different segmentation algorithms (Boys and Henderson, 2004; Braun and Müller, 1998; Braun et al., 2000; Churchill, 1989, 1992; Gwadera et al., 2008; Li, 2001). Due to the high complexity of this genome (which consists of 73 different genes), previous approaches often give slightly different results on both the number and locations of change-points, although all of them have been found to be biologically reasonable.

In particular, Gwadera et al. (2008) identify a total of 4 change-points, while Li (2001) and Boys and Henderson (2004) identify 5 change-points, Churchill (1992) and Braun and Müller

(1998) identify 6 change-points, and Braun et al. (2000) identify 8 change-points. These differences make it difficult to compare the performance of different methods quantitatively, but judging the biological relevance of the identified segments is still crucial. In order to do so, we will refer to the findings in the standard work of Liu et al. (2013).

The MCMC sampler of Section 5.1.2 was run with $D = 10$ and $\ell_{\max} = 10$. The resulting MCMC histogram approximation of the posterior over the number of change-points, shown in Figure 5.5, suggests that, with very high probability, there are either $\ell = 4$ or $\ell = 5$ change-points, with $\ell = 4$ being over seven times more likely than $\ell = 5$.

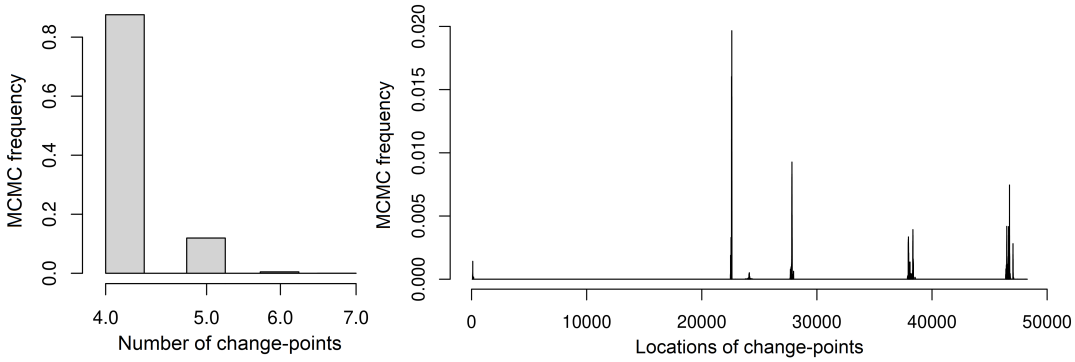


Figure 5.5: Bacteriophage lambda genome. Left: MCMC histogram of the posterior of the number of change-points. Right: MCMC histogram of the posterior of the change-point locations. In both cases, $N = 700,000$ samples were used, with the first 70,000 discarded as burn-in.

The posterior of the change-point locations, shown in Figure 5.5, clearly identifies four significant locations, as well as two more with significantly smaller weights (around positions 100 and 24,000). Zoomed-in versions, showing the posterior of each of the four main locations are given in Figure C.10 in Appendix C.2. The resulting MAP estimates for the change-point locations are shown in Table 5.1, where they are also compared with the biologically "true" change-points (Liu et al., 2013) and the estimates provided by one of the most successful previous approaches (Braun et al., 2000).

Table 5.1: Estimates of change-point locations in the bacteriophage lambda genome.

Gene	True	BCT	Braun et al. (2000)
ea47	22,686	22,607	22,544
ea59	26,973	27,832	27,829
cro	38,315	38,340	38,029
bor	46,752	46,731	46,528

According to Liu et al. (2013), the segments we identify correspond to a biologically meaningful and important partition in terms of gene expression. The first segment (1-22,607)

starts at the beginning of the genome and ends very close to the start of gene "ea47" (position 22,686), which signifies the end of the "late operon" and the beginning of the leftward "early operon", both of which play an important role in transcription. The second segment (22,608-27,832) essentially consists of the region "b2", an important region containing the three well-recognised "early" genes: "ea47", "ea31" and "ea59". The third segment (27,833-38,340) ends very close to the end of gene "cro" (position 38,315), which is the start of the rightward "early operon" that is also essential in transcription. Lastly, the fourth segment (38,341-46,731) ends very close to the end of gene "bor" (position 46,752), one of the major genes being translated.

Compared with previous findings, our results are similar enough to be plausible, while the places where they differ are precisely in the identification of biologically meaningful features, potentially improving performance. Specifically, our estimates are very close to those obtained by Gwadera et al. (2008), where an approach based on variable-memory chains is also used: Four change-points are identified by Gwadera et al. (2008), and their estimated locations lie inside the credible regions of our corresponding posteriors.

Comparing with the Bayesian HMM approach of Boys and Henderson (2004), the present method gives similar change-point locations, while also addressing some of its identified limitations: Firstly, Gupta and Liu (2004) argue that the assumption that all segments share the same memory length, which is adopted in the Bayesian HMM approach, may be problematic. Our methodology requires no such assumptions and, indeed, the MAP tree models obtained by the BCT algorithm in each segment have maximum depths $d = 5, 1, 2, 3$ and 0, respectively, indicating that the memory length is not constant throughout the genome. Also, it was seen that the Bayesian HMM framework may be sensitive to the assumed prior distribution on the state transition parameters. In contrast, the proposed framework avoids such problems as it uses a simple default value for the only prior hyperparameter, β.

El Niño occurrences. El Niño (Trenberth, 1997) is one of the most influential natural climate patterns on earth. It impacts ocean temperatures, the strength of ocean currents, and the local weather in South America. As a result, it has direct societal consequences on areas including the economy (Cashin et al., 2017) and public health (Kovats et al., 2003). Moreover, studying the frequency change of El Niño events can shed light on anthropogenic warming (Cai et al., 2014; Timmermann et al., 1999; Wang et al., 2019).

The data set considered here is a binary time series that consists of $n = 495$ annual observations between 1525 to 2020 (Quinn et al., 1987), with 0 representing the absence of an El Niño event and 1 indicating its presence; data for recent years are also available online through the US Climate Prediction Center, at: https://origin.cpc.ncep.noaa.gov/products/analysis_monitoring/ensostuff/ONI_v5.php.

The MCMC sampler of Section 5.1.2 was run with $D = 5$ and $\ell_{\max} = 5$. The resulting MCMC estimate of the posterior on the number of change-points, shown in Figure 5.6, suggests that the most likely value is $\ell = 2$, with $\ell = 1$ being a close second. The posterior of the change-point locations also shows two clear peaks; zoomed-in versions showing the location of each change-point separately are given in Figure C.11 in Appendix C.2.

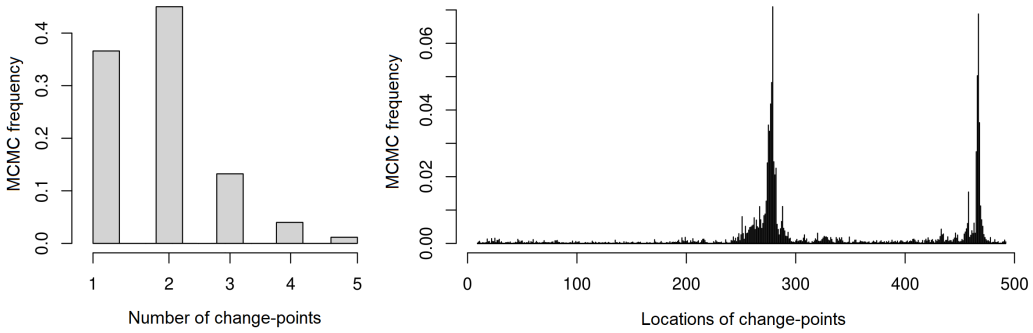


Figure 5.6: El Niño occurrences. Left: MCMC histogram of the posterior of the number of change-points. Right: MCMC histogram of the posterior of the change-point locations. In both cases, $N = 20,000$ MCMC samples were used, with the first 2,000 discarded as burn-in.

The MAP estimates of the two locations are at $\widehat{p}_1 = 278$ and $\widehat{p}_2 = 467$, corresponding to historically meaningful events during the years 1802 and 1991, respectively: The first change-point can be attributed to advancements in recording meteorological events, as prior to 1800 only the strong and extreme events were recorded (Quinn et al., 1987). The second change-point in the early 1990s likely indicates a response to greenhouse warming, which is expected to increase both the frequency and the intensity of El Niño events (Cai et al., 2014; Timmermann et al., 1999). Indeed, examining the first-order marginal $\pi = (\pi(0), \pi(1))$ of the stationary distribution associated with the MAP tree model in each segment, we find that the frequency $\pi(1)$ of the recorded El Niño events increases between consecutive segments, from $\pi(1) = 0.14$, to $\pi(1) = 0.35$, and finally to $\pi(1) = 0.54$.

Summary. Although change-point detection is a challenging task, the proposed methodology offers a general Bayesian approach that was found to perform well in various different settings, without requiring any preliminary information on the nature of the data. The efficacy of our methods was illustrated on simulated experiments, where the change-points were identified with high accuracy, and in real-world applications, where the resulting partitions were found to be meaningful; e.g., in genomic data, segments were found to coincide with important genes. Finally, by providing access to the entire posterior distribution of the change-points, the proposed Bayesian approach also offers a natural measure of uncertainty in the results, providing much more information than simple point estimates.

Chapter 6

The Bayesian Context Trees State Space Model

Up to this point, we have only considered discrete-valued time series, taking values in a finite alphabet. Here, we consider the more general setting of real-valued time series $\{X_n\}$ taking values in $\mathbb{R}$, hence broadening the scope of application of our methods by a lot.

As described in the Introduction, the goal of this chapter will be to develop a very general Bayesian modelling framework for building mixture models, that can be used with *any* existing model class as a base model, to produce flexible and *interpretable* mixture models. We do so by associating arbitrary real-valued time series models (like the AR model) at the leaves of context trees: A different time series model is associated to each leaf of the tree, hence constructing a mixture model. In this setting, the leaves of the tree play the role of discrete observable *states*, which admit natural and meaningful interpretations, and each one of them is associated to a different component model.

For this reason, we refer to the resulting model class of general mixture models as the *Bayesian Context Trees State Space Model* (BCT-SSM). Importantly, by appropriately extending the Bayesian modelling framework and the inference algorithms developed in Chapter 2 for the case of discrete-valued time series, we show that it is still possible to perform effective Bayesian inference in this much more general setting.

To illustrate the utility of the general framework, we study in detail the cases where AR and ARCH models are used as the base model placed at the leaves of the context trees. In both cases, the resulting mixture models are found to be flexible nonlinear models of high practical interest. In particular, the latter is found to give a systematic way of modelling the well-known asymmetric response in volatility due to positive and negative shocks, which is an important feature of financial time series. The proposed methods are found to outperform several state-of-the-art approaches in simulated experiments and real-world applications.

6.1 The general BCT-SSM model class

Discrete contexts. A key element of our development is the use of an *observable* state for each sample x_n, based on discretised versions of some of the samples $(\ldots, x_{n-2}, x_{n-1})$ preceding it. We refer to the string consisting of these discretised previous samples as the *discrete context*; it plays the role of a discrete-valued feature vector that can be used to identify additional useful structure in the data.

In order to extract these contexts, we consider simple piecewise constant quantisers from $\mathbb{R}$ to a finite alphabet $A = \{0, 1, \ldots, m-1\}$, of the form,

$$Q(x) = \begin{cases} 0, & x < c_1, \\ i, & c_i \leq x \leq c_{i+1},\ 1 \leq i \leq m-2, \\ m-1, & x > c_{m-1}, \end{cases} \tag{6.1}$$

where, throughout this section, the thresholds $\{c_1, \ldots, c_{m-1}\}$ and the resulting quantiser Q are considered fixed. A systematic Bayesian methodology to infer the thresholds from data is described in Section 6.2.3.

We note that this general framework can be used in conjunction with an arbitrary way of extracting discrete features, based on an arbitrary mapping to a discrete alphabet, not necessarily of the form in (6.1). However, the quantisation needs to be meaningful in order to lead to useful results. Quantisers as in (6.1) offer a generally reasonable choice although, depending on the application at hand, there are other useful approaches, e.g., quantising percentage differences between successive samples.

6.1.1 Model description

Given a quantiser Q with m levels as above, a maximum context length $D \geq 0$, and a proper m-ary context tree T, the context (or 'state') of each sample x_n is obtained as follows. Let $t = (Q(x_{n-1}), \ldots, Q(x_{n-D}))$ be the discretised string of length D preceding x_n; the *context* s of x_n is the unique leaf of T that is a suffix of t. For example, for the context tree of Figure 6.1, if $Q(x_{n-1}) = 0$ and $Q(x_{n-2}) = 1$ then $s = 01$, whereas if $Q(x_{n-1}) = Q(x_{n-2}) = 1$ then $s = 1$. The leaves of the tree define the set of discrete states in our hierarchical model. So, for the example BCT-SSM of Figure 6.1, the set of states is $\mathcal{S} = \{1, 01, 00\}$.

Equivalently, this process can be viewed as defining a partition of $\mathbb{R}^2$ into three regions indexed by the contexts $\mathcal{S}$ in T. As before, the fact that a tree T is *proper* means that for any string $t = (Q(x_{n-1}), \ldots, Q(x_{n-D}))$ there is always a unique context s that is a leaf of T. In our new setting, this means that proper trees define proper partitions, so that the resulting state-space regions are disjoint and their union is the whole space $\mathbb{R}^D$.

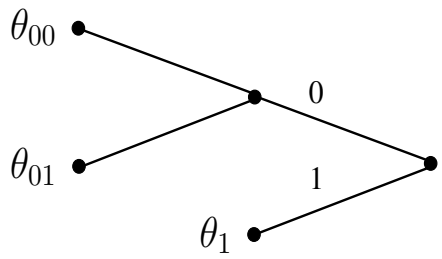


Figure 6.1: Example of a binary context tree T used for defining the set of discrete states.

To complete the specification of the BCT-SSM, we need to associate a different time series model $\mathcal{M}_s$ to each leaf s of the context tree T, giving a different conditional density for x_n: At time n, given the context s determined by the past D samples $(x_{n-1},\ldots,x_{n-D})$, the distribution of x_n is given by the model $\mathcal{M}_s$ assigned to s. Although general non-parametric models could also be used, for the rest of this chapter we consider parametric models with parameters θ_s at each leaf s. Altogether, the BCT-SSM consists of an m-ary quantiser Q, a proper m-ary tree T that defines the set of discrete states, and a collection of parameter vectors θ_s for the parametric models at the leaves of T.

Likelihood. Identifying T with the collection of its leaves $\mathcal{S}$ as before, the likelihood induced by the BCT-SSM can be written as,

$$p(x|T,\theta) := p(x_1^n|T,\theta,x_{-D+1}^0) = \prod_{i=1}^{n} p(x_i|T,\theta,x_{-D+1}^{i-1}) = \prod_{s\in T}\prod_{i\in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1}), \quad (6.2)$$

where B_s is the set of indices $i \in \{1,2,\ldots,n\}$ such that the context of x_i is s.

6.1.2 Bayesian modelling and inference

For the top level of the BCT-SSM, we consider collections of states represented by context trees T in the class $\mathcal{T}(D)$ of all proper m-ary trees with depth no greater than D.

Prior structure. For the trees $T \in \mathcal{T}(D)$ with maximum depth $D \geq 0$ at the top level of the hierarchical model, we use the BCT prior of (2.4),

$$\pi(T) = \pi_D(T;\beta) = \alpha^{|T|-1}\beta^{|T|-L_D(T)}.$$

This prior penalises larger trees by an exponential amount, which is a reasonable choice here, as it would be naturally desirable to penalise BCT-SSM models with large numbers of states and parameters, in order to to avoid overfitting. Given a tree model $T \in \mathcal{T}(D)$, we place an independent prior on each θ_s, so that $\pi(\theta|T) = \prod_{s\in T}\pi(\theta_s)$.

As described in the Introduction, the main obstacle in performing Bayesian inference is typically the computation of the normalising constant $p(x)$ of the posterior, which similarly with the discrete case in (1.1) can be written as,

$$p(x) = \sum_{T \in \mathcal{T}(D)} \pi(T) p(x|T) = \sum_{T \in \mathcal{T}(D)} \pi(T) \int_{\theta} p(x|T,\theta) \pi(\theta|T) \, d\theta, \tag{6.3}$$

where as a convention we now use lower-case $p(x)$ and $p(x|T,\theta)$ since we now have real-valued observations x and these quantities correspond to densities.

The power of the proposed Bayesian structure comes, in part, from the fact that, it is still possible to perform *exact* Bayesian inference efficiently, in this much more general setting. To that end, we introduce the Continuous Context Tree Weighting (CCTW) algorithm, and the Continuous Bayesian Context Tree (CBCT) algorithm, generalising the corresponding algorithms for discrete-valued time series from Chapter 2. It is shown that CCTW computes the normalising constant $p(x)$ exactly (Theorem 6.1), and CBCT identifies the MAP tree model (Theorem 6.2). In the Appendix we also discuss how the k-BCT algorithm is similarly modified to obtain the top-k *a posteriori* most likely trees, but the details are omitted here as this extension is completely analogous to the other two.

The main difference from the discrete case in Chapter 2, both in the algorithmic descriptions and in the proofs of the theorems (given in Appendix B.3), is that we introduce a new generalised form of *estimated probabilities* that is needed in place of their simple discrete versions in (2.7). These are given by,

$$P_e(s,x) = \int \prod_{i \in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1}) \; \pi(\theta_s) \, d\theta_s. \tag{6.4}$$

In the following, let $x = x_{-D+1}^{n}$ be a real-valued time series, and let $q_i = Q(x_i)$ denote the corresponding quantised samples.

CCTW: The Continuous Context Tree Weighting algorithm.

1. Build the tree T_{MAX}, whose leaves are all the discrete contexts q_{i-D}^{i-1}, $i = 1,2,\ldots,n$. Compute $P_e(s,x)$ as given in (6.4) for each node s of T_{MAX}.

2. Starting at the leaves and proceeding recursively towards the root compute:

$$P_{w,s} = \begin{cases} P_e(s,x), & \text{if } s \text{ is a leaf,} \\ \beta P_e(s,x) + (1-\beta) \prod_{j=0}^{m-1} P_{w,sj}, & \text{otherwise,} \end{cases}$$

   where sj is the concatenation of context s and symbol j.

3. Output the weighted probability $P_{w,\lambda}$ at the root λ.

CBCT: The Continuous Bayesian Context Tree algorithm.

1. Build the tree T_{MAX} and compute $P_e(s,x)$ for each node s of T_{MAX}, as in CCTW.
2. Starting at the leaves and proceeding recursively towards the root compute:

$$P_{m,s} = \begin{cases} P_e(s,x), & \text{if } s \text{ is a leaf at depth } D, \\ \beta, & \text{if } s \text{ is a leaf at depth } < D, \\ \max\left\{\beta P_e(s,x), (1-\beta)\prod_{j=0}^{m-1} P_{m,sj}\right\}, & \text{otherwise.} \end{cases}$$

3. Starting at the root and proceeding recursively with its descendants, for each node s: If the maximum above is achieved by the first term, prune all its descendants from T_{MAX}.
4. Output the resulting tree T_1^* and the maximal probability at the root $P_{m,\lambda}$.

Theorem 6.1. *The weighted probability $P_{w,\lambda}$ at the root λ computed by the CCTW algorithm is exactly the normalising constant $p(x)$ in* (6.3)*.*

Proof. The proof is given in Appendix B.3; it follows along the same lines as that of Theorem 2.1 for the discrete-valued case. □

Theorem 6.2. *For all $\beta \geq 1/2$, the tree T_1^* produced by the CBCT algorithm is the MAP tree model, satisfying,*

$$\pi(T_1^*|x) = \max_{T \in \mathcal{T}(D)} \pi(T|x).$$

Proof. The proof is given in Appendix B.3; it follows along the same lines as that of Theorem 2.2 for the discrete-valued case. □

Sampling from the posterior. Apart from the above algorithms – which together with k-BCT can be used to compute the evidence $p(x)$ and the top-k MAP tree models – it is also possible to obtain i.i.d. samples from the model posterior $\pi(T|x)$. This can be achieved by appropriately modifying the branching process representation of the model posterior $\pi(T|x)$ given in Section 3.1.2. Again, the difference in the branching process description is that we need to use the new version of the estimated probabilities $P_e(s,x)$ of (6.4) in the definition of the branching probabilities $P_{b,s}$ in (3.1).

Proposition 6.1. *The probability that the above branching procedure produces any particular tree $T \in \mathcal{T}(D)$ is given by $\pi(T|x)$.*

Proof. The proof is given in Appendix B.3; it follows along the same lines as that of Proposition 3.2 for the discrete-valued case. □

The methodology developed in this section is based on the introduction of the estimated probabilities of (6.4). In the next section, we illustrate the general principle via an interesting example where the estimated probabilities can be computed explicitly and the resulting mixture model is a flexible nonlinear model of practical interest. In specific, AR models $\mathcal{M}_s$ are associated to each context s, and we refer to the resulting model as the *Bayesian Context Trees Autoregressive* (BCT-AR) model, which is just a particular instance of the general BCT-SSM. It is noted that even in cases where the integrals in (6.4) are not tractable, the fact that they are in the form of standard marginal likelihoods makes it possible to compute them approximately by using standard methods, as e.g., Chib (1995); Chib and Jeliazkov (2001); Friel and Pettitt (2008); Kass and Raftery (1995); Wood (2011). The above algorithms can then be used with these approximations as a way of performing approximate inference for the BCT-SSM; this is further investigated in Section 6.3.

6.1.3 Further connections and comments

Context-trees for real-valued data. Since their introduction in the information-theoretic literature by Rissanen (1983a,b, 1986a), context trees have been employed very widely in connection with problems on *discrete* data, mostly in the context of data compression (Weinberger et al., 1994; Willems et al., 1995). In the previous chapters of this thesis, BCTs have been found to be very effective in the analysis of discrete-valued time series, in a range of important statistical tasks. The central conceptual novelty of this chapter is in showing that discrete context trees can in fact also be utilised in a very effective way for modelling *real-valued* time series, by representing meaningful context-based discrete states that are used to build flexible and interpretable mixture models of practical interest. In fact, this is remarkable and somewhat surprising, as at first glance one might think that context trees seem to be naturally suited only for discrete-valued data.

We introduce a new construction for building context-dependent mixtures of models in any existing model class, combining a discrete context tree with an arbitrary existing family of *continuous* models. An important step in the development of the BCT-SSM class is the introduction of a quantiser that extracts discrete contexts, along with a principled Bayesian procedure for actually selecting an appropriate quantiser in practice. Finally we are able to prove that, perhaps somewhat surprisingly, small modifications of the discrete BCT algorithms of Chapter 2 can be used for exact inference in the much more general BCT-SSM setting. In doing so, by introducing the Continuous CTW (CCTW) algorithm, we also give an answer to a long-standing discussion regarding whether it would be possible to extend the CTW algorithm to be useful in settings with real-valued observations.

Classification and regression trees. A possible alternative line of work would be to adapt classification and regression trees (CART) (Breiman et al., 1984) to the setting of time series. However, CARTs have been developed for regression and classification, where there is a set of independent input-output observations. So, in contrast with BCTs, these methods do not exploit the special sequential nature of time series data, where each observation is modelled as a function of the previous ones.

Starting with the algorithm of Breiman et al. (1984), CARTs either rely on greedy algorithms to a grow a tree with some stopping criteria – sometimes with additional penalties for pruning – or adopt a Bayesian CART approach and place a prior on trees (Chipman et al., 1998); see respectively Loh (2014) and Linero (2017) for reviews of these methods. The associated difficulties with selecting the tree are significant, mainly because greedily-constructed trees tend to overfit the data, while the proposed Bayesian approaches require the use of sophisticated MCMC to sample from the tree-posterior, something which is computationally expensive and not guaranteed to be effective in practice. These reasons partly explain why some previous approaches exploiting CARTs for time series (Audrino and Bühlmann, 2001; Dellaportas and Vrontos, 2007; Meek and Chickering, 2002) have not been used very widely, with the much simpler and more restricted class of threshold models (Tong, 2011; Zakoian, 1994) being overwhelmingly used in relevant applications where this kind of regime-switching is appropriate; see the following sections for more details on these.

So, instead of following the CART methodology here we employ the special sequential nature of time-series data and adopt the BCT point of view. In this way we are able to perform exact Bayesian inference for selecting the tree model. And actually, we are able to prove that our algorithms can identify exactly not only the MAP tree model, but also the top-k a posteriori most likely trees. Moreover, we are able to sample directly from the tree posterior, obtaining i.i.d. samples from it. These advantages make our approach much more promising than it would be to extend CART methods for time series.

Discrete patterns. Finally, we note that a number of earlier approaches employ discrete patterns in the analysis of real-valued time series (Alvarez et al., 2010; Alvisi et al., 2007; Berndt and Clifford, 1994; Fu et al., 2007; Hu et al., 2014; Liu et al., 2011; Ouyang et al., 2010; Sabeti et al., 2020). These works illustrate the fact that useful and meaningful information can indeed be extracted from discrete contexts. However, in most cases the methods are either application- or task-specific, and typically resort to *ad hoc* considerations for performing inference. In contrast, in this work discrete contexts are used in a natural manner, by defining a hierarchical Bayesian modelling structure upon which principled Bayesian inference is performed.

6.2 The BCT-AR model

Here we consider the BCT-SSM model class where an AR model of order p is associated to each leaf s of a context tree T, as,

$$x_n = \phi_{s,1} x_{n-1} + \cdots + \phi_{s,p} x_{n-p} + e_n = \phi_s{}^{\mathrm{T}} \widetilde{\mathbf{x}}_{n-1} + e_n, \quad e_n \sim \mathcal{N}(0, \sigma_s^2) \,, \tag{6.5}$$

where $\phi_s = (\phi_{s,1}, \ldots, \phi_{s,p})^{\mathrm{T}}$ and $\widetilde{\mathbf{x}}_{n-1} = (x_{n-1}, \ldots, x_{n-p})^{\mathrm{T}}$.

6.2.1 Bayesian modelling and inference

The parameters of the model are the AR coefficients and the noise variance, so that here $\theta_s = (\phi_s, \sigma_s^2)$. We use an inverse-gamma prior for the noise variance, and a Gaussian prior for the AR coefficients, so that the joint prior on the parameters is $\pi(\theta_s) = \pi(\phi_s|\sigma_s^2)\pi(\sigma_s^2)$, with,

$$\pi(\sigma_s^2) = \text{Inv-Gamma}(\tau, \lambda), \qquad \pi(\phi_s|\sigma_s^2) = \mathcal{N}(\mu_o, \sigma_s^2 \Sigma_o), \tag{6.6}$$

where $(\tau, \lambda, \mu_o, \Sigma_o)$ are the prior hyperparameters. This prior specification allows the exact computation of the estimated probabilities of (6.4), and also gives closed-form posteriors for the AR coefficients and the noise variance. These are given in Lemmas 6.1 and 6.2 below.

Lemma 6.1. *For the BCT-AR model, the estimated probabilities $P_e(s,x)$ as in* (6.4) *are given by,*

$$P_e(s,x) = C_s^{-1} \, \frac{\Gamma(\tau + |B_s|/2) \;\; \lambda^{\tau}}{\Gamma(\tau) \;\; (\lambda + D_s/2)^{\tau + |B_s|/2}} \,, \tag{6.7}$$

where $|B_s|$ is the cardinality of the set B_s in (6.2)*, i.e., the number of observations with context s, and,*

$$C_s = \sqrt{(2\pi)^{|B_s|} det(I + \Sigma_o S_3)}, \tag{6.8}$$

$$D_s = s_1 + \mu_o^T \Sigma_o^{-1} \mu_o - (\mathbf{s}_2 + \Sigma_o^{-1}\mu_o)^T (S_3 + \Sigma_o^{-1})^{-1} (\mathbf{s}_2 + \Sigma_o^{-1}\mu_o), \tag{6.9}$$

with the sums $s_1, \mathbf{s}_2, S_3$ defined as:

$$s_1 = \sum_{i \in B_s} x_i^2 \quad , \quad \mathbf{s}_2 = \sum_{i \in B_s} x_i \, \widetilde{\mathbf{x}}_{i-1} \quad , \quad S_3 = \sum_{i \in B_s} \widetilde{\mathbf{x}}_{i-1} \widetilde{\mathbf{x}}_{i-1}^T. \tag{6.10}$$

Proof. The complete proof is given in Appendix B.3; it is mostly based on explicit computations together with the factorisation of the likelihood $p(x|T,\theta)$ using the sets B_s in (6.2). □

Lemma 6.2. *Given a tree model T, at each leaf s, the posterior distributions of the AR coefficients and the noise variance are given by,*

$$\pi(\sigma_s^2|T,x) = \text{ Inv-Gamma}\left(\tau + |B_s|/2, \lambda + D_s/2\right), \qquad \pi(\phi_s|T,x) = t_\nu(\mathbf{m}_s, P_s), \quad (6.11)$$

where t_ν denotes a multivariate t-distribution with ν degrees of freedom. Here, $\nu = 2\tau + |B_s|$, and,

$$\mathbf{m}_s = (S_3 + \Sigma_o^{-1})^{-1}(\mathbf{s}_2 + \Sigma_o^{-1}\mu_o), \quad P_s^{-1} = \frac{2\tau + |B_s|}{2\lambda + D_s}(S_3 + \Sigma_o^{-1}). \quad (6.12)$$

Proof. The proof follows along the same lines as that of Lemma 6.1; it is also given in Appendix B.3. □

Corollary 6.1. *The MAP estimators of ϕ_s and σ_s^2 are given, respectively, by,*

$$\widehat{\phi}_s^{\text{MAP}} = \mathbf{m}_s\,, \quad \widehat{\sigma_s^2}^{\text{MAP}} = (2\lambda + D_s)/(2\tau + |B_s| + 2). \quad (6.13)$$

6.2.2 Computational complexity and sequential updates

Here we briefly discuss the computational complexity of our inference algorithms in this setting. Similarly with Section 2.3.5, for each symbol x_i in a time series x_1^n, exactly $(D+1)$ nodes of T_{MAX} need to be updated, corresponding to its contexts of length $0, 1, \ldots, D$. For each one of these nodes, only the quantities $\{|B_s|, s_1, \mathbf{s}_2, S_3\}$ need to be updated, which can be done efficiently by just adding an extra term to each sum. Using Lemma 6.1, the estimated probabilities $P_e(s,x)$ can then be computed for all nodes of T_{MAX}, whose size is bounded as a function of n. Since the recursive step only performs operations on T_{MAX}, the complexity of all algorithms as a function of n is only $O(n)$: *linear* in the length of the time series. This clearly shows that the present methods are computationally very efficient and scale well with large numbers of observations.

Actually, it is easy to see that the exact complexity as a function of n is $O\left(nDp^2\right)$. This is remarkable, because it is just D times larger than $O\left(np^2\right)$, i.e., the complexity of fitting a single AR model using least squares. So, we can perform exact Bayesian inference in this much richer model class at essentially no additional computational cost compared to just fitting a single AR model to the data.

The above discussion also shows that, importantly, all our algorithms can still be updated *sequentially*. For example, when observing a new sample x_{n+1} after executing CCTW for x_1^n, only $D+1$ nodes need to be updated, and $P_e(s,x)$ and $P_{w,s}$ have to be re-computed *only* at these nodes. This takes $O(D)$ operations, i.e., $O(1)$ as a function of n. In particular, this implies that sequential prediction can be performed very efficiently.

Empirical running times for all forecasting experiments are reported in Appendix D.3, showing that our methods are much more efficient than essentially all the alternatives examined. The difference is in fact very large, especially when comparing with state-of-the-art machine learning (ML) models that require heavy training and do not allow for efficient sequential updates, giving empirical running times that are typically larger by several orders of magnitude than ours; see also Makridakis et al. (2018b) for a relevant review comparing the computational requirements of ML versus statistical techniques.

6.2.3 Choosing the hyperparameters, quantiser and AR order

As Lemma 6.2 indicates, the posterior distributions of ϕ_s and σ_s^2 are not very sensitive to the prior hyperparameters. In all the experimental results below we make the simple choice $\mu_o = 0$ and $\Sigma_o = I$ in the AR coefficients' prior. In view of (6.11), τ and λ should be chosen to be relatively small in order to minimise their effect on the posterior, while keeping the mode of the inverse-gamma prior, $\lambda/(\tau+1)$, reasonable; setting $\lambda = \tau = 1$ seems a reasonable default value when no additional prior information are given. For the context tree prior, we use the default value $\beta = 1 - 2^{-m+1}$, and a maximum depth $D = 10$. It is noted that throughout our experiments there is no hyperparameter tuning from data, we just use the default values as described here.

Finally, we introduce a principled Bayesian way for selecting the quantiser thresholds $\{c_i\}$ of (6.1) and the AR order p. Viewing them as extra parameters on an additional layer above everything else, we place uniform priors on $\{c_i\}$ and p, and perform standard Bayesian model selection (MacKay, 1992, 2003) to obtain their MAP values. The resulting posterior $p(\{c_i\}, p|x)$ is proportional to the *evidence* $p(x|\{c_i\}, p)$, which can be computed exactly using the CCTW algorithm (Theorem 6.1).

So, in order to select appropriate values, we specify a suitable range of possible $\{c_i\}$ and p, and select the ones with the higher evidence. For the AR order we take $1 \leq p \leq p_{\max}$ for an appropriate $p_{\max}$ (e.g., $p_{\max} = 5$ in our experiments), and for the $\{c_i\}$ we perform a grid search in a reasonable range (e.g., between the 10th and 90th percentiles of the data). In our forecasting experiments, we select the AR order and the quantiser thresholds using the above procedure at the end of the training set.

6.2.4 Comparison with other AR mixtures

In this section, we briefly compare the BCT-AR model class with other popular mixtures of AR models that have been used widely throughout the years. In particular, we show that the BCT-AR model is a generalisation of some of these model classes.

Threshold AR models. Threshold autoregressive (TAR) models were introduced by Tong and Lim (1980), and have been used extensively in the analysis of nonlinear time series; see, e.g., the reviews by Hansen (2011); Tong (2011) and the texts by Cryer and Chan (2008); Tong (1990); Tsay (2005). Although numerous different versions of TAR models have been employed (see, e.g., the discussion in Tong (2011)), the most commonly used one is the self-exciting threshold AR (SETAR) model, which considers partitions of the state space based on the quantised value of x_{n-d}, for some *delay* parameter $d > 0$. Clearly, the BCT-AR model class is much richer and more general.

Mixture AR models. The mixture autoregressive (MAR) models of Wong and Li (2000) consist of a simple linear mixture of K Gaussian AR components.

The class of MAR models was first introduced by Wong and Li (2000) as a generalisation of the Gaussian MTD (GMTD) model of Le et al. (1996), which is itself an extension of MTD models for real-valued observations. Another nonlinear extension of the GMTD model, using GPs, was recently developed in Heiner and Kottas (2022a). Extensions of MAR models include the conditional heteroscedastic (MAR-ARCH) model (Wong and Li, 2001b), the use of exogenous variables (Wong and Li, 2001a), and the use of the Student-t distribution to model heavy tails (Wong et al., 2009), but these models seem less relevant for the type of applications considered here, as their benefits are limited to examples of datasets possessing these specific characteristics (conditional heteroscedasticity, heavy tails, etc).

When the BCT-AR posterior essentially concentrates on K models, $T_1, \ldots, T_K$, (which was commonly observed in practice), the posterior predictive distribution can be written as,

$$p(x_{n+1}|x) = \sum_{k=1}^{K} \pi(T_k|x)\ p(x_{n+1}|T_k, x)\ ,$$

so that BCT-AR can be viewed as a generalised MAR model, with components corresponding to the AR models at the leaves of each T_k, and Bayesian weights automatically selected as $\pi(T_k|x)$. So, the BCT-AR model can be viewed as a *generalisation* of both MAR and SETAR.

Markov Switching AR models. The Markov switching autoregressive model (MSA) (Hamilton, 1989) is a simple Hidden Markov Model where the hidden state process is modelled as a discrete-valued first-order Markov chain (usually binary or ternary), and a different AR model is associated to each state. The main difficulty in using the MSA is in estimating the model, since the discrete states are no longer directly observable (Tsay, 2005); usually, the Expectation-Maximisation (EM) algorithm is employed for this task. This was verified in practice in our experiments, where the MSA gave much larger running times compared to all classical methods that do not involve neural networks (Appendix D.3).

6.2.5 Experimental results

In this section, we evaluate the performance of the BCT-AR model on simulated and real data from standard applications of nonlinear time series in economics and finance, with possibly limited training data. We compare with the most successful previous approaches for these types of applications, considering both classical and modern ML methods. Useful resources include the `R` package `forecast` (Hyndman and Khandakar, 2008) and the Python library 'GluonTS' (Alexandrov et al., 2020), containing implementations of state-of-the-art classical and ML methods, respectively. Here, we briefly discuss the methods used, and refer to the packages' documentation and Appendix D.1 for more details on the methods themselves and the training procedures carried out.

Among classical statistical approaches, we compare with ARIMA and Exponential smoothing state space (ETS) models (Hyndman et al., 2008), which are implemented as `auto.arima` and `ets` in `forecast`, and with SETAR, MAR and MSA models, using respectively the `R` packages `TSA` (Chan and Ripley, 2020), `mixAR` (Boshnakov and Ravagli, 2021) and `MSwM` (Sanchez-Espigares and Lopez-Moreno, 2021). Among ML-based techniques, we compare with the Neural Network AR (NNAR) model (implemented in `forecast`), and with the most-successful RNN-based approaches, DeepAR (Salinas et al., 2020) and N-BEATS (Oreshkin et al., 2019), which are both implemented in GluonTS.

Simulated data. We first perform a simulated experiment illustrating that our methods are consistent and effective with data generated by BCT-SSM models. The context-tree model used here is the tree of Figure 6.1, the threshold of the binary quantiser is $c = 0$, and the AR order is $p = 2$. The exact BCT-AR model is given by:

$$x_n = \begin{cases} 0.7\, x_{n-1} - 0.3\, x_{n-2} + e_n, & e_n \sim \mathcal{N}(0, 0.15), & \text{if } s = 1\text{: } x_{n-1} > 0, \\ -0.3\, x_{n-1} - 0.2\, x_{n-2} + e_n, & e_n \sim \mathcal{N}(0, 0.10), & \text{if } s = 01\text{: } x_{n-1} \leq 0,\ x_{n-2} > 0, \\ 0.5\, x_{n-1} + e_n, & e_n \sim \mathcal{N}(0, 0.05), & \text{if } s = 00\text{: } x_{n-1} \leq 0,\ x_{n-2} \leq 0. \end{cases}$$

We first examine the posterior over trees, $\pi(T|x)$. On a time series consisting of only $n = 100$ observations, the MAP tree identified by the CBCT algorithm is the empty tree corresponding to a single AR model, with posterior probability 99.9%. This means that the data do not provide sufficient evidence to support a more complex state space partition. With $n = 300$ observations, the MAP tree is now the true underlying model, with posterior probability 57%. And with $n = 500$ observations, the posterior of the true model is 99.9%. Therefore, the posterior indeed concentrates on the true model, indicating that the BCT-AR inferential framework can be very effective even with limited training data.

The complete BCT-AR model fitted from $n = 1,000$ observations, with its MAP estimated parameters, clearly indicates that all parameter estimates have essentially converged:

$$x_n = \begin{cases} 0.66\, x_{n-1} - 0.19\, x_{n-2} + e_n, & e_n \sim \mathcal{N}(0, 0.16), & \text{if } x_{n-1} > 0, \\ -0.39\, x_{n-1} - 0.27\, x_{n-2} + e_n, & e_n \sim \mathcal{N}(0, 0.12), & \text{if } x_{n-1} \leq 0,\ x_{n-2} > 0, \\ 0.45\, x_{n-1} - 0.03\, x_{n-2} + e_n, & e_n \sim \mathcal{N}(0, 0.058), & \text{if } x_{n-1} \leq 0,\ x_{n-2} \leq 0. \end{cases}$$

Finally, although maximising the evidence is a well-justified Bayesian practice (MacKay, 1992, 2003), in Appendix D.2 we also report values of the evidence $p(x|c, p)$, which is maximised at the true values of $c = 0$ and $p = 2$, verifying that our inferential procedure for choosing the quantiser and the AR order is also effective.

Forecasting. Here, we evaluate the performance of our methods in out-of-sample 1-step ahead forecasts, and compare with state-of-the-art approaches in three simulated and three real datasets. The complete descriptions of all datasets can be found in Appendix D.2. The first simulated dataset consists of $n = 600$ observations generated from the BCT-AR model used above, the second (`sim_2`) consists of $n = 500$ observations generated by a BCT-AR model with a ternary tree of depth 2 (shown in Appendix D.2), and the third dataset (`sim_3`) consists of $n = 200$ observations generated from a SETAR model of order $p = 5$ (also shown in Appendix D.2). In each case, the training set consists of the first 50% of the observations; also, we update all models at every time-step in the test set. For the BCT-AR model, the MAP tree with its MAP parameters is used at every time-step, which can be updated efficiently as described in Section 6.2.2.

Table 6.1: Mean squared error (MSE) of forecasts in simulated and real experiments.

	BCT-AR	ARIMA	ETS	NNAR	DeepAR	N-BEATS	MSA	SETAR	MAR
`sim_1`	**0.131**	0.150	0.178	0.143	0.148	0.232	0.142	0.141	0.151
`sim_2`	**0.035**	0.050	0.054	0.048	0.061	0.112	0.049	0.050	0.064
`sim_3`	**0.891**	1.556	1.614	1.287	1.573	2.081	1.495	0.951	1.543
`unemp`	**0.034**	0.040	0.042	0.036	0.036	0.054	**0.034**	0.038	0.037
`gnp`	**0.324**	0.364	0.378	0.393	0.473	0.490	0.353	0.394	0.384
`ibm`	78.02	82.90	77.52	78.90	**75.71**	77.90	81.68	81.07	77.02

From Table 6.1, it is observed that our methods outperform the alternatives in all simulated experiments, in every case achieving a mean squared error (MSE) that is lower by $7 - 37\%$ compared to the second-best method. As discussed in Section 6.2.2, the BCT-SSM methods also outperform the alternatives in terms of empirical running times, reported in Appendix D.3. Next, we evaluate the performance of the proposed methods in real-world applications from economics and finance.

US unemployment rate. An important application of SETAR models is in modelling the US unemployment rate (Hansen, 2011; Koop and Potter, 1999; Montgomery et al., 1998; Rothman, 1998; Tsay, 2005). As described by Montgomery et al. (1998) and Tsay (2005), the unemployment rate moves countercyclically with business cycles, and rises quickly but decays slowly, indicating nonlinear behaviour.

Here, we study the quarterly US unemployment rate in the time period from 1948 to 2019 (dataset `unemp`, 288 observations). Following Montgomery et al. (1998), we consider the difference series $\Delta x_n = x_n - x_{n-1}$, and also include a constant term in the AR model. For the quantiser alphabet size, $m = 2$ is a natural choice here, as will become apparent below. The threshold selected using the procedure of Section 6.2.3 is $c = 0.15$, and the resulting MAP tree is the tree of Figure 6.1, with depth $d = 2$, leaves $\mathcal{S} = \{1, 01, 00\}$, and posterior 91.5%. The complete BCT-AR model with its MAP parameters is given below, where $e_n \sim \mathcal{N}(0,1)$,

$$\Delta x_n = \begin{cases} 0.09 + 0.72\ \Delta x_{n-1} - 0.30\ \Delta x_{n-2} + 0.42\ e_n, & \text{if } \Delta x_{n-1} > 0.15, \\ 0.04 + 0.29\ \Delta x_{n-1} - 0.32\ \Delta x_{n-2} + 0.32\ e_n, & \text{if } \Delta x_{n-1} \leq 0.15,\ \Delta x_{n-2} > 0.15, \\ -0.02 + 0.34\ \Delta x_{n-1} + 0.19\ \Delta x_{n-2} + 0.20\ e_n, & \text{if } \Delta x_{n-1} \leq 0.15,\ \Delta x_{n-2} \leq 0.15. \end{cases}$$

Interpretation. The BCT-AR model finds significant structure in the data, providing a very natural interpretation. It identifies 3 meaningful states: First, jumps in the unemployment rate higher than 0.15 signify economic contractions (context 1). If there is not a jump at the most recent time-point, the model looks further back to determine the state. Context 00 signifies a stable economy, as there are no jumps in the unemployment rate for two consecutive quarters. Finally, context 01 identifies an intermediate state: "stabilising just after a contraction". An important feature identified by the BCT-AR model is that the volatility is different in each case. It is higher in contractions ($\sigma = 0.42$), smaller in stable economy regions ($\sigma = 0.20$), and somewhere in-between for context 01 ($\sigma = 0.32$). This is an important finding, verifying that there is much higher uncertainty in economic contractions.

Forecasting. In addition to its appealing interpretation, the BCT-AR model together with the MSA model outperform all benchmarks in forecasting in this example, achieving the lowest MSE, as it can be observed from the results of Table 6.1. In terms of running times, BCT-AR vastly outperforms the MSA model (Appendix D.3).

US Gross National Product. Another important example of nonlinear time series in economics is the US Gross National Product (GNP) (Hansen, 2011; Potter, 1995). We study the quarterly US GNP in the time period from 1947 to 2019 (dataset `gnp`, 291 observations). Following Potter (1995), here we consider the difference in the logarithm of the series, $y_n = \log x_n - \log x_{n-1}$. As above, $m = 2$ is a natural choice for the quantiser, helping to differentiate economic expansions from contractions, which govern the underlying dynamics.

The MAP BCT-AR tree is shown in Figure 6.2: It has depth $d = 3$, and four leaves, $\mathcal{S} = \{0, 10, 110, 111\}$, while its posterior is 42.6%. The complete set of MAP parameters at its leaves are given in Appendix D.2.

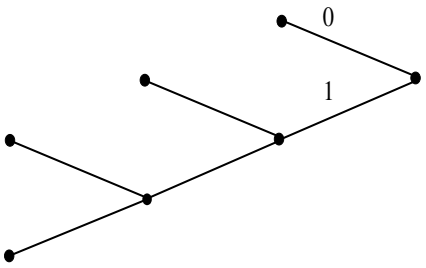


Figure 6.2: MAP tree model in the US GNP example.

Interpretation. Compared with the previous example, here the MAP BCT-AR model finds even richer structure in the data and identifies four meaningful states. First, as before, there is a single state corresponding to an economic contraction – which now corresponds to $s = 0$ instead of $s = 1$, as the GNP obviously increases in expansions and decreases in contractions. And again, the model does not look further back whenever a contraction is detected – this is a *renewal state*. In this example, the model shows that the effect of a contraction is still present even after *three* quarters ($s = 110$), and that the exact 'distance' from a contraction is also important, with the dynamics changing depending on how much time has elapsed. Finally, the state $s = 111$ corresponds to a flourishing, expanding economy, without a contraction in the recent past. An important feature captured by the model is again that the volatility is different in each case and, enhancing our previous findings, it is found to monotonically decrease with the distance from the last contraction. It starts with the maximum value of $\sigma = 1.23$ for $s = 0$, and strictly decreases to $\sigma = 0.75$ for $s = 111$; see Appendix D.2.

Forecasting. Because of the additional structure it identifies here, the BCT-AR model is found to outperform all benchmarks in forecasting (Table 6.1). It gives a significantly lower MSE than the second-best method (9% diference), and is computationally very efficient.

IBM stock price. Finally, we examine the daily IBM common stock closing price from May 17, 1961 to November 2, 1962 (dataset `ibm`, 369 observations), taken from Box et al. (2015). This is a well-studied dataset, with Box et al. (2015) fitting an ARIMA model, Tong (1990) fitting a SETAR model and Wong and Li (2000) fitting a MAR model to the data. Following previous approaches, we consider the first-difference series, $\Delta x_n = x_n - x_{n-1}$. For the alphabet size of the quantiser we choose $m = 3$, with contexts $\{0, 1, 2\}$ naturally corresponding to the states {down, steady, up} for the stock price.

Using the procedure of Section 6.2.3 to select the thresholds, the resulting quantiser regions are: $s = 0$ if $\Delta x_{n-1} < -7$, $s = 2$ if $\Delta x_{n-1} > 7$, and $s = 1$ otherwise. The MAP tree is shown in Figure 6.3: It has depth $d = 2$, and its leaves are $\mathcal{S} = \{0, 2, 10, 11, 12\}$, hence identifying five states. Its posterior is 99.3%, suggesting that there is very strong evidence in the data supporting this exact structure, even with only 369 observations. The complete BCT-AR model with its MAP parameters is given in Appendix D.2.

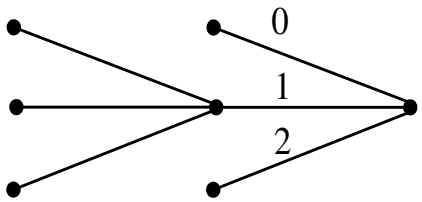


Figure 6.3: MAP tree model in the IBM stock price example.

Interpretation. The BCT-AR model reveals important information about apparent structure in the data, which has not been identified before. Firstly, it admits a very simple and natural interpretation: In order to determine the AR model generating the next value, we need to look back until there is a significant enough price change (corresponding to contexts 0, 2, 10, 12), or until we reach the maximum depth of 2 (context 11).

Another important feature captured by this model is the commonly observed asymmetric response in volatility due to positive and negative shocks, sometimes called the *leverage effect* (Box et al., 2015; Tsay, 2005). As desired, the MAP model shows that negative shocks increase the volatility much more. Specifically, $s = 0$ has the highest volatility ($\sigma = 12.3$), with $s = 10$ being a close second ($\sigma = 10.8$), showing that the effect of a past shock is still present. In all other cases the volatility is much smaller (between $\sigma = 5.17$ and $\sigma = 6.86$). Finally, we observe that when stabilising after a shock (contexts 10, 12), the latest value x_{n-1} is not as important as x_{n-2}, whereas x_{n-1} is dominant in all other cases.

Forecasting. From Table 6.1, it is observed that DeepAR outperforms all methods here, with the BCT-AR model giving a marginally higher MSE but much smaller empirical running times; see Appendix D.3.

Summary. In this section, we associated AR models to the leaves of the context trees as an example of the general BCT-SSM. The resulting BCT-AR model was found to be useful in practice, admitting interesting interpretations and outperforming several state-of-the-art methods in forecasting the future values of time series. Next, we associate conditional heteroscedastic models at the leaves of the context trees, and give another interesting example of the general BCT-SSM that is also of high practical importance.

6.3 The BCT-ARCH model

An important aspect of financial time series analysis is in modelling the dynamic evolution of volatility over time. To capture the well-known *volatility clustering* present in financial data, in his seminal work, Engle (1982) introduced the *autoregressive conditional heteroscedasticity* (ARCH) models, which together with their numerous extensions have been very successful in modelling the volatility of financial time series. Perhaps the most important limitation of ARCH models is their inability to model the well-known asymmetric response in volatility due to positive and negative shocks, sometimes called the *leverage effect* (Box et al., 2015; Tsay, 2005). A number of approaches have been developed to incorporate this feature in ARCH models, notably including the works of Glosten et al. (1993); Gray (1996); Nelson (1991); Zakoian (1994); see Section 6.3.4 for more details on these.

As a second example of the general BCT-SSM, we now associate ARCH models at the leaves of the context trees, and refer to the resulting mixture model as the *Bayesian Context Trees-ARCH* (BCT-ARCH) model. The BCT-ARCH model is of high practical importance because, together with its associated methods for Bayesian inference, it gives a systematic and powerful way of modelling the volatility asymmetries in financial data. It is found to outperform previous approaches in real examples, and to reveal important structure that has not been identified before.

6.3.1 Bayesian modelling and inference

To build the BCT-ARCH model, we associate an ARCH model of order p to each leaf s of the context tree T, so that,

$$x_n \sim \mathcal{N}\left(0, \sigma_n^2\right), \quad \sigma_n^2 = \alpha_{s,0} + \alpha_{s,1} x_{n-1}^2 + \cdots + \alpha_{s,p} x_{n-p}^2 = \alpha_s^{\mathrm{T}} \, \mathbf{z}_{n-1}, \tag{6.14}$$

where $\theta_s = \alpha_s = (\alpha_{s,0}, \alpha_{s,1}, \ldots, \alpha_{s,p})^{\mathrm{T}}$ and $\mathbf{z}_{n-1} = (1, x_{n-1}^2, \ldots, x_{n-p}^2)^{\mathrm{T}}$.

Prior specification. The parameters of the model here are the ARCH coefficients, $\theta_s = \alpha_s$. In contrast with the case of simple AR models, Bayesian inference for the ARCH coefficients is a harder task that cannot be performed in closed form, with a a number of MCMC approaches introduced in the literature for that; see Virbickaite et al. (2015) for a review. In this work we follow Vrontos et al. (2000) and use the following non-informative priors: $\pi(\alpha_{s,0}) = 1/\alpha_{s,0}$, $\pi(\alpha_{s,1}) = \pi(\alpha_{s,2}) = \cdots = \pi(\alpha_{s,p}) = U(0,1)$. It is noted that there are no hyperparameters to be tuned for this prior.

6.3.2 Approximating the estimated probabilities

As explained in Section 6.1.2, our inference algorithms rely on the estimated probabilities $P_e(s,x)$ given in (6.4). However, in this example one would not hope to compute these in closed form, as exact Bayesian inference is not possible even in the case of a single ARCH model. So, we are in the setting where we need to compute these quantities approximately and use the resulting approximations in our CCTW and CBCT algorithms, the remaining methodology remaining identical with before.

As noted in Section 6.1.2, the fact that $P_e(s,x)$ are in the form of marginal likelihoods allows us to use standard methods to approximate them; see Kass and Raftery (1995) for a general discussion on that. One approach would be to obtain MCMC samples from the parameter posterior and employ these to estimate $P_e(s,x)$ using the work of Chib (1995); Chib and Jeliazkov (2001). Instead, we use a variant of the Laplace method for approximating marginal likelihoods described in Kass and Raftery (1995), which was found to be effective in the case of a multivariate GARCH model in Vrontos et al. (2003a); see Kass and Raftery (1995) and Vrontos et al. (2003a) for justifications and theoretical guarantees of this approximation. The main difference needed to adapt this method to our setting is that we need to consider the likelihood at every context s, as below.

Approximating $P_e(s,x)$. The resulting approximation is given by:

$$\widehat{P}_e(s,x) = (2\pi)^{p+1/2} \, |\widehat{I}_s|^{1/2} \, \exp\left\{L_s(\widehat{\theta}_s)\right\} \, \pi(\widehat{\theta}_s), \tag{6.15}$$

where $L_s(\theta_s)$ is the log-likelihood of data with context s,

$$L_s(\theta_s) = \sum_{i \in B_s} \log p(x_i | x_{-D+1}^{i-1}, \theta_s), \tag{6.16}$$

$\widehat{\theta}_s$ is its maximiser, which can be computed iteratively using the Fisher scoring algorithm,

$$\widehat{\theta}_s^{\,(k)} = \widehat{\theta}_s^{\,(k-1)} + \widehat{I}_s^{-1} \left. \frac{\partial L_s}{\partial \theta_s} \right|_{\theta_s = \widehat{\theta}_s^{\,(k-1)}} , \tag{6.17}$$

and $\widehat{I}_s$ the corresponding expected information matrix computed at $\widehat{\theta}_s^{\,(k-1)}$. These quantities for the BCT-ARCH model are given in Lemma 6.3.

Lemma 6.3. *For the BCT-ARCH model, the terms $L_s(\theta_s), \partial L_s/\partial\theta_s$, and $\widehat{I}_s$ are given by,*

$$L_s(\theta_s) = -\frac{|B_s|}{2}\log(2\pi) - \frac{1}{2}\sum_{i\in B_s}\left(\log\sigma_i^2 + \frac{x_i^2}{\sigma_i^2}\right), \tag{6.18}$$

$$\frac{\partial L_s}{\partial\theta_s} = \frac{1}{2}\sum_{i\in B_s}\frac{1}{\sigma_i^2}\left(\frac{x_i^2}{\sigma_i^2} - 1\right)\mathbf{z}_{i-1}, \tag{6.19}$$

$$\widehat{I}_s = \left\{-\mathbb{E}\left(\frac{\partial^2 L_s}{\partial\theta_s^2}\right)\right\} = \frac{1}{2}\sum_{i\in B_s}\left(\frac{1}{\sigma_i^4}\right)\mathbf{z}_{i-1}\mathbf{z}_{i-1}^T, \tag{6.20}$$

where B_s is as in (6.2)*, and $\sigma_i^2 = \theta_s^T\,\mathbf{z}_{i-1}$.*

Proof. The proof of Lemma 6.3 is given in Appendix B.3; if follows mostly from explicit computations. □

6.3.3 Computational complexity

For a single iteration of the scoring algorithm, the situation is very similar with the BCT-AR case. For every symbol x_i in a time series x_1^n, exactly $D+1$ nodes need to be updated. And for each one of these nodes, only the quantities in (6.18)–(6.20) need to be updated, which can be done efficiently by just adding one term to each sum. The only difference is that this forward pass of the data now needs to be repeated for every iteration of the scoring algorithm, every time followed by the Fisher updates of (6.17) for the nodes of T_{MAX}. Finally, the recursive step remains identical with before, only performing operations on T_{MAX}.

Denoting the number of iterations of the scoring algorithm as M, the above discussion shows that the complexity of CCTW and CBCT as a function of n is $O(nMDp^2)$. Remarkably, this is again just D times higher than $O(nMp^2)$, i.e., from the complexity of fitting a single ARCH model to the data, showing that again, we can perform inference in this much richer model class at essentially no additional computational cost.

6.3.4 Alternative methods

Before presenting experimental results, we give a brief summary of commonly used alternative methods for modelling volatility and the leverage effect; we will be comparing with these methods in the experimental section. Useful resources include the `R` packages `rugarch` (Ghalanos, 2022), `MSGARCH` (Ardia et al., 2019), and `stochvol` (Kastner, 2016). Here we briefly describe the models used, and refer to the packages' documentation and Appendix D.1 for more details on the methods and the training procedures carried out.

GARCH and EGARCH models. The most widely used extension of ARCH models are the Generalised ARCH (GARCH) models of Bollerslev (1986). The GARCH(p,q) model is given by,

$$\sigma_n^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i x_{n-i}^2 + \sum_{i=1}^{q} \beta_i \sigma_{n-i}^2, \tag{6.21}$$

with the simple GARCH$(1,1)$ model being the most commonly used in practice. An important extension of the GARCH model is the Exponential GARCH (EGARCH) model of Nelson (1991), where the parametrisation in now in terms of the logarithm of σ_n^2.

GJR models. A common alternative of the GARCH model to explicitly capture the volatility asymmetries is the threshold model of Glosten, Jagannathan, and Runkle (GJR), which was introduced in Glosten et al. (1993) as,

$$\sigma_n^2 = \alpha_0 + \sum_{i=1}^{p} (\alpha_i + \gamma_i I_{n-i}) x_{n-i}^2 + \sum_{i=1}^{q} \beta_i \sigma_{n-i}^2, \tag{6.22}$$

where I_{n-i} is the indicator function for x_{n-i} being *negative*, hence enforcing negative returns to have higher volatility (i.e., all coefficients are positive). A similar threshold GARCH model parametrised in terms of the standard deviation instead of the variance of the process is given by Zakoian (1994).

MSGARCH models. The Markov switching GARCH (MSGARCH) model (Gray, 1996; Haas et al., 2004) is a Hidden Markov Model that is the exact analogue of the MSA model in this setting, with GARCH models in place of the AR models associated to each discrete state. Similarly with the MSA, performing inference in this setting is much more challenging compared to the previous approaches.

SV models. An alternative to the ARCH family – which models the evolution of volatility deterministically – is the family of *stochastic volatility* (SV) models, which model the volatility probabilistically. This is usually done using a Hidden Markov Model, where the hidden state is the logarithm of the variance of the process and its evolution is modelled as an AR process; see Kastner (2016); Shephard and Andersen (2009) for more details. As with MSGARCH, the randomness present in the hidden state process makes inference much more challenging and expensive compared to the other approaches.

6.3.5 Experimental results

As with the BCT-AR model, we first perform simulation experiments verifying that our inferential procedures are effective with data generated by BCT-ARCH models.

Simulated data. As a first example, data are generated from a BCT-ARCH model where the context tree is the binary tree of depth $d = 1$, with leaves $\mathcal{S} = \{0,1\}$. We consider the evolution of the posterior over trees, $\pi(T|x)$, as more data become available. With $n = 1,000$ observations the MAP tree model is the empty tree, corresponding to a single ARCH model, as there are not yet enough data to support a more complex tree. With $n = 2,500$ observations, the MAP tree model is now the true tree with a posterior of 42%, with $n = 5,000$ observations the posterior of the true tree becomes 90%, and with $n = 10,000$ observations it reaches 99.9%. The true parameters of the complete BCT-ARCH model (left), and the estimates from $n = 5,000$ observations are given below; note that this is a difficult example, as the ARCH models in the two regions are very similar.

$$\sigma_n^2 = \begin{cases} 0.10 + 0.20\, x_{n-1}^2 + 0.20\, x_{n-2}^2 \\ 0.10 + 0.20\, x_{n-1}^2 \end{cases}, \quad \widehat{\sigma}_n^2 = \begin{cases} 0.10 + 0.20\, x_{n-1}^2 + 0.16\, x_{n-2}^2, & \text{if } x_{n-1} \le 0, \\ 0.10 + 0.21\, x_{n-1}^2 + 0.02\, x_{n-2}^2, & \text{if } x_{n-1} > 0. \end{cases}$$

As a second example, we consider the binary tree with depth $d = 2$ and leaves $\mathcal{S} = \{1,01,00\}$. The true model (left) and the model fitted from $n = 5,000$ observations are given below, with the posterior of the true tree being 98%.

$$\sigma_n^2 = \begin{cases} 0.20 + 0.30\, x_{n-1}^2 + 0.10\, x_{n-2}^2 \\ 0.10 + 0.20\, x_{n-1}^2 + 0.10\, x_{n-2}^2 \\ 0.10 + 0.20\, x_{n-1}^2 \end{cases}, \quad \widehat{\sigma}_n^2 = \begin{cases} 0.20 + 0.30\, x_{n-1}^2 + 0.13\, x_{n-2}^2, & \text{if } s = 00, \\ 0.10 + 0.21\, x_{n-1}^2 + 0.14\, x_{n-2}^2, & \text{if } s = 01, \\ 0.10 + 0.20\, x_{n-1}^2 + 0.00\, x_{n-2}^2, & \text{if } s = 1. \end{cases}$$

More extensive simulation experiments are presented in Appendix D.2. In general, the posterior of the true underlying tree is found to converge to 1, and the estimates of the parameters to converge to their true values. So, overall it can be concluded that our inferential procedures are very effective with data generated from BCT-ARCH models.

Real-world applications. In the following, we employ the BCT-ARCH modelling framework to model the volatility in four major financial indices. In every example, we consider the daily values of a stock market index in the last thirty years up to 7 April 2023 (7,821 observations), and examine the transformed log-differenced time series, $y_n = 10\log(x_n/x_{n-1})$. In order to explicitly capture the leverage effect and distinguish between positive and negative shocks, for the BCT-ARCH model we use binary trees with a quantiser threshold $c = 0$.

Important interpretation. The most important feature captured from the BCT-ARCH model is that it allows for an *enhanced leverage effect*. In particular, it reveals that in modelling the asymmetries present in volatility, it is not only the sign of the previous stock change that is relevant, but the complete pattern of recent ups and downs of the stock price might be important. The exact set of patterns (or *states*) that are important are automatically identified from the data as the leaves of the fitted MAP tree model. This relevant set of patterns might depend on the specific example, but in all cases it was found to be richer than $\mathcal{S} = \{0, 1\}$, which corresponds to just the sign of the previous change as in traditional leverage modelling. Hence, it is concluded that this kind of enhanced leverage is required in practice, and that the BCT-ARCH model reveals important structure that has not been identified before.

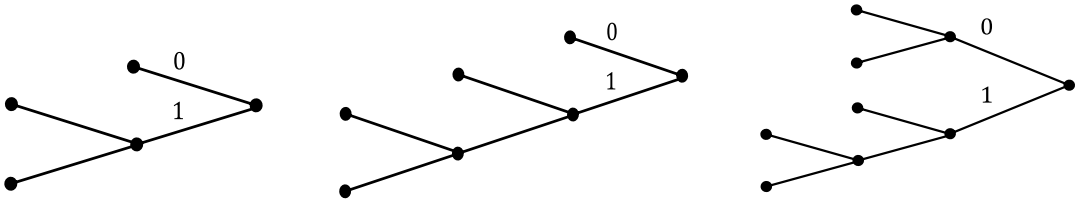


Figure 6.4: MAP tree models for major stock market indices; FTSE 100 (left), CAC 40 and DAX (middle), S&P 500 (right).

FTSE 100. As a first example, we consider the Financial Times Stock Exchange 100 Index, which is the most commonly used UK-based stock market indicator, including 100 companies listed on the London Stock Exchange. The fitted BCT-ARCH model validates the enhanced leverage effect, as it identifies the relevance of three meaningful states. The MAP tree model identified by the CBCT algorithm is shown in Figure 6.4; it has depth $d = 2$ and three leaves, $\mathcal{S} = \{0, 10, 11\}$. Its posterior is 95.2%, signifying that there is strong evidence in the data supporting this exact structure. Context $s = 0$ corresponds to a negative shock at the last timestep, $s = 10$ to stabilising just after a negative shock whose effect is still present, and $s = 11$ to a flourishing period. The complete BCT-ARCH model is given by,

$$\sigma_n^2 = \begin{cases} 0.00 + 0.21\, y_{n-1}^2 + 0.16\, y_{n-2}^2 + 0.21\, y_{n-3}^2 + 0.18\, y_{n-4}^2 + 0.13\, y_{n-5}^2, & \text{if } s = 0, \\ 0.00 + 0.02\, y_{n-1}^2 + 0.19\, y_{n-2}^2 + 0.21\, y_{n-3}^2 + 0.15\, y_{n-4}^2 + 0.12\, y_{n-5}^2, & \text{if } s = 10, \\ 0.00 + 0.00\, y_{n-1}^2 + 0.10\, y_{n-2}^2 + 0.12\, y_{n-3}^2 + 0.09\, y_{n-4}^2 + 0.11\, y_{n-5}^2, & \text{if } s = 11. \end{cases}$$

The ARCH coefficients of $s = 11$ are the smallest for every lag, with $s = 0$ and $s = 10$ having very similar coefficients apart from the one corresponding to the first lag, y_{n-1}^2, which is essentially zero for $s = 10$ (i.e., when it corresponds to a price increase). All these observations are coherent with the fact that negative shocks increase the volatility much more.

CAC 40 and DAX. The next examples we consider are CAC 40 and DAX, the most commonly used stock market indices in France and Germany, respectively, each one consisting of 40 major companies. It turns out that very similar structure is present in these two examples, resulting to an identical MAP tree model identified by the CBCT algorithm (Figure 6.4). The MAP tree has depth $d = 3$ and four leaves, $\mathcal{S} = \{0, 10, 110, 111\}$. The estimated ARCH coefficients at the leaves are given in Appendix D.2 for each dataset.

Importantly, the interpretation is very similar with the FTSE example, with negative shocks being again a *renewal state*. That is, in order to determine the current state, the model looks back in the past until a negative shock is detected. The only difference is in the memory of the process, with negative shocks now being relevant even if they occur *three* timesteps in the past. This gives a total of four different states, meaning that some additional structure has been identified here. In every case, the ARCH coefficients are very small when they correspond to a positive lag. So, for $s = 10$ the coefficient $\alpha_1 \approx 0$, while for $s = 110$ both α_1 and α_2 are small. Overall, the BCT-ARCH model gives evidence of a rich asymmetric response in volatility, with negative shocks always having a stronger effect.

S&P 500. Finally, we examine Standard and Poor's 500 Index, one of the most commonly followed indices worldwide, which consists of 500 of the largest companies in the US. Comparing with the previous examples, the MAP tree model shown in Figure 6.4 reveals even more additional structure: It has depth $d = 3$, five leaves, $\mathcal{S} = \{00, 01, 10, 110, 111\}$, and a posterior of 91.4%, which is again very high. Actually, the MAP tree model is the exact tree of CAC 40 and DAX, but with one additional branch added at $s = 0$. So, apart from identifying slightly more structure, the interpretation of the BCT-ARCH model is very similar with before. The ARCH coefficients (reported in Appendix D.2) are small when they correspond to a positive lag, capturing again an interesting asymmetric response in volatility.

Forecasting. Apart from the appealing interpretation and the apparent utility of the BCT-ARCH model in modelling volatility asymmetries, here we evaluate its performance in forecasting, by comparing with commonly used approaches. As the volatility is not directly observable, effectively comparing different volatility models is known to be a challenging task (Tsay, 2005). Following standard approaches as in Vrontos et al. (2003b) and Dellaportas and Vrontos (2007), in order to compare the predictive ability of the models we consider the predictive distributions in 1-step ahead out-of-sample forecasting. In specific, at every timestep in the test set, all models are estimated using the entire past, and the resulting predictive density is evaluated at the real observed datapoint; the complete set of training details for each method are given in Appendix D.1. We examine the cumulative log-loss, $\mathcal{L} = -\sum_i \log \widehat{p}(y_i | y_{-D+1}^{i-1})$, where the sum is over all datapoints in the test set.

The results are presented in Table 6.2 for the stock market indices. It is observed that in terms of its predictive ability, the BCT-ARCH model outperforms all the alternatives in almost all experiments, with the only exception being the S&P example where EGARCH achieves the best performance. Finally, because of its low computational complexity and its ability to allow for efficient sequential updates, the BCT-ARCH model is also found to outperform the alternatives in terms of their computational requirements, with empirical running times reported in Appendix D.3.

Table 6.2: Comparing the predictive ability of volatility models using the log-loss, $\mathcal{L}$.

	BCT-ARCH	ARCH	GARCH	GJR	EGARCH	MSGARCH	SV
`ftse`	$-\mathbf{161.9}$	-157.7	-154.5	-159.7	-159.0	-159.7	-154.4
`cac40`	$-\mathbf{112.5}$	-108.6	-108.7	-111.0	-112.4	-109.2	-106.9
`dax`	$-\mathbf{111.7}$	-105.9	-105.4	-106.4	-107.5	-106.1	-103.2
`s&p`	-78.73	-74.89	-81.04	-83.89	$-\mathbf{84.58}$	-80.95	-80.16

Summary. In the above experiments, the BCT-ARCH model was found to admit interesting interpretations, revealing important information that has not been identified before about the asymmetries present in volatility in financial data. At the same time, the BCT-ARCH model was found to be of high practical importance because of its performance in forecasting, where it was found to outperform several commonly used approaches.

Chapter 7

Concluding Remarks

In this final chapter we briefly summarise the main contributions of this thesis, and then discuss some promising directions for future research extending this work.

7.1 Summary of contributions

In general terms, the main purpose of this thesis has been to:

1. Develop flexible model classes that are appropriate and effective in various time series settings, including both discrete-valued and real-valued observations.
2. Introduce a Bayesian inference framework for the resulting model classes, including an appropriate prior structure together with associated methodological and algorithmic tools for efficient and effective Bayesian inference.
3. Illustrate the practical utility of the proposed methods in a number of important statistical tasks, comparing with state-of-the-art approaches in simulated experiments and real-world applications.
4. Provide theoretical results that further justify the use of the proposed methods.

In view of the above general direction, the main contributions of this thesis are briefly summarised below.

Bayesian Context Trees. Chapter 2 contains our core methodological contributions for discrete-valued time series. A Bayesian modelling framework is introduced for the class of variable-memory Markov chains, along with an associated collection of methodological and algorithmic tools allowing for efficient, *exact* Bayesian inference. Extensive experimental results and comparisons with state-of-the-art approaches illustrate the efficacy of our methods in the core statistical tasks of model selection, estimation and prediction.

Posterior representations. In Chapter 3, alternative representations are derived for the BCT prior and posterior on model space, in terms of simple branching processes. Apart from offering a very natural interpretation, these representations are also of high practical importance, providing a simple way of sampling from the BCT posterior, and allowing to establish theoretical results that further justify the BCT framework.

Fully-Bayesian estimation. In Chapters 4 and 5, building on the BCT framework, we consider the challenging tasks of entropy estimation and change-point detection. In both cases, we are able develop a fully-Bayesian approach, providing a way to sample from the desired posterior distribution (i.e., of the entropy rate and the change-points, respectively, in each case). In this way, we can estimate the entire posterior distribution of interest, providing much richer information than simple point estimates.

General Bayesian mixture models. In Chapter 6, by extending the BCT framework, we develop a very general Bayesian framework for building mixture models for *real-valued* time series. The proposed framework can be combined with *any* existing model class as a base model, producing flexible and interpretable mixture models. By extending our inference algorithms, we show that we are still able to perform effective Bayesian inference in this much more general setting, in a very efficient manner. The utility of the general framework is studied in detail when AR and ARCH models are used as the building block for the mixture models. In both cases, the resulting mixture models are found to be flexible nonlinear models of practical interest, which perform well in a number of real-world applications.

7.2 Future work

In closing, it is noted that a number of promising research directions exist for extending the work in thesis. These are briefly described below, starting with the more important, general, and promising directions, and progressing with more specific ideas.

The perhaps most notable direction for further research stems from the generality of the proposed framework for building mixture models. In particular, the BCT-SSM can be combined with any existing state-of-the-art model for each specific application, and give potential improvements. As a few examples, it could be combined, among others, with ARIMA and EGARCH models, with general state space models and HMMs, or even with modern machine learning models including GPs and neural networks, ultimately resulting in very flexible model classes. Perhaps even more importantly, our methods could also be extended to the setting of multivariate time series, which would greatly broaden the scope of applications of the proposed framework, by a lot.

In a different but also quite general direction, BCTs could also be used to model the hidden state process in general HMMs, resulting in a Hidden Bayesian Context Trees model class. In specific, consider any particular HMM that uses an ordinary discrete-valued Markov chain for its hidden state process. By replacing the ordinary Markov chain with a corresponding variable-memory chain modelled as a BCT, we could create a more flexible model, which might yield potential improvements in practice.

Another interesting direction would be to consider the challenging problems of online change-point detection and anomaly detection. These tasks come with critical applications in areas including security and the health sciences, where they can be used to automatically detect and prevent cyber attacks, and to monitor the health condition of patients. A promising idea in this setting would be to combine the BCT framework with the Bayesian Online Change-Point Detection (BOCPD) method of Adams and MacKay (2007), which has been used widely and with great success in combination with simple baseline models to detect change-points in dynamic environments. These two methods are perfectly suited for each other since the only thing that BOCPD really requires is a sequential update of the evidence, which is precisely what the BCT algorithms allow to compute exactly and sequentially.

Lastly, since throughout this thesis we have only considered discrete-time data, another possible extension would be to consider the *continuous-time* setting. In particular, extending the BCT model class to also include holding times for the corresponding states would create an interesting continuous-time jump process, in some sense generalising the class of continuous-time Markov chains. The resulting model might be useful in practice, especially in applications where the observations do not arrive in regular time intervals, so that modelling their arrival times apart from their values is also important.

References

A. Abakuks. The synoptic problem: On Matthew's and Luke's use of Mark. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 175(4):959–975, 2012.

R.P. Adams and D.J.C. MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

N.K. Ahmed, A.F. Atiya, N.E. Gayar, and H. El-Shishiny. An empirical comparison of machine learning models for time series forecasting. *Econometric reviews*, 29(5-6): 594–621, 2010.

H. Akaike. Information theory and an extension of the maximum likelihood principle. In *2nd International Symposium on Information Theory (ISIT)*, pages 267–281, 1973.

A. Alexandrov, K. Benidis, V. Bohlke-Schneider, M.and Flunkert, J. Gasthaus, T. Januschowski, D.C. Maddix, S.S. Rangapuram, D. Salinas, J. Schulz, et al. GluonTS: Probabilistic and neural time series modeling in Python. *Journal of Machine Learning Research*, 21(116):1–6, 2020.

F.M. Alvarez, A. Troncoso, J.C. Riquelme, and J.S.A. Ruiz. Energy time series forecasting based on pattern sequence similarity. *IEEE Transactions on Knowledge and Data Engineering*, 23(8):1230–1243, 2010.

S. Alvisi, M. Franchini, and A. Marinelli. A short-term, pattern-based model for water-demand forecasting. *Journal of Hydroinformatics*, 9(1):39–50, 2007.

S. Aminikhanghahi and D. Cook. A survey of methods for time series change point detection. *Knowledge and Information Systems*, 51(2):339–367, 2017.

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

E.W. Archer, I.M. Park, and J.W. Pillow. Bayesian entropy estimation for binary spike train data using parametric prior knowledge. *Advances in neural information processing systems*, 26, 2013.

D. Ardia, K. Bluteau, K. Boudt, L. Catania, and D.A. Trottier. Markov-switching GARCH models in R: The MSGARCH package. *Journal of Statistical Software*, 91:1–38, 2019.

Y.F. Atchade and J.S. Liu. The Wang-Landau algorithm for Monte Carlo computation in general state spaces. *Statistica Sinica*, 20:209–233, 2004.

K.B. Athreya and P.E. Ney. *Branching processes*. Courier Corporation, 2004.

F. Audrino and P. Bühlmann. Tree-structured generalized autoregressive conditional heteroscedastic models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(4):727–744, 2001.

S. Bacallado. Bayesian analysis of variable-order, reversible Markov chains. *The Annals of Statistics*, 39(2):838–864, 2011.

S. Bacallado, J.D. Chodera, and V. Pande. Bayesian comparison of Markov models of molecular dynamics with detailed balance constraint. *The Journal of chemical physics*, 131(4):045106, 2009.

S. Bacallado, S. Favaro, and L. Trippa. Bayesian nonparametric analysis of reversible Markov chains. *The Annals of Statistics*, 41(2):870–896, 2013.

S. Bacallado, V. Pande, S. Favaro, and L. Trippa. Bayesian regularization of the length of memory in reversible sequences. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(4):933–946, 2016.

A.R. Barron, J. Rissanen, and B. Yu. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–2760, 1998.

A.R. Barron, L. Birgé, and P. Massart. Risk bounds for model selection via penalization. *Probability theory and related fields*, 113(3):301–413, 1999.

R. Begleiter, R. El-yaniv, and G. Yona. On prediction using variable order Markov models. *Journal of Artificial Intelligence Research*, 22:385–421, 2004.

G. Bejerano and G. Yona. Variations on probabilistic suffix trees: Statistical modeling and prediction of protein families. *Bioinformatics*, 17(1):23–43, 2001.

R. Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

I. Ben-Gal, A. Shani, A. Gohr, J. Grau, S. Arviv, A. Shmilovici, S. Posch, and I. Grosse. Identification of transcription factor binding sites with variable-order Bayesian networks. *Bioinformatics*, 21(11):2657–2666, 2005.

K. Benidis, S.S. Rangapuram, V. Flunkert, B. Wang, D. Maddix, C. Turkmen, J. Gasthaus, M. Bohlke-Schneider, D. Salinas, L. Stella, et al. Neural forecasting: Introduction and literature overview. *arXiv preprint arXiv:2004.10240*, 2020.

A. Berchtold. Modélisation autorégressive des chaînes de Markov: Utilisation d'une matrice différente pour chaque retard. *Revue de Statistique Appliquée*, 44(3):5–25, 1996.

A. Berchtold. *Chaînes de Markov et modèles de transition: Applications aux sciences sociales*. Hermes, 1998.

A. Berchtold. Estimation in the mixture transition distribution model. *Journal of Time Series Analysis*, 22(4):379–397, 2001.

A. Berchtold and A.E. Raftery. The mixture transition distribution model for high-order Markov chains and non-Gaussian time series. *Statistical Science*, 17(3):328–356, 2002.

J.M. Bernardo and A.F.M. Smith. *Bayesian theory*. John Wiley & Sons, 1994.

D.J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD Workshop*, volume 10, pages 359–370, 1994.

C.M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

D. Blackwell. Conditional expectation and unbiased sequential estimation. *The Annals of Mathematical Statistics*, 18(1):105–110, 1947.

T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

G. N. Boshnakov and D. Ravagli. *mixAR: Mixture Autoregressive Models*, 2021. R package version 0.22.5. https://CRAN.R-project.org/package=mixAR.

G.E.P. Box, G.M. Jenkins, G.C. Reinsel, and G.M. Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

R.J. Boys and D.A. Henderson. A Bayesian approach to DNA sequence segmentation. *Biometrics*, 60(3):573–581, 2004.

J.V. Braun and H.G. Müller. Statistical methods for DNA sequence segmentation. *Statistical Science*, 13(2):142–162, 1998.

J.V. Braun, R.K. Braun, and H.G. Müller. Multiple changepoint fitting via quasilikelihood, with application to DNA sequence segmentation. *Biometrika*, 87(2):301–314, 2000.

L. Breiman. *Probability*. SIAM, 1992.

L. Breiman, J. Friedman, C.J. Stone, and R.A. Olshen. *Classification and regression trees*. CRC Press, 1984.

S.R. Browning. Multilocus association mapping using variable-length Markov chains. *The American Journal of Human Genetics*, 78(6):903–913, 2006.

P. Bühlmann. Model selection for variable length Markov chains and tuning the context algorithm. *Annals of the Institute of Statistical Mathematics*, 52(2):287–315, 2000.

P. Bühlmann and A.J. Wyner. Variable length Markov chains. *Annals of Statistics*, 27(2): 480–513, 1999.

S. Bunton. *On-line stochastic processes in data compression*. Ph.D. thesis, University of Washington, 1996.

J.R. Busch, P.A. Ferrari, A.G. Flesia, R. Fraiman, S.P. Grynberg, and F. Leonardi. Testing statistical hypothesis on random trees and applications to the protein classification problem. *Annals of Applied Statistics*, 3(2):542–563, 2009.

H. Cai, S.R. Kulkarni, and S. Verdú. Universal entropy estimation via block sorting. *IEEE Transactions on Information Theory*, 50(7):1551–1561, 2004.

W. Cai, S. Borlace, M. Lengaigne, P. Rensch, M. Collins, G. Vecchi, A. Timmermann, A. Santoso, M. McPhaden, L. Wu, M. England, G. Wang, E. Guilyardi, and F.F. Jin. Increasing frequency of extreme El Niño events due to greenhouse warming. *Nature Climate Change*, 4(2):111–116, 2014.

J.Q. Candela, A. Girard, J. Larsen, and C.E. Rasmussen. Propagation of uncertainty in Bayesian kernel models-application to multiple-step ahead forecasting. In *2003 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 701–704, 2003.

P. Cashin, K. Mohaddes, and M. Raissi. Fair weather or foul? The macroeconomic effects of El Niño. *Journal of International Economics*, 106(C):37–54, 2017.

O. Catoni. *Statistical learning theory and stochastic optimization*, volume 1851 of *Lecture Notes in Mathematics*. Springer Science & Business Media, 2004.

K.S. Chan. Consistency and limiting distribution of the least squares estimator of a threshold autoregressive model. *Annals of Statistics*, 21(1):520–533, 1993.

K.S. Chan and B. Ripley. *TSA: Time Series Analysis*, 2020. R package version 1.3. https://CRAN.R-project.org/package=TSA.

V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection for discrete sequences: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):823–839, 2012.

S. Chib. Marginal likelihood from the Gibbs output. *Journal of the American statistical association*, 90(432):1313–1321, 1995.

S. Chib and E. Greenberg. Understanding the Metropolis-Hastings algorithm. *The American Statistician*, 49(4):327–335, 1995.

S. Chib and I. Jeliazkov. Marginal likelihood from the Metropolis–Hastings output. *Journal of the American statistical association*, 96(453):270–281, 2001.

H. Chipman, E.I. George, R.E. McCulloch, M. Clyde, D.P. Foster, and R.A. Stine. The practical implementation of bayesian model selection. *Lecture Notes-Monograph Series*, pages 65–134, 2001.

H.A. Chipman, E.I. George, and R.E. McCulloch. Bayesian CART model search. *Journal of the American Statistical Association*, 93(443):935–948, 1998.

J. Chung, K. Kastner, L. Dinh, K. Goel, A.C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28, 2015.

K.L. Chung. *Markov chains with stationary transition probabilities*. Springer-Verlag, 1967.

G.A. Churchill. Stochastic models for heterogeneous DNA sequences. *Bulletin of Mathematical Biology*, 51(1):79–94, 1989.

G.A. Churchill. Hidden Markov chains and the analysis of genome structure. *Computers & Chemistry*, 16(2):107–115, 1992.

K. Clark, I. Karsch-Mizrachi, D.J. Lipman, J. Ostell, and E.W. Sayers. GenBank. *Nucleic Acids Research*, 44(D1):D67–D72, 2016. Online at: www.ncbi.nlm.nih.gov.

J. Cleary and I. Witten. Data compression using adaptive coding and partial string matching. *IEEE transactions on Communications*, 32(4):396–402, 1984.

T.M. Cover and J.A. Thomas. *Elements of information theory*. John Wiley & Sons, 1999.

M.K. Cowles and B.P. Carlin. Markov chain Monte Carlo convergence diagnostics: A comparative review. *Journal of the American Statistical Association*, 91(434):883–904, 1996.

W. Craig. *The song of the wood pewee (Myiochanes virens Linnaeus): A study of bird music*. University of the State of New York, 1943.

J.D. Cryer and K.S. Chan. *Time series analysis: with applications in R*. Springer Science & Business Media, 2008.

I. Csiszár and P.C. Shields. The consistency of the BIC Markov order estimator. *Annals of Statistics*, 28(6):1601–1619, 2000.

I. Csiszár and Z. Talata. Context tree estimation for not necessarily finite memory processes, via BIC and MDL. *IEEE Transactions on Information theory*, 52(3):1007–1016, 2006.

R.A. Davis, S.H. Holan, R. Lund, and N. Ravishanker. *Handbook of discrete-valued time series*. CRC Press, 2016.

P. Dayan and L.F. Abbott. *Theoretical neuroscience: Computational and mathematical modeling of neural systems*. MIT Press, 2001.

P. Dellaportas and I.D. Vrontos. Modelling volatility asymmetries: a Bayesian analysis of a class of tree structured multivariate GARCH models. *The Econometrics Journal*, 10(3): 503–520, 2007.

C. Dimitrakakis. Bayesian variable order Markov models. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 161–168, 2010.

A. Doucet, N. De Freitas, N.J. Gordon, et al. *Sequential Monte Carlo methods in practice*. Springer, 2001.

J. Durbin and S.J. Koopman. *Time series analysis by state space methods*, volume 38. Oxford University Press, 2012.

R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge University Press, 1998.

J.J. Dziak, D.L. Coffman, S.T. Lanza, R. Li, and L.S. Jermiin. Sensitivity and specificity of information criteria. *Briefings in Bioinformatics*, 21(2):553–565, 2019.

S. Eleftheriadis, T. Nicholson, M. Deisenroth, and J. Hensman. Identification of Gaussian process state space models. *Advances in neural information processing systems*, 30, 2017.

R.F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the econometric society*, pages 987–1007, 1982.

L. Fahrmeir and H. Kaufmann. Regression models for non-stationary categorical time series. *Journal of Time Series Analysis*, 8(2):147–160, 1987.

C. Faloutsos, J. Gasthaus, T. Januschowski, and Y. Wang. Forecasting big time series: old and new. *Proceedings of the VLDB Endowment*, 11(12):2102–2105, 2018.

P. Fearnhead. Exact and efficient Bayesian inference for multiple changepoint problems. *Statistics and Computing*, 16(2):203–213, 2006.

A. Feutrill and M. Roughan. A review of Shannon and differential entropy rate estimation. *Entropy*, 23(8), 2021.

K. Fokianos. Count time series models. In *Handbook of statistics*, volume 30, pages 315–347. Elsevier, 2012.

K. Fokianos and B. Kedem. Regression theory for categorical time series. *Statistical Science*, 18(3):357–376, 2003.

N. Friel and A.N. Pettitt. Marginal likelihood estimation via power posteriors. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(3):589–607, 2008.

R. Frigola. *Bayesian time series learning with Gaussian processes*. PhD thesis, University of Cambridge, 2015.

R. Frigola, F. Lindsten, T.B. Schön, and C.Ed. Rasmussen. Bayesian inference and learning in Gaussian process state-space models with particle MCMC. *Advances in neural information processing systems*, 26, 2013.

R. Frigola, Y. Chen, and C.E. Rasmussen. Variational Gaussian process state-space models. *Advances in neural information processing systems*, 27, 2014.

T.C. Fu, F.L. Chung, R. Luk, and C.M. Ng. Stock time series pattern matching: Template-based vs. rule-based approaches. *Engineering Applications of Artificial Intelligence*, 20 (3):347–364, 2007.

R.E. Funderlic and C.D. Meyer. Sensitivity of the stationary distribution vector for an ergodic Markov chain. *Linear Algebra and its Applications*, 76:1–17, 1986.

A. Gabadinho and G. Ritschard. Analyzing state sequences with probabilistic suffix trees: The PST R package. *Journal of Statistical Software*, 72(3):1–39, 2016.

A. Galves, C. Galves, J.E. Garcia, N.L. Garcia, and F. Leonardi. Context tree selection and linguistic rhythm retrieval from written texts. *Annals of Applied Statistics*, 6(1):186–209, 2012.

Y. Gao, I. Kontoyiannis, and E. Bienenstock. From the entropy to the statistical structure of spike trains. In *2006 IEEE International Symposium on Information Theory (ISIT)*, 2006.

Y. Gao, I. Kontoyiannis, and E. Bienenstock. Estimating the entropy of binary time series: Methodology, some theory and a simulation study. *Entropy*, 10(2):71–99, 2008.

J.E. García and V.A. González-López. Consistent estimation of partition Markov models. *Entropy*, 19(4):160, 2017.

J.E. García and V.A. González-López. Minimal Markov models. In *Fourth Workshop on Information Theoretic Methods in Science and Engineering*, page 25, 2011.

S. Geisser. *Predictive inference: An introduction*. CRC Press, 1993.

A.E. Gelfand and A.F.M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

A. Gelman and D.B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4):457–472, 1992.

A. Gelman, J.B. Carlin, H.S. Stern, D.B. Dunson, A. Vehtari, and D.B. Rubin. *Bayesian data analysis*. CRC press, 2013.

A. Ghalanos. *rugarch: Univariate GARCH Models*, 2022. R package version 1.4. https://CRAN.R-project.org/package=rugarch.

A.L. Gibbs and F.E. Su. On choosing and bounding probability metrics. *International Statistical Review*, 70(3):419–435, 2002.

A. Girard, C. Rasmussen, J.Q. Candela, and R. Murray-Smith. Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. *Advances in neural information processing systems*, 15, 2002.

L.R. Glosten, R. Jagannathan, and D.E. Runkle. On the relation between the expected value and the volatility of the nominal excess return on stocks. *The journal of finance*, 48(5): 1779–1801, 1993.

G.H. Golub and C.D. Meyer. Using the QR factorization and group inversion to compute, differentiate, and estimate the sensitivity of stationary probabilities for Markov chains. *SIAM Journal on Algebraic Discrete Methods*, 7(2):273–281, 1986.

A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

S.F. Gray. Modeling the conditional distribution of interest rates as a regime-switching process. *Journal of Financial Economics*, 42(1):27–62, 1996.

P.J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

G.G. Gregoriou, S.J. Gotts, H. Zhou, and R. Desimone. High-frequency, long-range coupling between prefrontal and visual cortex during attention. *Science*, 324(5931):1207–1210, 2009.

G.G. Gregoriou, S.J. Gotts, and R. Desimone. Cell-type-specific synchronization of neural activity in FEF with V4 during attention. *Neuron*, 73(3):581–594, 2012.

P. Grünwald. *The minimum description length principle*. MIT Press, 2007.

M. Gupta and J.S. Liu. Discussion of "A Bayesian approach to DNA sequence segmentation". *Biometrics*, 60(3):582–583, 2004.

R. Gwadera, A. Gionis, and H. Mannila. Optimal segmentation using tree models. *Knowledge and Information Systems*, 15(3):259–283, 2008.

M. Haas, S. Mittnik, and Marc S Paolella. A new approach to Markov-switching GARCH models. *Journal of financial Econometrics*, 2(4):493–530, 2004.

J.D. Hamilton. A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica: Journal of the econometric society*, pages 357–384, 1989.

J.D. Hamilton. *Time series analysis*. Princeton University Press, 2020.

E.J. Hannan and B.G. Quinn. The determination of the order of an autoregression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 41(2):190–195, 1979.

B.E. Hansen. Threshold autoregression in economics. *Statistics and its Interface*, 4(2): 123–127, 2011.

T.E. Harris. *The theory of branching processes*, volume 6. Springer, 1963.

W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97, 1970.

M. Heiner and A. Kottas. Autoregressive density modeling with the Gaussian process mixture transition distribution. *Journal of Time Series Analysis*, 43(2):157–177, 2022a.

M. Heiner and A. Kottas. Estimation and selection for high-order Markov chains with Bayesian mixture transition distribution models. *Journal of Computational and Graphical Statistics*, 31(1):100–112, 2022b.

M. Heiner, A. Kottas, and S. Munch. Structured priors for sparse probability vectors with application to model selection in Markov chains. *Statistics and Computing*, 29(5):1077–1093, 2019.

A.O. Hero and O.J. Michel. Asymptotic theory of greedy approximations to minimal k-point random graphs. *IEEE Transactions on Information Theory*, 45(6):1921–1938, 1999.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Q. Hu, P. Su, D. Yu, and J. Liu. Pattern-based wind speed prediction based on generalized principal component analysis. *IEEE Transactions on Sustainable Energy*, 5(3):866–874, 2014.

R. Hyndman, A.B. Koehler, J.K. Ord, and R.D. Snyder. *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008.

R. J. Hyndman. *fma: Data sets from "Forecasting: methods and applications" by Makridakis, Wheelwright & Hyndman (1998)*, 2020. R package version 2.4. https://cran.r-project.org/package=fma.

R.J. Hyndman and Y. Khandakar. Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*, 26(3):1–22, 2008. https://CRAN.R-project.org/package=forecast.

I.A. Ibragimov. Some limit theorems for stationary processes. *Theory of Probability and its Applications*, 7(4):349–382, 1962.

V. Jääskinen, J. Xiong, J. Corander, and T. Koski. Sparse Markov chains for sequence data. *Scandinavian Journal of Statistics*, 41(3):639–655, 2014.

T. Jacob and R.K. Bansal. Sequential change detection based on universal compression algorithms. In *2008 IEEE International Symposium on Information Theory (ISIT)*, pages 1108–1112, 2008.

H. Jeffreys. *The theory of probability*. Oxford University Press, 1998.

J. Jiao, H.H. Permuter, L. Zhao, Y.H. Kim, and T. Weissman. Universal estimation of directed information. *IEEE Transactions on Information Theory*, 59(10):6220–6242, 2013.

D.R. Juvvadi and R.K. Bansal. Sequential change detection using estimators of entropy & divergence rate. In *2013 National Conference on Communications (NCC)*, pages 1–5, 2013.

R.E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the American Society of Mathematical Engineering, Journal of Basic Engineering*, 82(D): 35–45, 1960.

M. Karl, M. Soelch, J. Bayer, and P. Van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432*, 2016.

R.E. Kass and A.E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

G. Kastner. Dealing with stochastic volatility in time series using the R package stochvol. *Journal of Statistical Software*, 69(5):1–30, 2016.

A. Kehagias. A hidden Markov model segmentation procedure for hydrological and environmental time series. *Stochastic Environmental Research and Risk Assessment*, 18(2): 117–130, 2004.

A. Kershenbaum. Entropy rate as a measure of animal vocal complexity. *Bioacoustics*, 23 (3):195–208, 2014.

Y. Kharin and A. Piatlitski. Statistical analysis of discrete time series based on the MC(s,r)-model. *Austrian Journal of Statistics*, 40(1&2):75–84, 2011.

D.P. Kingma and M. Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

I. Kontoyiannis. Second-order noiseless source coding theorems. *IEEE Transactions on Information Theory*, 43(4):1339–1341, 1997.

I. Kontoyiannis. Context-tree weighting and Bayesian Context Trees: Asymptotic and non-asymptotic justifications. *arXiv preprint arXiv:2211.02676*, 2023.

I. Kontoyiannis and M. Skoularidou. Estimating the directed information and testing for causality. *IEEE Transactions on Information Theory*, 62(11):6053–6067, 2016.

I. Kontoyiannis, P.H. Algoet, Y.M. Suhov, and A.J. Wyner. Nonparametric entropy estimation for stationary processes and random fields, with applications to English text. *IEEE Transactions on Information Theory*, 44(3):1319–1327, 1998.

I. Kontoyiannis, L. Mertzanis, A. Panotopoulou, I. Papageorgiou, and M. Skoularidou. Bayesian Context Trees: Modelling and exact inference for discrete time series. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(4):1287–1323, 2022.

W.M. Koolen and S. de Rooij. Universal codes from switching strategies. *IEEE Transactions on Information Theory*, 59(11):7168–7185, 2013.

G. Koop and S.M. Potter. Dynamic asymmetries in US unemployment. *Journal of Business & Economic Statistics*, 17(3):298–312, 1999.

S. Kovats, M. Bouma, S. Hajat, E. Worrall, and A. Haines. El Niño and health. *Lancet*, 362 (9394):1481–1489, 2003.

R. Krichevsky and V. Trofimov. The performance of universal encoding. *IEEE Transactions on Information Theory*, 27(2):199–207, 1981.

R. Krishnan, U. Shalit, and D. Sontag. Structured inference networks for nonlinear state space models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

R.G. Krishnan, U. Shalit, and D. Sontag. Deep Kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

J. Lan, J. Ge, J. Yu, et al. Structure of the SARS-CoV-2 spike receptor-binding domain bound to the ACE2 receptor. *Nature*, 581(7807):215–220, 2020.

E. Lander, L. Linton, B. Birren, et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001.

N.D. Le, R.D. Martin, and A.E. Raftery. Modeling flat stretches, bursts outliers in time series using mixture transition distribution models. *Journal of the American Statistical Association*, 91(436):1504–1515, 1996.

M. Levene and G. Loizou. Computing the entropy of user navigation in the web. *International Journal of Information Technology & Decision Making*, 2(03):459–476, 2003.

W. Li. DNA segmentation as a model selection process. In *Proceedings of the Fifth Annual International Conference on Computational Biology*, pages 204–210, 2001.

A.R. Linero. A review of tree-based Bayesian methods. *Communications for Statistical Applications and Methods*, 24(6):543–559, 2017.

X. Liu, Z. Ni, D. Yuan, Y. Jiang, Z. Wu, J. Chen, and Y. Yang. A novel statistical time-series pattern based interval forecasting strategy for activity durations in workflow systems. *Journal of Systems and Software*, 84(3):354–376, 2011.

X. Liu, H. Jiang, Z. Gu, and J.W. Roberts. High-resolution view of bacteriophage lambda gene expression by ribosome profiling. *Proceedings of the National Academy of Sciences*, 110(29):11928–11933, 2013.

W.Y. Loh. Fifty years of classification and regression trees. *International Statistical Review*, 82(3):329–348, 2014.

M. London, A. Schreibman, M. Häusser, M.E. Larkum, and I. Segev. The information efficacy of a synapse. *Nature Neuroscience*, 5(4):332–340, 2002.

V. Lungu, I. Papageorgiou, and I. Kontoyiannis. Change-point detection and segmentation of discrete data using Bayesian Context Trees. *arXiv preprint arXiv:2203.04341*, 2022a.

V. Lungu, I. Papageorgiou, and I. Kontoyiannis. Bayesian change-point detection via context-tree weighting. In *2022 IEEE Information Theory Workshop (ITW)*, pages 125–130, 2022b.

M. Mächler. *VLMC: Variable Length Markov Chains ('VLMC') Models*, 2022. R package version 1.4. https://CRAN.R-project.org/package=VLMC.

M. Mächler and P. Bühlmann. Variable length Markov chains: Methodology, computing, and software. *Journal of Computational and Graphical Statistics*, 13(2):435–455, 2004.

D.J.C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992.

D.J.C. MacKay. *Information theory, inference and learning algorithms*. Cambridge University Press, 2003.

O. Maitre, K. Emery, O. Buschor, and A. Berchtold. *march: Markov Chains*, 2022. R package version 1.4. https://CRAN.R-project.org/package=march.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, 34(4):802–808, 2018a.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. Statistical and machine learning forecasting methods: Concerns and ways forward. *PloS one*, 13(3):e0194889, 2018b.

J. Massey. Causality, feedback and directed information. In *1990 International Symposium on Information Theory and its Applications (ISITA)*, pages 27–30, 1990.

T. Matsushima and S. Hirasawa. A Bayes coding algorithm using context tree. In *1994 IEEE International Symposium on Information Theory (ISIT)*, page 386, 1994.

T. Matsushima and S. Hirasawa. Reducing the space complexity of a Bayes coding algorithm using an expanded context tree. In *2009 IEEE International Symposium on Information Theory (ISIT)*, pages 719–723, 2009.

C. Meek and D. Chickering, D.M.and Heckerman. Autoregressive tree models for time-series analysis. In *Proceedings of the 2002 SIAM International Conference on Data Mining*, pages 229–244, 2002.

N. Merhav. On the minimum description length principle for sources with piecewise constant parameters. *IEEE Transactions on Information Theory*, 39(6):1962–1967, 1993.

N. Merhav and M. Feder. A strong version of the redundancy-capacity theorem of universal coding. *IEEE Transactions on Information Theory*, 41(3):714–722, 1995.

N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6): 1087–1092, 1953.

C.D. Meyer. The role of the group generalized inverse in the theory of finite Markov chains. *Siam Review*, 17(3):443–464, 1975.

S.P. Meyn and R.L. Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

A.L. Montgomery, V. Zarnowitz, R.S. Tsay, and G.C. Tiao. Forecasting the US unemployment rate. *Journal of the American Statistical Association*, 93(442):478–493, 1998.

K.P. Murphy. *Machine learning: A probabilistic perspective*. MIT Press, 2012.

R. Murray-Smith and A. Girard. Gaussian Process priors with ARMA noise models. In *Irish Signals and Systems Conference*, volume 147, page 152, 2001.

D.B. Nelson. Conditional heteroskedasticity in asset returns: A new approach. *Econometrica: Journal of the econometric society*, 59(2):347–370, 1991.

I. Nemenman, W. Bialek, and R.R.D.R. Van Steveninck. Entropy and information in neural spike trains: Progress on the sampling problem. *Physical Review E*, 69(5):056111, 2004.

B.N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.

G. Ouyang, C. Dang, D.A. Richards, and X. Li. Ordinal pattern based similarity analysis for eeg recordings. *Clinical Neurophysiology*, 121(5):694–703, 2010.

L. Paninski. Estimation of entropy and mutual information. *Neural Computation*, 15(6): 1191–1253, 2003.

L. Paninski. Estimating entropy on *m* bins given fewer than *m* samples. *IEEE Transactions on Information Theory*, 50(9):2200–2203, 2004.

I. Papageorgiou and I. Kontoyiannis. The posterior distribution of Bayesian Context-Tree models: Theory and applications. In *2022 IEEE International Symposium on Information Theory (ISIT)*, pages 702–707, 2022.

I. Papageorgiou and I. Kontoyiannis. Context-tree weighting for real-valued time series: Bayesian inference with hierarchical mixture models. In *2023 IEEE International Symposium on Information Theory (ISIT)*, 2023a.

I. Papageorgiou and I. Kontoyiannis. The Bayesian Context Trees State Space Model: Interpretable mixture models for time series. *In preparation*, 2023b.

I. Papageorgiou and I. Kontoyiannis. Posterior representations for Bayesian Context Trees: Sampling, estimation and convergence. *Bayesian Analysis, to appear, arXiv preprint arXiv:2202.02239*, 2023c.

I. Papageorgiou and I. Kontoyiannis. Truly Bayesian entropy estimation. In *2023 IEEE Information Theory Workshop (ITW)*, pages 497–502, 2023d.

I. Papageorgiou, I. Kontoyiannis, L. Mertzanis, A. Panotopoulou, and M. Skoularidou. Revisiting context-tree weighting for Bayesian inference. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 2906–2911, 2021.

I. Papageorgiou, V. Lungu, and I. Kontoyiannis. *BCT: Bayesian Context Trees for Discrete Time Series*, 2022. R package version 1.2. https://CRAN.R-project.org/package=BCT.

W. Philipp and W. Stout. *Almost sure invariance principles for partial sums of weakly dependent random variables*, volume 161. American Mathematical Society, 1975.

S.M. Potter. A nonlinear approach to US GNP. *Journal of applied econometrics*, 10(2): 109–125, 1995.

W.H. Quinn, V.T. Neal, and S.E. Antunez De Mayolo. El Niño occurrences over the past four and a half centuries. *Journal of Geophysical Research: Oceans*, 92(C13):14449–14461, 1987.

A.E. Raftery. A model for high-order Markov chains. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 47(3):528–539, 1985.

A.E. Raftery and S. Tavaré. Estimation and modelling repeated patterns in high order Markov chains with the mixture transition distribution model. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 43(1):179–199, 1994.

S.S. Rangapuram, M.W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski. Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31, 2018.

C. Rasmussen and Z. Ghahramani. Occam's razor. *Advances in Neural Information Processing Systems*, 13, 2000.

C.E. Rasmussen and C.K.I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

V.B. Reddy, B. Thimmappaya, R. Dhar, K.N. Subramanian, B.S. Zain, J. Pan, P.K. Ghosh, M.L. Celma, and S.M. Weissman. The genome of Simian virus 40. *Science*, 200(4341): 494–502, 1978.

D.J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286, 2014.

F. Rieke, D. Warland, R. de Ruyter van Steveninck, and W. Bialek. *Spikes: exploring the neural code*. MIT Press, 1999.

E. Rio. The functional law of the iterated logarithm for stationary strongly mixing sequences. *The Annals of Probability*, 23(3):1188–1203, 1995.

J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, 11(2):416–431, 1983a.

J. Rissanen. A universal data compression system. *IEEE Transactions on Information Theory*, 29(5):656–664, 1983b.

J. Rissanen. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information Theory*, 30(4):629–636, 1984.

J. Rissanen. Complexity of strings in the class of Markov sources. *IEEE Transactions on Information Theory*, 32(4):526–532, 1986a.

J. Rissanen. Stochastic complexity and modeling. *Annals of Statistics*, 14(3):1080–1100, 1986b.

J. Rissanen. Stochastic complexity. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 49(3):223–239, 1987.

J. Rissanen. *Stochastic complexity in statistical inquiry*. World Scientific, 1989.

C.P. Robert and G. Casella. *Monte Carlo statistical methods*. Springer, 2004.

C.P. Robert, N. Chopin, and J. Rousseau. Harold Jeffreys's theory of probability revisited. *Statistical Science*, 24(2):141–172, 2009.

S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984):20110550, 2013.

D. Ron, Y. Singer, and N. Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25(2-3):117–149, 1996.

P. Rothman. Forecasting asymmetric unemployment rates. *Review of Economics and Statistics*, 80(1):164–168, 1998.

J.C. Rotondo, E. Mazzoni, I. Bononi, M. Tognon, and F. Martini. Association between Simian virus 40 and human tumors. *Frontiers in Oncology*, 9:670, 2019.

V. Roy. Convergence diagnostics for Markov chain Monte Carlo. *Annual Review of Statistics and Its Application*, 7(1):387–412, 2020.

E. Sabeti, P.X.K. Song, and A.O. Hero. Pattern-based analysis of time series: Estimation. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 1236–1241, 2020.

D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3): 1181–1191, 2020.

J.A. Sanchez-Espigares and A. Lopez-Moreno. *MSwM: Fitting Markov Switching Models*, 2021. R package version 1.5. https://CRAN.R-project.org/package=MSwM.

F. Sanger, A.R. Coulson, G.F. Hong, D.F. Hill, and G.B. Petersen. Nucleotide sequence of bacteriophage λ DNA. *Journal of Molecular Biology*, 162(4):729–773, 1982.

A. Sarkar and D.B. Dunson. Bayesian nonparametric modeling of higher order Markov chains. *Journal of the American Statistical Association*, 111(516):1791–1803, 2016.

A.A. Saunders. The Song of the Wood Pewee. *The Auk*, 61(4):658–660, 1944.

A.O. Schmitt and H. Herzel. Estimating the entropy of DNA sequences. *Journal of theoretical biology*, 188(3):369–377, 1997.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

P.J. Schweitzer. Perturbation theory and finite Markov chains. *Journal of Applied Probability*, 5(2):401–413, 1968.

G.I. Shamir. On strongly sequential compression of sources with abrupt changes in statistics. In *2003 IEEE International Conference on Communications (ICC)*, volume 4, pages 2924–2927, 2003.

G.I. Shamir and D.J. Costello. Asymptotically optimal low-complexity sequential lossless coding for piecewise-stationary memoryless sources. I. The regular case. *IEEE Transactions on Information Theory*, 46(7):2444–2467, 2000.

G.I. Shamir and D.J. Costello. On the redundancy of universal lossless coding for general piecewise stationary sources. *Communications in Information and Systems*, 1(7):305–332, 2001.

G.I. Shamir and N. Merhav. Low-complexity sequential lossless coding for piecewise-stationary memoryless sources. *IEEE Transactions on Information Theory*, 45(5):1498–1519, 1999.

C.E. Shannon. Prediction and entropy of printed English. *Bell System Technical Journal*, 30 (1):50–64, 1951.

N. Shephard and T.G. Andersen. Stochastic volatility: origins and overview. In *Handbook of financial time series*, pages 233–254. Springer, 2009.

R. Shibata. Asymptotically efficient selection of the order of the model for estimating parameters of a linear process. *Annals of Statistics*, 8(1):147–164, 1980.

K. Shimada, S. Saito, and T. Matsushima. An efficient Bayes coding algorithm for the non-stationary source in which context tree model varies from interval to interval. In *2021 IEEE Information Theory Workshop (ITW)*, pages 1–6, 2021.

E. Simion. Entropy and randomness: From analogic to quantum world. *IEEE Access*, 8: 74553–74561, 2020.

S.P. Strong, R. Koberle, R.R.D.R. Van Steveninck, and W. Bialek. Entropy and information in neural spike trains. *Physical Review Letters*, 80(1):197, 1998.

J. Takeuchi and A.R. Barron. Stochastic complexity for tree models. In *2014 IEEE Workshop on Information Theory (ITW)*, pages 222–226, 2014.

L. Tierney. Markov chains for exploring posterior distributions. *Annals of Statistics*, 22(4): 1701–1728, 1994.

N.M. Timme and C. Lapish. A tutorial for information theory in neuroscience. *eNeuro*, 5(3): 1–40, 2018.

A. Timmermann, J.M. Oberhuber, A. Bacher, M. Esch, M. Latif, and E. Roeckner. Increased El Niño frequency in a climate model forced by future greenhouse warming. *Nature*, 398 (6729):694–697, 1999.

T.J. Tjalkens, F.M.J. Willems, and Y.M. Shtarkov. Multi-alphabet universal coding using a binary decomposition context tree weighting algorithm. In *15th Symposium on Information Theory in the Benelux*, 1994.

H. Tong. *Non-linear time series: a dynamical system approach*. Oxford University Press, 1990.

H. Tong. Threshold models in time series analysis—30 years on. *Statistics and its Interface*, 4(2):107–118, 2011.

H. Tong and K.S. Lim. Threshold autoregression, limit cycles and cyclical data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 42(3):245–268, 1980.

J.A. Totterdell, D. Nur, and K.L. Mengersen. Bayesian hidden Markov models in DNA sequence segmentation using R: The case of simian vacuolating virus (SV40). *Journal of Statistical Computation and Simulation*, 87(14):2799–2827, 2017.

K.E. Trenberth. The definition of El Niño. *Bulletin of the American Meteorological Society*, 78(12):2771–2778, 1997.

C. Truong, L. Oudre, and N. Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 176:107299, 2020.

R.S. Tsay. *Analysis of financial time series*, volume 543. John Wiley & Sons, 2005.

R. Turner, M. Deisenroth, and C.E. Rasmussen. State-space inference and learning with Gaussian processes. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 868–875, 2010.

R. D. Turner. *Gaussian processes for state space models and change point detection*. PhD thesis, University of Cambridge, 2012.

G.J.J. Van den Burg and C.K.I. Williams. An evaluation of change point detection algorithms. *arXiv preprint arXiv:2003.06222*, 2020.

A.W. Van der Vaart. *Asymptotic statistics*. Cambridge University Press, 2000.

J. Veness, K.S. Ng, M. Hutter, and M.H. Bowling. Context tree switching. In *2012 Data Compression Conference*, pages 327–0336, 2012.

J. Veness, M. White, M. Bowling, and A. György. Partition tree weighting. In *2013 Data Compression Conference*, pages 321–330, 2013.

S. Verdú. Empirical estimation of information measures: A literature guide. *Entropy*, 21(8): 720, 2019.

A. Verma and R.K. Bansal. Sequential change detection based on universal compression for Markov sources. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 2189–2193, 2019.

A. Virbickaite, M.C. Ausín, and P. Galeano. Bayesian inference methods for univariate and multivariate GARCH models: A survey. *Journal of Economic Surveys*, 29(1):76–96, 2015.

I.D. Vrontos, P. Dellaportas, and D.N. Politis. Full Bayesian inference for GARCH and EGARCH models. *Journal of Business & Economic Statistics*, 18(2):187–198, 2000.

I.D. Vrontos, P. Dellaportas, and D.N. Politis. A full-factor multivariate GARCH model. *The Econometrics Journal*, 6(2):312–334, 2003a.

I.D. Vrontos, P. Dellaportas, and D.N. Politis. Inference for some multivariate ARCH and GARCH models. *Journal of Forecasting*, 22(6-7):427–446, 2003b.

B. Wang, X. Luo, Y.M. Yang, W. Sun, M.A. Cane, W. Cai, S.W. Yeh, and J. Liu. Historical change of El Niño properties sheds light on future changes of extreme El Niño. *Proceedings of the National Academy of Sciences*, 116(45):22512–22517, 2019.

C.Z. Wei. On predictive least squares principles. *Annals of Statistics*, 20(1):1–42, 1992.

M.J. Weinberger, N. Merhav, and M. Feder. Optimal sequential probability assignment for individual sequences. *IEEE Transactions on Information Theory*, 40(2):384–396, 1994.

F.M.J. Willems. Coding for a binary independent piecewise-identically-distributed source. *IEEE Transactions on Information Theory*, 42(6):2210–2217, 1996.

F.M.J. Willems. The context-tree weighting method: Extensions. *IEEE Transactions on Information Theory*, 44(2):792–798, 1998.

F.M.J. Willems and P.A.J. Volf. Context maximizing: Finding MDL decision trees. In *15th Symposium on Information Theory in the Benelux*, 1994.

F.M.J. Willems and P.A.J. Volf. A study of the context tree maximizing method. In *16th Symposium on Information Theory in the Benelux*, 1995.

F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens. Context weighting: General finite context sources. In *14th Symposium on Information Theory in the Benelux*, 1993a.

F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens. Context tree weighting: Redundancy bounds and optimality. In *6th Joint Swedish-Russian International Workshop on Information Theory*, 1993b.

F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens. Context tree weighting: Basic properties. Unpublished manuscript. Available online at: `www.sps.tue.nl/wp-content/uploads/2015/09/ctw1submission.pdf`, 1993c.

F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens. The context tree weighting method: Basic properties. *IEEE Transactions on Information Theory*, 41(3):653–664, 1995.

F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens. Context tree maximizing. In *2000 Conference on Information Sciences and Systems*, 2000.

F.M.J. Willems, A. Nowbahkt-Irani, and P.A.J. Volf. Maximum a-posteriori probability tree models. In *4th International ITG Conference on Source and Channel Coding*, 2002.

R.M. Willems, S.L. Frank, A.D. Nijhof, P. Hagoort, and A. Van den Bosch. Prediction during natural language comprehension. *Cerebral Cortex*, 26(6):2506–2516, 2016.

C. S. Wong and W. K. Li. On a mixture autoregressive model. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(1):95–115, 2000.

C. S. Wong and W. K. Li. On a logistic mixture autoregressive model. *Biometrika*, 88(3): 833–846, 2001a.

C. S. Wong and W. K. Li. On a mixture autoregressive conditional heteroscedastic model. *Journal of the American Statistical Association*, 96(455):982–995, 2001b.

C. S. Wong, W. S. Chan, and P. L. Kam. A student t-mixture autoregressive model with applications to heavy-tailed financial data. *Biometrika*, 96(3):751–760, 2009.

S.N. Wood. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(1):3–36, 2011.

F. Wu, S. Zhao, B. Yu, et al. A new coronavirus associated with human respiratory disease in China. *Nature*, 579(7798):265–269, 2020.

A.D. Wyner and J. Ziv. Some asymptotic properties of the entropy of a stationary ergodic data source with applications to data compression. *IEEE Transactions on Information Theory*, 35(6):1250–1258, 1989.

Q. Xie and A.R. Barron. Asymptotic minimax regret for data compression, gambling, and prediction. *IEEE Transactions on Information Theory*, 46(2):431–445, 2000.

J. Xiong, V. Jääskinen, and J. Corander. Recursive learning for sparse Markov models. *Bayesian Analysis*, 11(1):247–263, 2016.

K. Yamanishi and S. Fukushima. Model change detection with the MDL principle. *IEEE Transactions on Information Theory*, 64(9):6115–6126, 2018.

R. Yan, Y. Zhang, Y. Li, L. Xia, Y. Guo, and Q. Zhou. Structural basis for the recognition of SARS-CoV-2 by full-length human ACE2. *Science*, 367(6485):1444–1448, 2020.

Y. Yang and D.B. Dunson. Bayesian conditional tensor factorizations for high-dimensional classification. *Journal of the American Statistical Association*, 111(514):656–669, 2016.

T.W. Yee. The VGAM package for categorical data analysis. *Journal of Statistical Software*, 32(10):1–34, 2010.

H.F. Yu, N. Rao, and I.S. Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. *Advances in neural information processing systems*, 29, 2016.

J.M. Zakoian. Threshold heteroskedastic models. *Journal of Economic Dynamics and control*, 18(5):931–955, 1994.

S.L. Zeger and K.Y. Liang. Longitudinal data analysis for discrete and continuous outcomes. *Biometrics*, 42(1):121–130, 1986.

G. Zhang, B.E. Patuwo, and M.Y. Hu. Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14(1):35–62, 1998.

O. Zhao and M. Woodroofe. Law of the iterated logarithm for stationary processes. *The Annals of Probability*, 36(1):127–142, 2008.

X. Zheng, M. Zaheer, A. Ahmed, Y. Wang, E.P. Xing, and A.J. Smola. State space LSTM models with particle MCMC inference. *arXiv preprint arXiv:1711.11179*, 2017.

X. Zheng, A. Kottas, and B. Sansó. On construction and estimation of stationary mixture transition distribution models. *Journal of Computational and Graphical Statistics*, 31(1): 283–293, 2022.

J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343, 1977.

W. Zucchini and I.L. MacDonald. *Hidden Markov models for time series: An introduction using R*. CRC Press, 2017.

Appendix A

The actual k-BCT algorithm

Here we describe a practical version of the k-BCT algorithm, which only differs from the idealised version of Section 2.3.3 in the initialisation step of the maximal probabilities and the position vectors at the leaves of depth $d < D$. After describing the algorithm, we also make some comments on its implementation complexity.

A.1 Algorithm description

(i) Initially, perform two pre-processing steps:

(ia) Execute the first two steps (i) and (ii) of the idealised k-BCT algorithm on the complete m-ary tree of depth D, with all the count vectors a_s assumed to be equal to zero, $a_s = (0,0,\ldots,0)$ for all s. Since all nodes at the same depth are identical, this process can be carried out effectively, by performing the relevant computations only at a single node (instead of all m^d nodes) for each depth $0 \leq d \leq D$.

(ib) Build the tree T_{MAX} from the contexts of x_{-D+1}^{n} and compute the count vectors a_s and the probabilities $P_{e,s} = P_e(a_s)$ at all nodes s of T_{MAX}, as in steps (i)–(iii) of the CTW algorithm.

(ii) Starting at the leaves and proceeding recursively towards the root, at each node s we compute a list of k maximal probabilities $P_{m,s}^{(i)}$ and k position vectors $c_s^{(i)} = (c_s^{(i)}(0), c_s^{(i)}(1), \ldots, c_s^{(i)}(m-1))$, for $i = 1,2,\ldots,k$, recursively as follows.

(iia) At each leaf s at depth D, with a nonzero count vector a_s, let $P_{m,s}^{(1)} = P_{e,s}$ and $c_s^{(1)} = (0,0,\ldots,0)$. For $i = 2,3,\ldots,k$, we leave $P^{(i)}$ and $c_s^{(i)}$ undefined.

(iib) Similarly, at each leaf s at depth D, with an all-zero count vector a_s, let $P_{m,s}^{(1)} = P_{e,s} = 1$, $c_s^{(1)} = (0,0,\ldots,0)$, and for $i = 2,3,\ldots,k$, leave $P^{(i)}$ and $c_s^{(i)}$ undefined.

(iic) At each leaf s at depth $d < D$, we let the list of the maximal probabilities $P_{m,s}^{(i)}$ and position vectors $c_s^{(i)}$ of s be those that are computed for a node at depth d in the preprocessing stage (ia).

(iid) Continue with steps (iib) and (iic) as in the idealised version of k-BCT.

(iii) We perform the same steps as described in ($iiia$)–($iiid$) of the idealised version, with the following addition:

($iiie$) While examining a node s at depth $0 < d < D$ in step ($iiib$) or ($iiic$) of the idealised algorithm, we may reach a point where the algorithm dictates that we examine its m children, when these children are *not* included in the tree T_{MAX}. In that case, we add them to T_{MAX}, and we define their corresponding maximal probabilities and position vectors according to the initialisation described in step (iib) or (iic) above, depending on whether $d = D-1$ or $d < D-1$, respectively.

(iv) As in the idealised version, output the k resulting trees T_i^* and the k maximal probabilities at the root, $P_{m,\lambda}^{(i)}$, $i = 1,2,\ldots,k$.

A.2 Computational complexity

As discussed in detail in Section 2.3.5, the complexity of the CTW and the BCT algorithms is $O(nmD)$. Following a similar argument, it is easy to see that the complexity of the k-BCT algorithm in its most naive implementation would be $O(nmD \times k^m)$. The factor k^m comes from step (iic) of the idealised algorithm, where for every internal node s, all possible combinations from the m lists of probabilities $P_{m,sj}$ are computed. However, the fact that these lists $P_{m,sj}$ are already ordered means that the top-k combinations can be found more efficiently, without performing an exhaustive search over all the k^m possible combinations. Instead, a best-first search can be employed to find only the top-k combinations: A priority queue can be used to maintain all possible candidates and efficiently pick the next best. This reduces the complexity to $O\left(nmD \times km\log(km)\right)$.

Appendix B

Proofs of theoretical results

In this section we provide complete proofs for all the theoretical results established in this thesis, for most of which only proof sketches were given in the main text.

B.1 Results of Chapter 2

Chapter 2 contains our main theoretical results for the setting of discrete-valued time series. In specific, these include the fact that the BCT prior $\pi_D(T;\beta)$ is a probability distribution on the model space $\mathcal{T}(D)$ (Lemma 2.1), together with the main results for the CTW, BCT and k-BCT algorithms (Theorems 2.1–2.3).

B.1.1 Proof of Lemma 2.1

Throughout this section, for the sake of clarity of notation, we simply write $\pi_D(T)$ for the model prior $\pi_D(T;\beta)$, without explicit reference to β.

The proof is by induction. Note first that for $D = 0, 1$ the result is trivial to check. Also observe that we can write any tree T which does not contain only the root node λ, as the union $T = \cup_j T_j$ of a collection of m subtrees $T_0, T_1, \ldots, T_{m-1}$. Clearly we will then have,

$$|T| = \sum_j |T_j|, \text{ and } L_D(T) = \sum_j L_{D-1}(T_j). \tag{B.1}$$

For the inductive step, suppose that the result holds for all depths less than or equal to some $d \leq D-1$. We will show that it holds for $d+1$ as well. Let $\Lambda = \{\lambda\}$ denote the empty

tree that consists only of the root node λ. Using (B.1), we have,

$$\begin{aligned}
\sum_{T\in\mathcal{T}(d+1)} \pi_{d+1}(T) &= \pi_{d+1}(\Lambda) + \sum_{T\in\mathcal{T}(d+1),T\neq\Lambda} \alpha^{|T|-1}\beta^{|T|-L_{d+1}(T)} \\
&= \beta + \sum_{T_0,T_1,\ldots,T_{m-1}\in\mathcal{T}(d)} \alpha^{\sum_j |T_j|-1}\beta^{\sum_j |T_j|-\sum_j L_d(T_j)} \qquad \text{(B.2)} \\
&= \beta + \alpha^{m-1} \sum_{T_0,T_1,\ldots,T_{m-1}\in\mathcal{T}(d)} \prod_j \alpha^{|T_j|-1}\beta^{|T_j|-L_d(T_j)} \\
&= \beta + \alpha^{m-1} \prod_j \sum_{T_j\in\mathcal{T}(d)} \pi_d(T_j) \\
&= \beta + \alpha^{m-1} = 1, \qquad \text{(B.3)}
\end{aligned}$$

where (B.2) follows by (B.1), and (B.3) follows from the inductive hypothesis and the assumption that $\beta = 1-\alpha^{m-1}$. □

B.1.2 Proof of Theorem 2.1

Before giving the proof of Theorem 2.1, we note for later use the following simple properties of the prior $\pi_D(T)$. Recall that we write $\Lambda = \{\lambda\}$ for the empty tree consisting of only the root node λ.

Lemma B.1. *(i) If $T \in \mathcal{T}(D)$, $T \neq \Lambda$, is expressed as the union $T = \cup_j T_j$ of the subtrees $T_j \in \mathcal{T}(D-1)$, then,*

$$\pi_D(T) = a^{m-1} \prod_{j=0}^{m-1} \pi_{D-1}(T_j), \qquad \text{(B.4)}$$

where $\alpha^{m-1} = 1-\beta$.

(ii) If $T \in \mathcal{T}(D)$, $t \in T$ is at depth $d < D$, $S \in \mathcal{T}(D-d)$ is nonempty, and $T \cup S$ consists of the tree T with S added as a subtree rooted at t, then,

$$\pi_D(T\cup S) = \beta^{-1}\pi_D(T)\pi_{D-d}(S).$$

Proof. The result of (i) immediately follows from the observation (B.1) in the proof of Lemma 2.1. Similarly, (ii) follows from the simple observations that $|T\cup S| = |T|+|S|-1$ and $L_D(T\cup S) = L_D(T)+L_{D-d}(S)$, together with the definition of the prior. □

For the proof of Theorem 2.1, we first note that, without loss of generality, we may assume that the tree T_{MAX} is the complete tree of depth D; if some node s of the complete tree is not in T_{MAX}, we simply assume that it has an all-zero count vector a_s.

The proof is again by induction. We adopt the notation of the proof of Lemma 2.1 and observe that, in view of Lemma 2.2, it suffices to show that,

$$P_{w,\lambda} = \sum_{T \in \mathcal{T}(D)} \pi_D(T) \prod_{s \in T} P_e(a_s). \tag{B.5}$$

We claim that the following more general statement holds: For any node s at depth d with $0 \leq d \leq D$, we have,

$$P_{w,s} = \sum_{U \in \mathcal{T}(D-d)} \pi_{D-d}(U) \prod_{u \in U} P_e(a_{su}), \tag{B.6}$$

where su denotes the concatenation of contexts s and u. Clearly (B.6) implies (B.5) upon taking $s = \lambda$, and (B.6) is trivially true for nodes s at depth D, since it reduces to the fact that $P_{w,s} = P_{e,s}$ for leaves s, by definition.

Suppose (B.6) holds for all nodes s at depth d for some fixed $0 < d \leq D$. Let s be a node at depth $d-1$; then, by the inductive hypothesis,

$$\begin{aligned} P_{w,s} &= \beta P_e(a_s) + (1-\beta) \prod_{j=0}^{m-1} P_{w,sj} \\ &= \beta P_e(a_s) + (1-\beta) \prod_{j=0}^{m-1} \left[\sum_{T_j \in \mathcal{T}(D-d)} \pi_{D-d}(T_j) \prod_{t \in T_j} P_e(a_{sjt}) \right], \end{aligned}$$

where sjt denotes the concatenation of context s, then symbol j, then context t, in that order. Therefore,

$$\begin{aligned} P_{w,s} &= \beta P_e(a_s) + (1-\beta) \sum_{T_0, T_1, \ldots, T_{m-1} \in \mathcal{T}(D-d)} \prod_{j=0}^{m-1} \left[\pi_{D-d}(T_j) \prod_{t \in T_j} P_e(a_{sjt}) \right] \\ &= \beta P_e(a_s) + \frac{1-\beta}{\alpha^{m-1}} \sum_{T_0, T_1, \ldots, T_{m-1} \in \mathcal{T}(D-d)} \pi_{D-d+1}(\cup_j T_j) \left[\prod_{j=0}^{m-1} \prod_{t \in T_j} P_e(a_{sjt}) \right], \end{aligned}$$

where for the last step we have used (B.4) from Lemma B.1.

Concatenating every symbol j with every leaf of the corresponding tree T_j, we end up with all the leaves of the larger tree $\cup_j T_j$. Therefore,

$$P_{w,s} = \beta P_e(a_s) + \frac{1-\beta}{\alpha^{m-1}} \sum_{T_0,T_1,\ldots,T_{m-1}\in\mathcal{T}(D-d)} \pi_{D-d+1}(\cup_j T_j) \prod_{t\in\cup_j T_j} P_e(a_{st}),$$

and since $1-\beta = \alpha^{m-1}$ and $\pi_d(\Lambda) = \beta$ for all $d \geq 1$,

$$\begin{aligned} P_{w,s} &= \pi_{D-d+1}(\Lambda)P_e(a_s) + \sum_{T_0,T_1,\ldots,T_{m-1}\in\mathcal{T}(D-d)} \pi_{D-d+1}(\cup_j T_j) \prod_{t\in\cup_j T_j} P_e(a_{st}) \\ &= \pi_{D-d+1}(\Lambda)P_e(a_s) + \sum_{T\in\mathcal{T}(D-d+1),T\neq\Lambda} \pi_{D-d+1}(T) \prod_{t\in T} P_e(a_{st}) \\ &= \sum_{T\in\mathcal{T}(D-d+1)} \pi_{D-d+1}(T) \prod_{t\in T} P_e(a_{st}). \end{aligned}$$

This establishes (B.6) for all nodes s at depth $d-1$, completing the inductive step and the proof of the theorem. □

B.1.3 Proof of Theorem 2.2

Again we observe that, without loss of generality, we may assume that the tree T_{MAX} is the complete tree of depth D; if some node s of the complete tree is not in T_{MAX}, we simply assume that it has an all-zero count vector a_s. It is easy to see that, for $\beta \geq 1/2$, these assumptions are equivalent to the ones in the description of the BCT algorithm, giving the same initial values to all leaves of T_{MAX}.

The proof is once again by induction, and we adopt the same notation as in the proofs of Lemma 2.1 and Theorem 2.1. First we will prove that,

$$P_{m,\lambda} = \max_{T\in\mathcal{T}(D)} P(x,T), \tag{B.7}$$

which, in view of Lemma 2.2, is equivalent to,

$$P_{m,\lambda} = \max_{T\in\mathcal{T}(D)} \pi_D(T) \prod_{s\in T} P_e(a_s). \tag{B.8}$$

As in the proof of Theorem 2.1, we claim that the following more general statement holds: For any node s at depth d with $0 \leq d \leq D$, we have,

$$P_{m,s} = \max_{U\in\mathcal{T}(D-d)} \pi_{D-d}(U) \prod_{u\in U} P_e(a_{su}), \tag{B.9}$$

where su denotes the concatenation of contexts s and u.

Taking $s = \lambda$ in (B.9) gives (B.8), and (B.9) is trivially true for nodes s at depth D, since it reduces to the fact that $P_{m,s} = P_{e,s}$ for leaves s, by definition.

For the inductive step, we assume that (B.9) holds for all nodes s at depth d for some fixed $0 < d \leq D$, and consider a node s at depth $d-1$. By the inductive hypothesis we have,

$$\begin{aligned}
P_{m,s} &= \max\left\{\beta P_e(a_s), (1-\beta)\prod_{j=0}^{m-1} P_{m,sj}\right\} \\
&= \max\left\{\beta P_e(a_s), (1-\beta)\prod_{j=0}^{m-1}\left[\max_{T_j\in\mathcal{T}(D-d)} \pi_{D-d}(T_j)\prod_{t\in T_j} P_e(a_{sjt})\right]\right\} \\
&= \max\left\{\beta P_e(a_s), (1-\beta)\max_{T_0,T_1,\ldots,T_{m-1}\in\mathcal{T}(D-d)}\prod_{j=0}^{m-1}\left[\pi_{D-d}(T_j)\prod_{t\in T_j} P_e(a_{sjt})\right]\right\} \\
&= \max\left\{\beta P_e(a_s), \frac{1-\beta}{\alpha^{m-1}}\max_{T_0,T_1,\ldots,T_{m-1}\in\mathcal{T}(D-d)} \pi_{D-d+1}(\cup_j T_j)\left[\prod_{j=0}^{m-1}\prod_{t\in T_j} P_e(a_{sjt})\right]\right\},
\end{aligned}$$

where the last step follows by (B.4) from Lemma B.1. Arguing as in the proof of Theorem 2.1,

$$\begin{aligned}
P_{m,s} &= \max\left\{\pi_{D-d+1}(\Lambda)P_e(a_s), \max_{T_0,T_1,\ldots,T_{m-1}\in\mathcal{T}(D-d)} \pi_{D-d+1}(\cup_j T_j)\prod_{t\in\cup_j T_j} P_e(a_{st})\right\} \\
&= \max\left\{\pi_{D-d+1}(\Lambda)P_e(a_s), \max_{T\in\mathcal{T}(D-d+1),T\neq\Lambda} \pi_{D-d+1}(T)\prod_{t\in T} P_e(a_{st})\right\} \\
&= \max_{T\in\mathcal{T}(D-d+1)} \pi_{D-d+1}(T)\prod_{t\in T} P_e(a_{st}).
\end{aligned}$$

This establishes (B.9) for all nodes s at depth $d-1$, completing the inductive step and hence also proving (B.7) and (B.8). To complete the proof of the theorem, it now suffices to show that,

$$P_{m,\lambda} = P(x, T_1^*), \tag{B.10}$$

because this is exactly (2.12), and combined with (B.7) it implies,

$$\max_{T\in\mathcal{T}(D)} P(x,T) = P(x,T_1^*),$$

which, after dividing both sides by the prior predictive likelihood, $P_D^*(x)$, gives (2.11).

By Lemma 2.2, (B.10) is equivalent to,

$$P_{m,\lambda} = \pi_D(T_1^*) \prod_{s \in T_1^*} P_e(a_s), \tag{B.11}$$

and, once again, we will establish the following more general statement: For any node s at depth d with $0 \leq d \leq D$, we have,

$$P_{m,s} = \max \left\{ \beta P_e(a_s), (1-\beta) \prod_{j=0}^{m-1} P_{m,sj} \right\} = \pi_{D-d}(T(s)) \prod_{t \in T(s)} P_e(a_{st}), \tag{B.12}$$

where $T(s)$ is the tree that the BCT algorithm would produce if it started its step (iii) at node s. Taking $s = \lambda$ in (B.12) gives (B.11), and (B.12) is again trivially true for leaves s at depth D, by the definition of the maximal probabilities $P_{m,s}$.

Finally, for the inductive step, suppose (B.12) holds for all nodes at depth $0 < d \leq D$, and let s be a node at depth $d-1$. We consider two separate cases: (i) If the maximum in (B.12) is achieved by the first term, then $P_{m,s} = \beta P_{e,s}$ and $T(s)$ consists of s only, so that (B.12) holds trivially; (ii) If the maximum in (B.12) is achieved by the second term, then $T(s) = \cup_j T(sj)$, and using (B.4) from Lemma B.1 and the inductive hypothesis, we obtain:

$$\begin{aligned}
P_{m,s} &= (1-\beta) \prod_{j=0}^{m-1} P_{m,sj} \\
&= (1-\beta) \prod_{j=0}^{m-1} \left[\pi_{D-d}(T(sj)) \prod_{t \in T(sj)} P_e(a_{sjt}) \right] \\
&= \pi_{D-d+1}(\cup_j T(sj)) \prod_{j=0}^{m-1} \prod_{t \in T(sj)} P_e(a_{sjt}) \\
&= \pi_{D-d+1}(\cup_j T(sj)) \prod_{t \in \cup_j T(sj)} P_e(a_{st}) = \pi_{D-d+1}(T(s)) \prod_{t \in T(s)} P_e(a_{st}).
\end{aligned}$$

This establishes (B.12) and completes the proof of the theorem. □

B.1.4 Proof of Theorem 2.3

Before giving the proof of Theorem 2.3, we observe that it is easy (though somewhat tedious) to verify that the two versions of the k-BCT algorithm are equivalent, in that they produce identical results as long as $\beta \geq 1/2$. Therefore, for the sake of simplicity, the proof below is given only for the idealised k-BCT algorithm.

The proof is again by induction, and we adopt the same notation as in the proof of Theorem 2.2. First we will prove that, for all $1 \leq i \leq k$,

$$P_{m,\lambda}^{(i)} = \max_{T \in \mathcal{T}(D)}^{(i)} P(x,T), \tag{B.13}$$

which, in view of Lemma 2.2, is equivalent to,

$$P_{m,\lambda}^{(i)} = \max_{T \in \mathcal{T}(D)}^{(i)} \pi_D(T) \prod_{s \in T} P_e(a_s). \tag{B.14}$$

As in the proof of Theorem 2.2, we claim that the following more general statement holds: For all i and any node s at depth d with $0 \leq d \leq D$, we have,

$$P_{m,s}^{(i)} = \max_{U \in \mathcal{T}(D-d)}^{(i)} \pi_{D-d}(U) \prod_{u \in U} P_e(a_{su}), \tag{B.15}$$

where su denotes the concatenation of contexts s and u. Taking $s = \lambda$ in (B.15) gives (B.14), and (B.15) is trivially true for nodes s at depth D, since at depth D we only consider $i = 1$ and then $P_{m,s}^{(1)} = P_{e,s}$ for leaves s, by definition.

For the inductive step, we assume that (B.15) holds for all i and all nodes s at depth d for some fixed $0 < d \leq D$, and consider a node s at depth $d-1$. By the inductive hypothesis we have, in the notation of (2.13), that $P_{m,s}^{(i)}$ is equal to,

$$\begin{aligned}
P_{m,s}^{(i)} &= \max^{(i)} \left[\bigcup_{i_0=1}^{k_0} \cdots \bigcup_{i_{m-1}=1}^{k_{m-1}} \left\{ (1-\beta) \prod_{j=0}^{m-1} P_{m,sj}^{(i_j)} \right\} \bigcup \{\beta P_e(a_s)\} \right] \\
&= \max^{(i)} \left[\bigcup_{i_0=1}^{k_0} \cdots \bigcup_{i_{m-1}=1}^{k_{m-1}} \left\{ (1-\beta) \prod_{j=0}^{m-1} \left[\max_{T_j \in \mathcal{T}(D-d)}^{(i_j)} \pi_{D-d}(T_j) \prod_{t \in T_j} P_e(a_{sjt}) \right] \right\} \bigcup \{\beta P_e(a_s)\} \right] \\
&= \max^{(i)} \left[\bigcup_{i_0=1}^{k_0} \cdots \bigcup_{i_{m-1}=1}^{k_{m-1}} \right. \\
&\qquad \left\{ (1-\beta) \max_{T_0 \in \mathcal{T}(D-d)}^{(i_0)} \cdots \max_{T_{m-1} \in \mathcal{T}(D-d)}^{(i_{m-1})} \prod_{j=0}^{m-1} \left[\pi_{D-d}(T_j) \prod_{t \in T_j} P_e(a_{sjt}) \right] \right\} \\
&\qquad \left. \bigcup \{\beta P_e(a_s)\} \right].
\end{aligned}$$

And using (B.4) from Lemma B.1 and arguing as before finally gives,

$$
\begin{aligned}
P_{m,s}^{(i)} = \max{}^{(i)} \Bigg[& \bigcup_{i_0=1}^{k_0} \cdots \bigcup_{i_{m-1}=1}^{k_{m-1}} \\
& \left\{ \max_{T_0 \in \mathcal{T}(D-d)}^{(i_0)} \cdots \max_{T_{m-1} \in \mathcal{T}(D-d)}^{(i_{m-1})} \pi_{D-d+1}(\cup_j T_j) \left[\prod_{j=0}^{m-1} \prod_{t \in T_j} P_e(a_{sjt}) \right] \right\} \\
& \bigcup \left\{ \beta P_e(a_s) \right\} \Bigg] \\
= \max{}^{(i)} \Bigg[& \bigcup_{i_0=1}^{k_0} \cdots \bigcup_{i_{m-1}=1}^{k_{m-1}} \left\{ \max_{T_0 \in \mathcal{T}(D-d)}^{(i_0)} \cdots \max_{T_{m-1} \in \mathcal{T}(D-d)}^{(i_{m-1})} \pi_{D-d+1}(\cup_j T_j) \prod_{t \in \cup_j T_j} P_e(a_{st}) \right\} \\
& \bigcup \left\{ \pi_{D-d+1}(\Lambda) P_e(a_s) \right\} \Bigg] \\
= \max{}^{(i)} \Bigg[& \bigcup_{i'=1}^{k} \left\{ \max_{T \in \mathcal{T}(D-d+1), T \neq \Lambda}^{(i')} \pi_{D-d+1}(T) \prod_{t \in T} P_e(a_{st}) \right\} \bigcup \left\{ \pi_{D-d+1}(\Lambda) P_e(a_s) \right\} \Bigg] \\
= & \max_{T \in \mathcal{T}(D-d+1)}^{(i)} \pi_{D-d+1}(T) \prod_{t \in T} P_e(a_{st}).
\end{aligned}
$$

This establishes (B.15) for all i and all nodes s at depth $d-1$, completing the inductive step and hence also proving (B.13) and (B.14). To complete the proof of the theorem, it now suffices to show that, for all $1 \leq i \leq k$,

$$
P_{m,\lambda}^{(i)} = P(x, T_i^*), \tag{B.16}
$$

because, combined with (B.13) this implies,

$$
\max_{T \in \mathcal{T}(D)}^{(i)} P(x, T) = P(x, T_i^*),
$$

which, after dividing both sides by the prior predictive likelihood, gives (2.14). In view of Lemma 2.2, (B.16) is equivalent to,

$$
P_{m,\lambda}^{(i)} = \pi_D(T_i^*) \prod_{s \in T_i^*} P_e(a_{st}), \tag{B.17}
$$

and, once again, we will establish the following more general statement.

For all i and any node s at depth d with $0 \leq d \leq D$, we have,

$$P_{m,s}^{(i)} = \max^{(i)} \left[\bigcup_{i_0=1}^{k_0} \bigcup_{i_1=1}^{k_1} \cdots \bigcup_{i_{m-1}=1}^{k_{m-1}} \left\{ (1-\beta) \prod_{j=0}^{m-1} P_{m,sj}^{(i_j)} \right\} \bigcup \left\{ \beta P_e(a_s) \right\} \right] \quad \text{(B.18)}$$

$$= \pi_{D-d}(T^{(i)}(s)) \prod_{t \in T^{(i)}(s)} P_e(a_{st}), \quad \text{(B.19)}$$

where $T^{(i)}(s)$ is the ith tree that $k-BCT$ would produce if it started its step (iii) at node s. Taking $s = \lambda$ in (B.19) gives (B.17), and (B.19) is again trivially true for leaves s at depth D, by the definition of the maximal probabilities $P_{m,s}$.

Finally, for the inductive step assume (B.19) holds for all nodes at depth $0 < d \leq D$, and let s be a node at at depth $d-1$. We consider two separate cases: (i) If the maximum in (B.18) is achieved by the last term, then $P_{m,s}^{(i)} = \beta P_{e,s}$ and $T^{(i)}(s)$ consists of s only, so that (B.19) holds trivially; (ii) If the maximum in (B.18) is achieved by the collection of indices $(i_0, i_1, \ldots, i_{m-1})$, then $T^{(i)}(s) = \cup_j T^{(i_j)}(sj)$, and using (B.4) from Lemma B.1 and the inductive hypothesis, we obtain:

$$\begin{aligned}
P_{m,s}^{(i)} &= (1-\beta) \prod_{j=0}^{m-1} P_{m,sj}^{(i_j)}(a_{sj}) \\
&= (1-\beta) \prod_{j=0}^{m-1} \left[\pi_{D-d}(T^{(i_j)}(sj)) \prod_{t \in T^{(i_j)}(sj)} P_e(a_{sjt}) \right] \\
&= \pi_{D-d+1}(\cup_j T^{(i_j)}(sj)) \prod_{j=0}^{m-1} \prod_{t \in T^{(i_j)}(sj)} P_e(a_{sjt}) \\
&= \pi_{D-d+1}(\cup_j T^{(i_j)}(sj)) \prod_{t \in \cup_j T^{(i_j)}(sj)} P_e(a_{st}) \\
&= \pi_{D-d+1}(T^{(i)}(s)) \prod_{t \in T^{(i)}(s)} P_e(a_{st}).
\end{aligned}$$

This establishes (B.19) and completes the proof of the theorem. □

B.1.5 Arbitrary Dirichlet parameters

In some cases it may be desirable to consider a more general Dirichlet prior $\pi(\theta|T)$ on the parameters θ associated with a given model $T \in \mathcal{T}(D)$. Instead of the $\mathrm{Dir}(1/2, 1/2, \ldots, 1/2)$ distribution defined in (2.5) we may place a different, general $\mathrm{Dir}(\gamma_s(0), \gamma_s(1), \ldots, \gamma_s(m-1))$ prior on each context $s \in T$.

For an arbitrary collection of hyperparameters $\gamma = \{\gamma_s = (\gamma_s(0), \ldots, \gamma_s(m-1)) \,;\, s \in T\}$, with each $\gamma_s(j) > 0$, let $\pi(\theta|T) = \prod_{s\in T} \pi(\theta_s)$, with,

$$\pi(\theta_s) = \pi(\theta_s(0), \theta_s(1), \ldots, \theta_s(m-1)) = \frac{\Gamma(M'_s)}{\prod_{j=0}^{m-1} \Gamma(\gamma_s(j))} \prod_{j=0}^{m-1} \theta_s(j)^{\gamma_s(j)-1} \propto \prod_{j=0}^{m-1} \theta_s(j)^{\gamma_s(j)-1},$$

where $M'_s := \sum_{j=0}^{m-1} \gamma_s(j)$, $s \in T$.

As in Section 2.2.2, the marginal likelihood $P(x|T)$ of a data string x given a model $T \in \mathcal{T}(D)$ can be computed explicitly. Lemma B.2 below is the analog of Lemma 2.2 in this case; its proof is a straightforward modification of that of Lemma 2.2.

Lemma B.2. *The marginal likelihood $P(x|T)$ of the observations x given a model T is,*

$$P(x|T) = \int P(x, \theta|T) d\theta = \int P(x|\theta, T) \pi(\theta|T) d\theta = \prod_{s\in T} P_e(a_s, \gamma_s),$$

where the count vectors $a_s = (a_s(0), a_s(1), \ldots, a_s(m-1))$ are defined in (2.3) *as before and the* estimated probabilities *$P_e(a_s, \gamma_s)$ are now defined by,*

$$P_e(a_s, \gamma_s) := \frac{\Gamma(M'_s)}{\Gamma(M_s + M'_s)} \prod_{j=0}^{m-1} \frac{\Gamma(a_s(j) + \gamma_s(j))}{\Gamma(\gamma_s(j))}, \tag{B.20}$$

where $M_s := a_s(0) + a_s(1) + \cdots + a_s(m-1)$ and $M'_s := \gamma_s(0) + \gamma_s(1) + \cdots + \gamma_s(m-1)$, again with the convention that any empty product is taken to be equal to 1.

Now, it is possible to modify the CTW, BCT and k-BCT algorithms, simply by replacing the estimated probabilities $P_e(a_s)$ of (2.7) by $P_e(a_s, \gamma_s)$ as in (B.20). A careful inspection of the proofs of Theorems 2.1, 2.2 and 2.3 shows that their results remain valid in this more general case by replacing $P_e(a_s)$ with $P_e(a_s, \gamma_s)$ as in (B.20).

Similarly, all the additional results in Chapter 2 remain valid as stated there, with the exception of the expression for the full conditional density in (2.16). In the case of a general prior $\pi(\theta|T)$ in terms of the hyperparameters γ, the same computation shows that here:

$$\pi(\theta|T, x) = \prod_{s\in T} \mathrm{Dir}\big(a_s(0) + \gamma_s(0), \ldots, a_s(m-1) + \gamma_s(m-1)\big).$$

Finally, we note that sequential updates can be carried out for the CTW, BCT, and k-BCT algorithms in this setting as before; see Section 2.3.5. The only change is in the update needed for $P_e(a_s, \gamma_s)$, which should now be updated by multiplying its earlier value by $(a_s(j) + \gamma_s(j) - 1)/(M'_s + M_s - 1)$, using the updated values of a_s and M_s.

B.2 Results of Chapter 3

Chapter 3 contains theoretical results connected with the branching process representation of the BCT posterior on model space. The equivalence of this representation is established in Proposition 3.2, and is later used to provide asymptotic results that further justify the BCT framework. In specific, these results include the asymptotic concentration of the model posterior on the 'true' underlying model (Theorem 3.1), together with an explicit almost sure rate for this convergence (Theorem 3.2), and with corresponding results when the data are generated from general stationary ergodic processes (Theorems 3.3 and 3.4).

B.2.1 Proof of Proposition 3.2

The proof parallels that of Proposition 3.1 for the BCT prior, which is included in the main text. When $D = 0$, $\mathcal{T}(D)$ consists of a single tree, $T = \{\lambda\}$, which has probability 1 under both the BCT posterior $\pi(\cdot|x)$ and under the distribution $\pi_b(\cdot)$ induced by the branching process construction. Suppose $D \geq 1$.

As before, we view every tree $T \in \mathcal{T}(D)$ as a collection of k of m-branches, and we proceed by induction on k. For $k = 0$, i.e., for $T = \{\lambda\}$, by the definitions,

$$\pi_b(\{\lambda\}) = P_{b,\lambda} = \frac{\beta P_{e,\lambda}}{P_{w,\lambda}} = \pi_D(\{\lambda\};\beta)\frac{P_{e,\lambda}}{P_{w,\lambda}},$$

and using Lemma 2.2 and the fact that $P_{w,\lambda}$ is exactly the normalising constant $P(x)$,

$$\pi_b(\{\lambda\}) = \frac{\pi_D(\{\lambda\};\beta)P(x|\{\lambda\})}{P(x)} = \pi(\{\lambda\}|x),$$

completing the proof for the base case $k = 0$.

Now assume the result of the proposition holds for all trees with k m-branches, and suppose $T' \in \mathcal{T}(D)$ contains $(k+1)$ m-branches and is obtained from some $T \in \mathcal{T}(D)$ by adding a single m-branch to one of its leaves, s. Again, we consider two cases.

(i) If s is at depth $D-2$ or smaller, then by construction,

$$\pi_b(T') = \frac{\pi_b(T)}{P_{b,s}}(1 - P_{b,s})\prod_{j=0}^{m-1} P_{b,sj},$$

and therefore, using the inductive hypothesis,

$$\pi_b(T') = \pi(T|x)\Big(\frac{1-P_{b,s}}{P_{b,s}}\Big)\prod_j P_{b,sj} = \pi(T'|x)\frac{\pi(T|x)}{\pi(T'|x)}\Big(\frac{1-P_{b,s}}{P_{b,s}}\Big)\prod_j P_{b,sj}. \tag{B.21}$$

Using the definitions of π_D and $P_{b,s}$ together with Lemma 2.2, we can express the posterior odds $\pi(T'|x)/\pi(T|x)$ in (B.21) as,

$$\begin{aligned}\frac{\pi(T'|x)}{\pi(T|x)} = \frac{\pi_D(T';\beta)}{\pi_D(T;\beta)}\frac{P(x|T')}{P(x|T)} =& \frac{\pi_D(T';\beta)}{\pi_D(T;\beta)}\frac{\prod_{j=0}^{m-1}P_{e,sj}}{P_{e,s}} \\ =& \frac{\beta^m(1-\beta)}{\beta}\frac{\prod_j P_{e,sj}}{P_{e,s}} \\ =& (1-P_{b,s})\frac{1-\beta}{\beta P_{e,s}(1-P_{b,s})}\prod_j \beta P_{e,sj} \\ =& (1-P_{b,s})\frac{P_{w,s}}{\beta P_{e,s}}\frac{1-\beta}{(P_{w,s}-\beta P_{e,s})}\prod_j \beta P_{e,sj},\end{aligned}$$

and from the definitions of $P_{w,s}$ and $P_{b,s}$ in (2.9) and (3.1) we obtain,

$$\frac{\pi(T'|x)}{\pi(T|x)} = (1-P_{b,s})\frac{1}{P_{b,s}}\frac{1}{\prod_j P_{w,sj}}\prod_j \beta P_{e,sj} = \Big(\frac{1-P_{b,s}}{P_{b,s}}\Big)\prod_j P_{b,sj}. \tag{B.22}$$

Substituting (B.22) into (B.21) yields, $\pi_b(T') = \pi(T'|x)$, as claimed.

(ii) Similarly, if s is at depth $D-1$, from the inductive hypothesis,

$$\pi_b(T') = \frac{\pi_b(T)}{P_{b,s}}(1-P_{b,s}) = \pi(T|x)\Big(\frac{1-P_{b,s}}{P_{b,s}}\Big) = \pi(T'|x)\frac{\pi(T|x)}{\pi(T'|x)}\Big(\frac{1-P_{b,s}}{P_{b,s}}\Big), \tag{B.23}$$

where the posterior odds can be expressed as,

$$\begin{aligned}\frac{\pi(T'|x)}{\pi(T|x)} =& \frac{\pi_D(T';\beta)}{\pi_D(T;\beta)}\frac{\prod_{j=0}^{m-1}P_{e,sj}}{P_{e,s}} \\ =& \Big(\frac{1-\beta}{\beta}\Big)\frac{\prod_j P_{w,sj}}{P_{e,s}} \\ =& \frac{P_{w,s}-\beta P_{e,s}}{\beta P_{e,s}} = \frac{1-P_{b,s}}{P_{b,s}},\end{aligned} \tag{B.24}$$

where in the second equality we used that $P_{w,sj} = P_{e,sj}$, as all nodes sj are at depth $d = D$ in this case. Substituting (B.24) above in (B.23) yields $\pi_b(T') = \pi(T'|x)$, and completes the proof of the proposition. □

B.2.2 Proof of Lemma 3.2

Lemma 3.2 considers the asymptotic behaviour of the branching probabilities $P_{b,s}$ for internal nodes s of T^*, which is the first step in proving Theorem 3.1. Here we establish the two missing steps in the proof of Lemma 3.2 given in Section 3.2.1 of the main text.

Proof of (3.8). Using the upper bound of Lemma 3.1 for a fixed context s, and the corresponding lower bound for the context sj, we obtain the upper bound,

$$\frac{1}{M_s}\left(\log P_{e,s} - \sum_{j=0}^{m-1} \log P_{e,sj}\right) \leq \sum_{i=0}^{m-1} \frac{a_s(i)}{M_s} \log \frac{a_s(i)}{M_s} - \sum_{j=0}^{m-1} \frac{M_{sj}}{M_s} \sum_{i=0}^{m-1} \frac{a_{sj}(i)}{M_{sj}} \log \frac{a_{sj}(i)}{M_{sj}} + \frac{m-1}{2M_s}\left(\sum_{j=0}^{m-1} \log M_{sj} - \log M_s\right) + \frac{C}{M_s}, \tag{B.25}$$

for some constant C. Since M_s and M_{sj} both tend to infinity a.s. as $n \to \infty$ by positive-ergodicity, the last two terms above both vanish a.s.

For the first two terms, we first note that, by the ergodic theorem for Markov chains (e.g., (Chung, 1967, p. 92)),

$$\frac{a_s(i)}{M_s} = \frac{a_s(i)}{n}\,\frac{n}{M_s} \to \frac{\pi(si)}{\pi(s)} = \pi(i|s), \quad \text{a.s.}, \tag{B.26}$$

where for the stationary distribution π, the notation we use is that si denotes the concatenation of context s followed by symbol i moving 'forward' in time.

Recalling the definition of X and J, we have,

$$\mathbb{P}(X = i|s) = \frac{\pi(si)}{\pi(s)} = \pi(i|s), \quad \mathbb{P}(J = j|s) = \frac{\pi(js)}{\pi(s)}, \tag{B.27}$$

so that for the first term of (B.25), as $n \to \infty$,

$$\sum_{i=0}^{m-1} \frac{a_s(i)}{M_s} \log \frac{a_s(i)}{M_s} \to -H(X|s), \quad \text{a.s.} \tag{B.28}$$

Similarly, for the second term of (B.25), from the ergodic theorem,

$$\frac{a_{sj}(i)}{M_{sj}} = \frac{a_{sj}(i)}{n}\,\frac{n}{M_{sj}} \to \frac{\pi(jsi)}{\pi(js)} = \pi(i|js) = \mathbb{P}(X = i|s, J = j), \quad \text{a.s.}, \tag{B.29}$$

$$\frac{M_{sj}}{M_s} = \frac{M_{sj}}{n}\,\frac{n}{M_s} \to \frac{\pi(js)}{\pi(s)} = \mathbb{P}(J = j|s), \quad \text{a.s.}, \tag{B.30}$$

so that, as $n \to \infty$,

$$-\sum_{j=0}^{m-1} \frac{M_{sj}}{M_s} \sum_{i=0}^{m-1} \frac{a_{sj}(i)}{M_{sj}} \log \frac{a_{sj}(i)}{M_{sj}} \to \sum_{j=0}^{m-1} \mathbb{P}(J=j|s) H(X|s,J=j) = H(X|s,J), \tag{B.31}$$

by the definition of conditional entropy. Finally, combining with (B.28), we get,

$$\sum_{i=0}^{m-1} \frac{a_s(i)}{M_s} \log \frac{a_s(i)}{M_s} - \sum_{j=0}^{m-1} \frac{M_{sj}}{M_s} \sum_{i=0}^{m-1} \frac{a_{sj}(i)}{M_{sj}} \log \frac{a_{sj}(i)}{M_{sj}} \to -I(X;J|s), \quad \text{a.s.} \tag{B.32}$$

Following the same sequence of steps, we can similarly obtain a lower bound corresponding to (B.25) as,

$$\begin{aligned} \frac{1}{M_s}\left(\log P_{e,s} - \sum_{j=0}^{m-1} \log P_{e,sj}\right) \geq \sum_{i=0}^{m-1} \frac{a_s(i)}{M_s} \log \frac{a_s(i)}{M_s} - \sum_{j=0}^{m-1} \frac{M_{sj}}{M_s} \sum_{i=0}^{m-1} \frac{a_{sj}(i)}{M_{sj}} \log \frac{a_{sj}(i)}{M_{sj}} \\ + \frac{m-1}{2M_s}\left(\sum_{j=0}^{m-1} \log M_{sj} - \log M_s\right) + \frac{C'}{M_s}, \end{aligned} \tag{B.33}$$

where the only difference from (B.25) is the constant C'. Therefore,

$$\frac{1}{M_s}\left(\log P_{e,s} - \sum_{j=0}^{m-1} \log P_{e,sj}\right) \to -I(X;J|s), \quad \text{a.s.},$$

or, equivalently,

$$\log P_{e,s} - \sum_{j=0}^{m-1} \log P_{e,sj} = -M_s I(X;J|s) + o(M_s), \quad \text{a.s.} \tag{B.34}$$

And since $M_s = n\pi(s) + o(1)$ a.s. by the ergodic theorem, we finally get (3.8). □

Proof of final step in Lemma 3.1. As already noted in the main text, (3.8) implies $P_{b,s} \to 0$ a.s. for nodes whose children are leaves of T^*. The same holds for all internal nodes s of T^* for which $I(X;J|s) > 0$. The only remaining case is that of internal nodes u for which $I(X;J|u) = 0$. The fact that again $P_{b,u} \to 0$ a.s. is an immediate consequence of the result given as Lemma B.3 in Appendix B.2. □

B.2.3 Proof of Lemma 3.3

Lemma 3.3 now considers the asymptotic behaviour of the branching probabilities $P_{b,s}$ for the leaves s of T^*, which is the second step in proving Theorem 3.1. Here we establish the two missing steps in the proof of Lemma 3.3 given in Section 3.2.1 of the main text.

Proof of (3.9). We can rewrite (B.33) as,

$$\sum_{j=0}^{m-1} \log P_{e,sj} - \log P_{e,s} \leq \sum_{j=0}^{m-1}\sum_{i=0}^{m-1} a_{sj}(i) \log \frac{a_{sj}(i)}{M_{sj}} - \sum_{i=0}^{m-1} a_s(i) \log \frac{a_s(i)}{M_s} - \frac{m-1}{2}\left(\sum_{j=0}^{m-1} \log M_{sj} - \log M_s\right) - C'. \tag{B.35}$$

We write $\widehat{p}_s$ and π_s for the empirical and stationary conditional distributions of the symbol following s,

$$\widehat{p}_s(i) := \frac{a_s(i)}{M_s}, \quad \pi_s(i) := \pi(i|s) = \frac{\pi(si)}{\pi(s)}, \qquad i \in A.$$

Let $D(p\|q)$ denote the relative entropy (or Kullback-Leibler divergence) between two probability mass functions p, q on the same discrete alphabet (Cover and Thomas, 1999, Ch. 2). By the nonnegativity of the relative entropy we have,

$$\sum_{i=0}^{m-1} a_s(i) \log \pi_s(i) - \sum_{i=0}^{m-1} a_s(i) \log \frac{a_s(i)}{M_s} = M_s \sum_{i=0}^{m-1} \widehat{p}_s(i) \log \frac{\pi_s(i)}{\widehat{p}_s(i)} = -M_s D(\widehat{p}_s\|\pi_s) \leq 0.$$

Adding and subtracting the term $\sum_{i=0}^{m-1} a_s(i) \log \pi_s(i)$ to (B.35) and using the last inequality,

$$\sum_{j=0}^{m-1} \log P_{e,sj} - \log P_{e,s} \leq \sum_{j=0}^{m-1}\sum_{i=0}^{m-1} a_{sj}(i) \log \widehat{p}_{sj}(i) - \sum_{i=0}^{m-1} a_s(i) \log \pi_s(i) - \frac{m-1}{2}\left(\sum_{j=0}^{m-1} \log M_{sj} - \log M_s\right) - C'. \tag{B.36}$$

We examine the first and second terms in (B.36) separately. For the first (and main) term, since for all count vectors, $a_s(i) = \sum_{j=0}^{m-1} a_{sj}(i)$, we can express,

$$\begin{aligned}
\sum_{j=0}^{m-1}\sum_{i=0}^{m-1} a_{sj}(i) \log \widehat{p}_{sj}(i) - \sum_{i=0}^{m-1} a_s(i) \log \pi_s(i) &= \sum_{j=0}^{m-1}\sum_{i=0}^{m-1} a_{sj}(i) \log \frac{\widehat{p}_{sj}(i)}{\pi_s(i)} \\
&= \sum_{j=0}^{m-1} M_{sj} \sum_{i=0}^{m-1} \widehat{p}_{sj}(i) \log \frac{\widehat{p}_{sj}(i)}{\pi_s(i)} \\
&= \sum_{j=0}^{m-1} M_{sj} D(\widehat{p}_{sj}\|\pi_s) \\
&= \sum_{j=0}^{m-1} M_{sj} D(\widehat{p}_{sj}\|\pi_{sj}),
\end{aligned}$$

where the last equality holds because s is either a leaf or an external nodes of T^*, so that $I(X;J|s) = 0$ and $\pi_{sj} = \pi_{sj'} = \pi_s$, for all j, j'.

In order to bound the relative entropy between the empirical and the stationary conditional distributions, we first recall that the relative entropy is bounded above by the χ^2-distance, e.g., (Gibbs and Su, 2002),

$$D(\widehat{p}_{sj}\|\pi_{sj}) \leq d_{\chi^2}(\widehat{p}_{sj},\pi_{sj}) = \sum_{i=0}^{m-1} \frac{(\widehat{p}_{sj}(i)-\pi_{sj}(i))^2}{\pi_{sj}(i)}. \tag{B.37}$$

From the law of the iterated logarithm (LIL) for Markov chains (Chung, 1967, p. 106), we have, a.s. as $n \to \infty$,

$$a_{sj}(i) = n\pi(jsi) + O(\sqrt{n\log\log n}), \quad M_{sj} = n\pi(js) + O(\sqrt{n\log\log n}), \tag{B.38}$$

so that,

$$\widehat{p}_{sj}(i) = \frac{a_{sj}(i)}{M_{sj}} = \frac{\pi(jsi)}{\pi(js)} + O\left(\sqrt{\frac{\log\log n}{n}}\right) = \pi_{sj}(i) + O\left(\sqrt{\frac{\log\log n}{n}}\right).$$

Substituting in (B.37) yields,

$$D(\widehat{p}_{sj}\|\pi_{sj}) \leq O\left(\frac{\log\log n}{n}\right)\sum_{i=0}^{m-1}\frac{1}{\pi_{sj}(i)} = O\left(\frac{\log\log n}{n}\right), \quad \text{a.s.}, \tag{B.39}$$

and finally, using (B.38) again,

$$\sum_{j=0}^{m-1} M_{sj} D(\widehat{p}_{sj}\|\pi_{sj}) = O(\log\log n), \quad \text{a.s.} \tag{B.40}$$

For the third term in (B.36),

$$\sum_{j=0}^{m-1}\log M_{sj} - \log M_s = \sum_{j=0}^{m-1}\log\frac{M_{sj}}{M_s} + (m-1)\log M_s. \tag{B.41}$$

Using the LIL, $M_s = n\pi(s) + O(\sqrt{n\log\log n})$, a.s., so,

$$\log M_s = \log\left(n\pi(s)\left\{1 + O(\sqrt{\log\log n/n})\right\}\right) = \log n + O(1), \quad \text{a.s.},$$

and using the LIL again as above, $M_{sj}/M_s = \pi(js)/\pi(s) + O(\sqrt{\log\log n/n}) = O(1)$, a.s., so that, $\log M_{sj}/M_s = O(1)$, a.s.

So, we finally get that,

$$\sum_{j=0}^{m-1} \log M_{sj} - \log M_s = (m-1)\log n + O(1), \quad \text{a.s.}, \tag{B.42}$$

which together with (B.36) and (B.40) complete the proof of (3.9). □

Proof of final step in Lemma 3.3. As discussed in the proof of Lemma 3.3 in the main text, the asymptotic relation (3.9) implies that $\prod_j P_{e,sj}/P_{e,s} \to 0$, a.s., for all leaves and external nodes s of T^*. The next proposition states that it also implies that $\prod_j P_{w,sj}/P_{e,s} \to 0$, so that $P_{b,s} \to 1$ a.s. by (3.6), completing the proof of Lemma 3.3. Note that it suffices to consider leaves at depths $d \leq D-1$, since for leaves s at depth D we already have $P_{b,s} = 1$.

Proposition B.1. *Under the assumptions of Theorem 3.1, for all leaves and external nodes s of T^* at depths $d \leq D-1$ we have, as $n \to \infty$:*

$$\frac{\prod_{j=0}^{m-1} P_{w,sj}}{P_{e,s}} \to 0, \quad \textit{a.s.}$$

Proof. Note that, since the stationary distribution is positive on all finite contexts, the tree T_{MAX} is eventually a.s. the complete tree of depth D, so we need not consider special cases of contexts s that do not appear in the data separately. Let s be a leaf or external node of T^* at depth $0 \leq d \leq D-1$. The proof is by induction on d.

For $d = D-1$, the claim is satisfied trivially as $P_{w,sj} = P_{e,sj}$, since nodes sj are at depth D. For the inductive step, we assume that the claim holds for all leaves and external nodes s of T^* at some depth $d \leq D-1$, and consider a leaf or external node s of T^* at depth $d-1$. We have, as $n \to \infty$,

$$\begin{aligned}
\frac{\prod_{j=0}^{m-1} P_{w,sj}}{P_{e,s}} &= \frac{\prod_{j=0}^{m-1} \left[\beta P_{e,sj} + (1-\beta)\prod_{t=0}^{m-1} P_{w,sjt}\right]}{P_{e,s}} \\
&= \frac{\prod_{j=0}^{m-1} \left[\beta P_{e,sj} + (1-\beta)\prod_{t=0}^{m-1} P_{w,sjt}\right]}{\prod_{j=0}^{m-1} \beta P_{e,sj}} \, \frac{\prod_{j=0}^{m-1} \beta P_{e,sj}}{P_{e,s}} \\
&= \prod_{j=0}^{m-1} \left(1 + \frac{1-\beta}{\beta} \frac{\prod_{t=0}^{m-1} P_{w,sjt}}{P_{e,sj}}\right) \, \frac{\prod_{j=0}^{m-1} P_{e,sj}}{P_{e,s}} \beta^m \to 0, \quad \text{a.s.},
\end{aligned} \tag{B.43}$$

as $\prod_{t=0}^{m-1} P_{w,sjt}/P_{e,sj} \to 0$ by the inductive hypothesis, and $\prod_{j=0}^{m-1} P_{e,sj}/P_{e,s} \to 0$ for a node s which is either a leaf or external node of T^*. This establishes the inductive step and completes the proof of the proposition. □

B.2.4 Proof of Theorem 3.2

The starting point of the proof is the representation of the posterior given in equation (3.2) of the main text. In particular, we examine the asymptotic behaviour of the branching probabilities $P_{b,s}$ separately for leaves and internal nodes.

Leaves. Let s be a leaf or an external node of T^*. We already have a strong upper bound for the estimated probabilities of s in (3.9). Write $r = (m-1)^2/2$. Since the bound (3.9) holds a.s., a straightforward sample-path-wise computation immediately implies that, for all $\varepsilon > 0$,

$$\frac{\prod_{j=0}^{m-1} P_{e,sj}}{P_{e,s}} = O\big(n^{-r+\varepsilon}\big), \quad \text{a.s.} \tag{B.44}$$

Proposition B.2 states that $\prod_j P_{w,sj}/P_{e,s}$ has the same asymptotic behaviour.

Proposition B.2. *For all leaves and external nodes s of T^* at depths $d \leq D-1$, for any $\varepsilon > 0$ we have, as $n \to \infty$:*

$$\frac{\prod_{j=0}^{m-1} P_{w,sj}}{P_{e,s}} = O\left(n^{-\frac{(m-1)^2}{2}+\varepsilon}\right), \quad \textit{a.s.}$$

Proof. The proof is similar to that of Proposition B.1, by induction on d. If $d = D-1$, then $P_{w,sj} = P_{e,sj}$ and the claim follows from (B.44). For the inductive step, assume the claim holds for all leaves and external nodes at some depth $d \leq D-1$, and consider a leaf or external node s at depth $d-1$. Then substituting (B.44) into (B.43) and noting that $\prod_t P_{w,sjt}/P_{e,sj} \to 0$ a.s. by Proposition B.1, completes the proof of the proposition. □

Combining Proposition B.2 with equation (3.6), we get that, for any leaf or external node s at depth d, a.s. as $n \to \infty$:

$$P_{b,s} = 1 - O\left(n^{-\frac{(m-1)^2}{2}+\varepsilon}\right), \quad \text{if } d \leq D-1, \qquad \text{and,} \quad P_{b,s} = 1, \quad \text{if } d = D. \tag{B.45}$$

Internal nodes. As in the proof of Lemma 3.2, we first consider internal nodes whose children are leaves of T^*, so that $I = I(X;J|s) > 0$. For these nodes, equation (3.8) gives,

$$\frac{P_{e,s}}{\prod_{j=0}^{m-1} P_{e,sj}} = \exp\big\{-nI\pi(s) + o(n)\big\}, \quad \text{a.s.}, \tag{B.46}$$

so that for any $\varepsilon > 0$,

$$\frac{P_{e,s}}{\prod_{j=0}^{m-1} P_{e,sj}} = o\big(\exp\big\{-n(1-\varepsilon)I\pi(s)\big\}\big), \quad \text{a.s.} \tag{B.47}$$

Substituting (B.47) in equation (3.7) of the main text we obtain the same bound for the branching probabilities,

$$P_{b,s} = o\Big(\exp\big\{-n(1-\varepsilon)I\pi(s)\big\}\Big), \quad \text{a.s.} \tag{B.48}$$

Next, we establish a corresponding bound for all internal nodes; indeed, for any node u that is a suffix of a node s that has $I(X;J|s) > 0$.

Lemma B.3. *Let s be a context of length $l(s) \leq D-1$ for which $I(X;J|s) > 0$, and let u be any suffix of s. Then, for any $\varepsilon > 0$, we have as $n \to \infty$:*

$$P_{b,u} = o\Big(\exp\big\{-n(1-\varepsilon)I(X;J|s)\pi(s)\big\}\Big), \quad \textit{a.s.} \tag{B.49}$$

Proof. Let $\Delta l = l(s) - l(u) \geq 0$; the proof is by induction on Δl. For $\Delta l = 0$, the claim is satisfied trivially as $u = s$, for which $I(X;J|s) > 0$, corresponding to the previous case.

For the inductive step, we assume that the claim holds for context uj which is the suffix of s with $\Delta l = k \geq 0$, and prove that it also holds for the context u, which is the suffix of s with $\Delta l = k+1$. For node u, which is at depth $d < D-1$, from (3.6) and the definition of the branching probabilities in (3.1) we get,

$$P_{b,u} \leq \Big(\frac{\beta}{1-\beta}\Big)\frac{P_{e,u}}{\prod_{t=0}^{m-1} P_{w,ut}} = C\frac{P_{e,u}}{\prod_{t=0}^{m-1} P_{e,ut}} \prod_{t=0}^{m-1} P_{b,ut},$$

where the constant $C = [\beta^{m-1}(1-\beta)]^{-1}$. And, as $P_{b,ut} \leq 1$ for all t, we can further bound,

$$P_{b,u} \leq C\,\frac{P_{e,u}}{\prod_{t=0}^{m-1} P_{e,ut}}\, P_{b,uj}, \tag{B.50}$$

keeping only the specific child uj of u which is a suffix of s. From (B.46) for node u, we know that, even if $I(X;J|u)$ is zero, we have,

$$\frac{P_{e,u}}{\prod_{t=0}^{m-1} P_{e,ut}} = \exp(o(n)), \quad \text{a.s.}, \tag{B.51}$$

and combining this with (B.50) and the inductive hypothesis that (B.49) holds for uj in place of u, we get,

$$P_{b,u} = o\Big(\exp\big\{-n(1-2\varepsilon)I(X;J|s)\pi(s)\big\}\Big), \quad \text{a.s.},$$

completing the proof of the inductive step and hence the proof of Lemma B.3. □

Substituting the bounds on the branching probabilities for the leaves (B.45) and internal nodes (B.49) in the expression for the posterior of T^* from (3.2), yields the result claimed in Theorem 3.2, completing its proof. Finally, a simple examination of (B.45) and (B.49) in the case when T^* is the full tree of depth D shows that $P_{b,s} = 1$ for all leaves s, so the rate is determined by the exponential bounds in (B.49) as claimed in Corollary 3.1. □

B.3 Results of Chapter 6

Chapter 6 contains our main theoretical results associated with the general BCT-SSM model class for real-valued time series, together with more specific results for the BCT-AR and BCT-ARCH models. In particular, Theorems 6.1–6.2 provide corresponding results for the CCTW and CBCT algorithms in the general BCT-SSM setting, with Proposition 6.1 establishing the equivalent branching process representation of the posterior in this case. Lemmas 6.1–6.3 establish expressions for the estimated probabilities $P_e(s,x)$ to be used with our inference algorithms in the cases of BCT-AR and BCT-ARCH models, respectively.

B.3.1 Proofs of Theorems 6.1 and 6.2

Theorems 6.1–6.2 are the extensions of Theorems 2.1–2.2 from the discrete-valued case, establishing the corresponding results for the CCTW and CBCT algorithms in the much more general setting of the BCT-SSM for real-valued time series.

The important observation in their proofs is that, using our different form of the estimated probabilities $P_e(s,x)$ defined in (6.4), here it is still possible to factorise the marginal likelihoods $p(x|T)$ for the general BCT-SSM, as,

$$p(x|T) = \int p(x|\theta,T)\pi(\theta|T)d\theta = \int \prod_{s\in T} \left(\prod_{i\in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1})\, \pi(\theta_s)\, d\theta_s \right) = \prod_{s\in T} P_e(s,x),$$

where the second equality follows from the likelihood of the general BCT-SSM in (6.2), together with the fact that we still use independent priors on the parameters at the leaves, so that $\pi(\theta|T) = \prod_{s\in T} \pi(\theta_s)$. The above factorisation of the marginal likelihoods $p(x|T)$ is the exact analog of Lemma 2.2 for the general BCT-SSM setting.

With this modification, a careful inspection of the proofs of Theorems 2.1–2.2 shows that the remaining steps can be still be carried out as before. So, the proofs of Theorems 6.1–6.2 follow along the same lines as those of Theorems 2.1–2.2, with the main difference being that our new general version of the estimated probabilities $P_e(s,x)$ of (6.4) need to be used in place of their simple discrete versions $P_e(a_s)$ of (2.7). Similarly, the proof of Proposition 6.1 follows along the same lines as that of Proposition 3.2. □

B.3.2 The *k*-CBCT algorithm

The k-BCT algorithm can be generalised in a similar manner to the way the CTW and BCT algorithms were generalised for the BCT-SSM setting. The resulting k-CBCT algorithm identifies the top-k *a posteriori* most likely context trees in this case. The proof of the theorem claiming this is similar to the proof of Theorem 2.3 and thus omitted here. Again, the important difference, both in the algorithm description and in the proof of the theorem, is that the estimated probabilities $P_e(s,x)$ of (6.4) are used in place of their simple discrete version $P_e(a_s)$ of (2.7).

B.3.3 Proof of Lemma 6.1

The proofs of Lemmas 6.1-6.2 are mostly based on explicit computations. Recall that, for each context s, the set B_s consists of those indices $i \in \{1,2,\ldots,n\}$ such that the context of x_i is s. The important step in the following is the factorisation of the likelihood using the sets B_s as in (6.2). In order to prove the lemmas for the AR model with parameters $\theta_s = (\phi_s, \sigma_s^2)$, we first consider an intermediate step in which we assume the noise variance to be known and equal to σ^2.

Known noise variance. Here, to any leaf s of the context tree T, we associate an AR model with known variance σ^2, so that,

$$x_n = \phi_{s,1}x_{n-1} + \cdots + \phi_{s,p}x_{n-p} + e_n = {\phi_s}^{\mathrm{T}}\,\widetilde{\mathbf{x}}_{n-1} + e_n, \quad e_n \sim \mathcal{N}(0,\sigma^2). \tag{B.52}$$

In this setting, the parameters of the model are only the AR coefficients $\theta_s = \phi_s$. For these, we use a Gaussian prior,

$$\theta_s \sim \mathcal{N}(\mu_o, \Sigma_o)\ , \tag{B.53}$$

where μ_o, Σ_o are hyperparameters. In this setting we prove the following for the estimated probabilities $P_e(s,x)$.

Lemma B.4. *The estimated probabilities $P_e(s,x)$ for the known-variance case are given by,*

$$P_e(s,x) = \frac{1}{(2\pi\sigma^2)^{|B_s|/2}}\ \frac{1}{\sqrt{det(I+\Sigma_o S_3/\sigma^2)}}\ \exp\left\{-\frac{E_s}{2\sigma^2}\right\}, \tag{B.54}$$

where I is the identity matrix and E_s is given by,

$$E_s = s_1 + \sigma^2\mu_o^T\Sigma_o^{-1}\mu_o - (\mathbf{s}_2 + \sigma^2\Sigma_o^{-1}\mu_o)^T(S_3+\sigma^2\Sigma_o^{-1})^{-1}(\mathbf{s}_2+\sigma^2\Sigma_o^{-1}\mu_o). \tag{B.55}$$

Proof. For the AR model of (B.52),

$$\prod_{i\in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1}) = \frac{1}{(\sqrt{2\pi\sigma^2})^{|B_s|}}\ \exp\left\{-\frac{1}{2\sigma^2}\sum_{i\in B_s}(x_i-{\theta_s}^{\mathrm{T}}\widetilde{\mathbf{x}}_{i-1})^2\right\}.$$

Expanding the sum in the exponent gives,

$$\begin{aligned}\sum_{i\in B_s}(x_i-{\theta_s}^{\mathrm{T}}\widetilde{\mathbf{x}}_{i-1})^2 &= \sum_{i\in B_s}x_i^2-2\theta_s^{\mathrm{T}}\sum_{i\in B_s}x_i\widetilde{\mathbf{x}}_{i-1}+\theta_s^{\mathrm{T}}\sum_{i\in B_s}\widetilde{\mathbf{x}}_{i-1}\widetilde{\mathbf{x}}_{i-1}^{\mathrm{T}}\theta_s\\ &= s_1-2\theta_s^{\mathrm{T}}\mathbf{s}_2+\theta_s^{\mathrm{T}}S_3\theta_s,\end{aligned}$$

from which we obtain that,

$$\begin{aligned}\prod_{i\in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1}) =& \frac{1}{(\sqrt{2\pi\sigma^2})^{|B_s|}}\ \exp\left\{-\frac{1}{2\sigma^2}(s_1-2\theta_s^{\mathrm{T}}\mathbf{s}_2+\theta_s^{\mathrm{T}}S_3\theta_s)\right\}\\ =&(\sqrt{2\pi})^p\rho_s\,\mathcal{N}(\theta_s;\mu,S),\end{aligned}$$

by completing the square, where $\mu = S_3^{-1}\mathbf{s}_2$, $S=\sigma^2S_3^{-1}$, and,

$$\rho_s = \sqrt{\frac{\det(\sigma^2S_3^{-1})}{(2\pi\sigma^2)^{|B_s|}}}\ \exp\left\{-\frac{1}{2\sigma^2}(s_1-\mathbf{s}_2^{\mathrm{T}}S_3^{-1}\mathbf{s}_2)\right\}. \tag{B.56}$$

So, multiplying with the prior:

$$\prod_{i\in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1})\pi(\theta_s) = (\sqrt{2\pi})^p\rho_s\,\mathcal{N}(\theta_s;\mu,S)\,\mathcal{N}(\theta_s;\mu_o,\Sigma_o) = \rho_s Z_s\,\mathcal{N}(\theta_s;\mathbf{m},\Sigma),$$

where $\Sigma^{-1}=\Sigma_o^{-1}+S^{-1}$, $\mathbf{m}=\Sigma\,(\Sigma_o^{-1}\mu_o+S^{-1}\mu)$, and,

$$Z_s = \frac{1}{\sqrt{\det(\Sigma_o+\sigma^2S_3^{-1})}}\ \exp\left\{-\frac{1}{2}(\mu_o-S_3^{-1}\mathbf{s}_2)^{\mathrm{T}}(\Sigma_o+\sigma^2S_3^{-1})^{-1}(\mu_o-S_3^{-1}\mathbf{s}_2)\right\}. \tag{B.57}$$

Therefore,

$$\prod_{i\in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1})\pi(\theta_s) = \rho_s Z_s\,\mathcal{N}(\theta_s;\mathbf{m},\Sigma), \tag{B.58}$$

and hence,

$$P_e(s,x) = \int\prod_{i\in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1})\ \pi(\theta_s)\,d\theta_s\ = \rho_s Z_s.$$

Using standard matrix inversion properties, after some algebra the product $\rho_s Z_s$ can be rearranged to give exactly the required expression in (B.54). □

Now, we move back to the original case, where the noise variance is considered to be a parameter of the AR model, so that $\theta_s = (\phi_s, \sigma_s^2)$. Here, the joint prior on the parameters is $\pi(\theta_s) = \pi(\phi_s|\sigma_s^2)\pi(\sigma_s^2)$, where,

$$\sigma_s^2 \sim \text{Inv-Gamma}(\tau, \lambda), \tag{B.59}$$

$$\phi_s|\sigma_s^2 \sim \mathcal{N}(\mu_o, \sigma_s^2\Sigma_o), \tag{B.60}$$

and where $(\tau, \lambda, \mu_o, \Sigma_o)$ are the prior hyperparameters. For the estimated probabilities $P_e(s,x)$, we just need to compute the integral:

$$P_e(s,x) = \int \prod_{i\in B_s} p(x_i|T, \theta_s, x_{-D+1}^{i-1})\ \pi(\theta_s)\ d\theta_s \tag{B.61}$$

$$= \int \pi(\sigma_s^2) \left(\int \prod_{i\in B_s} p(x_i|T, \phi_s, \sigma_s^2, x_{-D+1}^{i-1})\ \pi(\phi_s|\sigma_s^2)\ d\phi_s \right) d\sigma_s^2. \tag{B.62}$$

The inner integral has exactly the form of the estimated probabilities $P_e(s,x)$ from the previous section, where the noise variance was fixed. The only difference is that the prior $\pi(\phi_s|\sigma_s^2)$ of (B.60) now has covariance matrix $\sigma_s^2\Sigma_o$ instead of Σ_o. So, using (B.54)-(B.55), with Σ_o replaced by $\sigma_s^2\Sigma_o$, we get,

$$P_e(s,x) = \int \pi(\sigma_s^2) \left\{ C_s^{-1} \left(\frac{1}{\sigma_s^2} \right)^{|B_s|/2} \exp\left(-\frac{D_s}{2\sigma_s^2} \right) \right\} d\sigma_s^2,$$

with C_s and D_s as in Lemma 1. And using the inverse-gamma prior $\pi(\sigma_s^2)$ of (B.59),

$$P_e(s,x) = C_s^{-1}\ \frac{\lambda^\tau}{\Gamma(\tau)} \int \left(\frac{1}{\sigma_s^2} \right)^{\tau'+1} \exp\left(-\frac{\lambda'}{\sigma_s^2} \right) d\sigma_s^2, \tag{B.63}$$

with $\tau' = \tau + \frac{|B_s|}{2}$ and $\lambda' = \lambda + \frac{D_s}{2}$.

The integral in (B.63) has the form of an inverse-gamma density with parameters τ' and λ', whose closed-form solution is,

$$P_e(s,x) = C_s^{-1}\ \frac{\lambda^\tau}{\Gamma(\tau)}\ \frac{\Gamma(\tau')}{(\lambda')^{\tau'}}\ ,$$

which, as required, completes the proof of Lemma 6.1. □

B.3.4 Proof of Lemma 6.2

In order to derive the required expressions for the posterior distributions of ϕ_s and σ_s^2 , for a leaf s of model T, first consider the joint posterior $\pi(\theta_s|T,x) = \pi(\phi_s, \sigma_s^2|T,x)$, given by,

$$\pi(\theta_s|T,x) \propto p(x|T,\theta_s)\pi(\theta_s) = \prod_{i=1}^{n} p(x_i|T,\theta_s,x_{-D+1}^{i-1})\pi(\theta_s) \propto \prod_{i\in B_s} p(x_i|T,\theta_s,x_{-D+1}^{i-1})\pi(\theta_s),$$

where we used the fact that, in the product, only the terms involving indices $i \in B_s$ are functions of θ_s. So,

$$\pi(\phi_s, \sigma_s^2|T,x) \propto \left(\prod_{i\in B_s} p(x_i|T,\phi_s,\sigma_s^2,x_{-D+1}^{i-1})\ \pi(\phi_s|\sigma_s^2) \right) \pi(\sigma_s^2).$$

Here, the first two terms can be computed from (B.58) of the previous section, where the noise variance was known. Again, the only difference is that we have to replace Σ_o with $\sigma_s^2\Sigma_o$ because of the prior $\pi(\phi_s|\sigma_s^2)$ defined in (B.60). After some algebra, this gives,

$$\pi(\phi_s, \sigma_s^2|T,x) \propto \left(\frac{1}{\sigma_s^2}\right)^{|B_s|/2} \exp\left(-\frac{D_s}{2\sigma_s^2}\right) \mathcal{N}(\phi_s;\mathbf{m}_s,\Sigma_s)\ \pi(\sigma_s^2),$$

with $\mathbf{m}_s$ defined as in Lemma 6.2, and $\Sigma_s = \sigma_s^2(S_3 + \Sigma_o^{-1})^{-1}$. Substituting the prior $\pi(\sigma_s^2)$ in the last expression gives,

$$\pi(\phi_s, \sigma_s^2|T,x) \propto \left(\frac{1}{\sigma_s^2}\right)^{\tau+1+|B_s|/2} \exp\left(-\frac{\lambda + D_s/2}{\sigma_s^2}\right) \mathcal{N}(\phi_s;\mathbf{m}_s,\Sigma_s). \tag{B.64}$$

From (B.64), it is easy to integrate out ϕ_s and get the posterior of σ_s^2,

$$\pi(\sigma_s^2|T,x) = \int \pi(\phi_s, \sigma_s^2|T,x)\ d\phi_s \propto \left(\frac{1}{\sigma_s^2}\right)^{\tau+1+|B_s|/2} \exp\left(-\frac{\lambda + D_s/2}{\sigma_s^2}\right),$$

which is of the form of an inverse-gamma distribution with parameters $\tau' = \tau + \frac{|B_s|}{2}$ and $\lambda' = \lambda + \frac{D_s}{2}$, proving the first part of the lemma.

However, as Σ_s is a function of σ_s^2, integrating out σ_s^2 requires more algebra. In specific, we have that,

$$\begin{aligned}\mathcal{N}(\phi_s;\mathbf{m}_s,\Sigma_s) &\propto \frac{1}{\sqrt{\det(\Sigma_s)}} \exp\left\{ -\frac{1}{2}(\phi_s - \mathbf{m}_s)^{\mathrm{T}}\Sigma_s^{-1}(\phi_s - \mathbf{m}_s) \right\} \\ &\propto \left(\frac{1}{\sigma_s^2}\right)^{p/2} \exp\left\{ -\frac{1}{2\sigma_s^2}(\phi_s - \mathbf{m}_s)^{\mathrm{T}}(S_3 + \Sigma_o^{-1})(\phi_s - \mathbf{m}_s) \right\},\end{aligned}$$

and substituting this in (B.64) gives that $\pi(\phi_s, \sigma_s^2|T,x)$ is proportional to,

$$\left(\frac{1}{\sigma_s^2}\right)^{\tau+1+\frac{|B_s|+p}{2}} \exp\left\{-\frac{1}{2\sigma_s^2}\left(2\lambda + D_s + (\phi_s - \mathbf{m}_s)^{\mathrm{T}}(S_3+\Sigma_o^{-1})(\phi_s-\mathbf{m}_s)\right)\right\},$$

which as a function of σ_s^2 has the form of an inverse-gamma density, allowing us to integrate out σ_s^2. Denoting $L = 2\lambda + D_s + (\phi_s - \mathbf{m}_s)^{\mathrm{T}}(S_3+\Sigma_o^{-1})(\phi_s-\mathbf{m}_s)$, and $\widetilde{\tau} = \tau + \frac{|B_s|+p}{2}$,

$$\pi(\phi_s|T,x) = \int \pi(\phi_s, \sigma_s^2|T,x)\, d\sigma_s^2 \propto \int \left(\frac{1}{\sigma_s^2}\right)^{\widetilde{\tau}+1} \exp\left(-\frac{L}{2\sigma_s^2}\right) d\sigma_s^2 = \frac{\Gamma(\widetilde{\tau})}{(L/2)^{\widetilde{\tau}}}\,.$$

So, as a function of ϕ_s, the posterior $\pi(\phi_s|T,x)$ is,

$$\begin{aligned}
\pi(\phi_s|T,x) \propto L^{-\widetilde{\tau}} &= \left(2\lambda + D_s + (\phi_s - \mathbf{m}_s)^{\mathrm{T}}(S_3+\Sigma_o^{-1})(\phi_s-\mathbf{m}_s)\right)^{-\frac{2\tau+|B_s|+p}{2}} \\
&\propto \left(1 + \frac{1}{2\tau+|B_s|}\,(\phi_s-\mathbf{m}_s)^{\mathrm{T}}\frac{(S_3+\Sigma_o^{-1})(2\tau+|B_s|)}{(2\lambda+D_s)}(\phi_s-\mathbf{m}_s)\right)^{-\frac{2\tau+|B_s|+p}{2}} \\
&\propto \left(1 + \frac{1}{\nu}\,(\phi_s-\mathbf{m}_s)^{\mathrm{T}}P_s^{-1}(\phi_s-\mathbf{m}_s)\right)^{-\frac{\nu+p}{2}},
\end{aligned}$$

which is exactly in the form of a multivariate t-distribution, with p being the dimension of ϕ_s, and with $\nu, \mathbf{m}_s$ and P_s exactly as given in Lemma 6.2, completing its proof. □

B.3.5 Proof of Lemma 6.3

For the BCT-ARCH model, at every leaf s,

$$x_n \sim \mathcal{N}(0,\sigma_n^2), \quad \sigma_n^2 = \alpha_{s,0} + \alpha_{s,1}x_{n-1}^2 + \cdots + \alpha_{s,p}x_{n-p}^2 = \alpha_s^{\mathrm{T}}\,\mathbf{z}_{n-1}, \tag{B.65}$$

where $\theta_s = \alpha_s = (\alpha_{s,0}, \alpha_{s,1}, \ldots, \alpha_{s,p})^{\mathrm{T}}$ and $\mathbf{z}_{n-1} = (1, x_{n-1}^2, \ldots, x_{n-p}^2)^{\mathrm{T}}$. The proof of the lemma follows directly by considering the log-likelihood of data with context s, given by,

$$L_s(\theta_s) = \sum_{i\in B_s} \log p(x_i|x_{-D+1}^{i-1}, \theta_s) = -\frac{|B_s|}{2}\log(2\pi) - \frac{1}{2}\sum_{i\in B_s}\left(\log\sigma_i^2 + \frac{x_i^2}{\sigma_i^2}\right), \tag{B.66}$$

and taking its derivatives with respect to θ_s.

As the dependence is implicit through $\sigma_i^2 = \theta_s^{\mathrm{T}}\, \mathbf{z}_{i-1}$, taking the first derivative gives,

$$\frac{\partial L_s}{\partial \theta_s} = \frac{1}{2}\sum_{i\in B_s} \frac{1}{\sigma_i^2}\left(\frac{x_i^2}{\sigma_i^2} - 1\right)\frac{\partial \sigma_i^2}{\partial \theta_s} = \frac{1}{2}\sum_{i\in B_s} \frac{1}{\sigma_i^2}\left(\frac{x_i^2}{\sigma_i^2} - 1\right)\mathbf{z}_{i-1}, \tag{B.67}$$

and similarly taking the second derivative and its expectation finally gives,

$$\widehat{I}_s = \left\{-\mathbb{E}\left(\frac{\partial^2 L_s}{\partial \theta_s^2}\right)\right\} = \frac{1}{2}\sum_{i\in B_s}\left(\frac{1}{\sigma_i^4}\right)\mathbf{z}_{i-1}\mathbf{z}_{i-1}^{\mathrm{T}}, \tag{B.68}$$

completing the proof of the lemma. □

B.4 MCMC samplers

Finally, in this section we give explicit expressions for the accept-reject ratios in the various MCMC samplers developed in this thesis. The exact expressions follow easily from standard Metropolis-Hastings methodology, by considering the corresponding proposal distribution that is used in each case.

RW sampler. The ratios $r(T,T')$ in the acceptance probabilities of the RW sampler in Section 2.3.4 are given by,

$$r(T,T') := \frac{\pi(T'|x)}{\pi(T|x)} \times \begin{cases} 1/2, & \text{in case (a);} \\ m^{D-1}/2, & \text{in case (b);} \\ (|T| - L_D(T))/N_D(T'), & \text{in case (c), if } T' \neq T_c(D); \\ 2m^{-D+1}, & \text{in case (c), if } T' = T_c(D); \\ N_D(T)/(|T'| - L_D(T')), & \text{in case (d), if } T' \neq \Lambda; \\ 2, & \text{in case (d), if } T' = \Lambda, \end{cases}$$

where $N_D(T)$ denotes the number of internal nodes in a tree T having only m descendants, and the posterior odds $\pi(T'|x)/\pi(T|x)$ can be easiliy computed via (2.15).

Jump sampler. The jump sampler of Section 2.3.4 incorporates the RW sampler's proposal distribution as one of its steps: Let $q(T'|T)$ denote the proposal probabilities of the RW sampler, so that $q(T'|T)$ equals 1, m^{-D+1}, $1/[2(|T| - L_D(T))]$ and $1/[2N_D(T)]$, in each of its cases (ia), (ib), (ic) and (id), respectively. We say that two models T, T' are *neighbours* if they differ by exactly one branch of m children.

The ratios $r(T,T')$ in the acceptance probabilities of the jump sampler are given by,

$$r(T,T') := \frac{\pi(T'|x)}{\pi(T|x)} \times \begin{cases} \frac{(1-p)q(T|T')+pk^{-1}\mathbb{I}\{T\in\mathcal{T}^*\}}{(1-p)q(T'|T)+pk^{-1}\mathbb{I}\{T'\in\mathcal{T}^*\}}, & \text{if } T,T' \text{ are neighbours,} \\ \mathbb{I}\{T\in\mathcal{T}^*\}, & \text{if } T,T' \text{ are not neighbours,} \end{cases}$$

where $\mathbb{I}\{\cdot\}$ denotes the indicator function of the event $\{\cdot\}$.

Change-point detection. Here we consider the MCMC sampler introduced in Chapter 5 in the setting of change-point detection, where the number of change-points is unknown. In this case, the ratio $r((\ell,\mathbf{p}),(\ell',\mathbf{p}'))$ in the acceptance probability of the MCMC sampler described in Section 5.1.2 is given by,

$$\frac{P(x|\mathbf{p}',\ell')}{P(x|\mathbf{p},\ell)} \times \frac{\prod_{j=0}^{\ell'}(p'_{j+1}-p'_j-1)}{\prod_{j=0}^{\ell}(p_{j+1}-p_j-1)} \times \begin{cases} \frac{2(n-2)}{(n-3)(n-4)}, \text{ if } \ell=0; \\ \frac{(n-3)(n-4)}{2(n-2)}, \text{ if } \ell=1, \ell'=0; \\ \frac{3(2\ell_{\max}+1)(n-\ell_{\max}-1)}{(n-2\ell_{\max}-2)(n-2\ell_{\max}-1)}, \text{ if } \ell=\ell_{\max}-1,\ \ell'=\ell_{\max}; \\ \frac{(n-2\ell_{\max}-2)(n-2\ell_{\max}-1)}{3(2\ell_{\max}+1)(n-\ell_{\max}-1)}, \text{ if } \ell=\ell_{\max},\ \ell'=\ell_{\max}-1; \\ \frac{(n-2\ell-2)(n-2\ell-1)}{2(2\ell+1)(n-\ell-1)}, \text{ if } \ell'=\ell-1,\ \ell\neq 1,\ell_{\max}; \\ \frac{2(2\ell'+1)(n-\ell'-1)}{(n-2\ell'-2)(n-2\ell'-1)}, \text{ if } \ell'=\ell+1,\ \ell'\neq 1,\ell_{\max}; \\ 1, \text{ otherwise,} \end{cases}$$

where, as before, $\ell_{\max}$ is the maximum possible number of change-points, and the terms $P(x|\mathbf{p},\ell)$ and $P(x|\mathbf{p}',\ell')$ can be easily computed using the CTW algorithm.

Appendix C

Additional results with discrete-valued data

In this section we provide additional experimental results with discrete-valued time series. These include both simulation experiments and real-world applications, focusing in the tasks of model selection and change-point detection.

C.1 Model selection

Renewal process. The binary variable-memory chain $\{X_n\}$ with model and parameters shown in Figure C.1 was examined in the VLMC work of (Bühlmann, 2000, p. 303). Note that the distribution of the next symbol produced by the chain only depends on how far in the past the most recent "0" appeared.

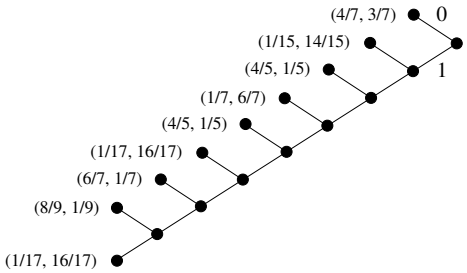


Figure C.1: The model and parameters of the renewal-like binary chain.

• $n = 200$. With $n = 200$ samples, the BCT algorithm with $\beta = 1/2$ and $D = 10$ produces a MAP model T_1^* which is the same as the true model pruned at depth 4; its prior probability is $\pi(T_1^*) \approx 0.002$ and its posterior is $\pi(T_1^*|x) \approx 0.0725$. The next four most likely models produced by the k-BCT algorithm are small variations of T_1^* of maximal depth 4 or 5. The total posterior probability of the top-5 models is ≈ 0.1782.

The best-BIC-VLMC gives the same model as the BCT algorithm (with good AIC and BIC scores), while the default-VLMC and best-AIC-VLMC both produce a tree of depth 8. It is the same as the true model up to depth 4, but it also includes the depth-8 branch corresponding to the context $s = 01111001$, which does not appear in the true model. It has a marginally better AIC score than T_1^* (by $\approx 1\%$), but a much worse BIC score, somewhat overfitting the data.

The best-BIC and best-AIC versions of MTD both give $D = 2$; and the best-BIC and best-AIC versions of MTDg both give $D = 4$. In all four cases, the resulting AIC and BIC scores are not competitive with those of the BCT and the best-BIC-VLMC algorithms.

• $n = 1,000$. The results of the k-BCT algorithm (with $k = 5$, $D = 10$ and $\beta = 1/2$) on a sample of length $n = 1,000$, reproduce more of the true underlying structure; see Figure C.2. The MAP tree T_1^* is the true model pruned at depth 6, and it has at least as good BIC and AIC scores as all the other methods.

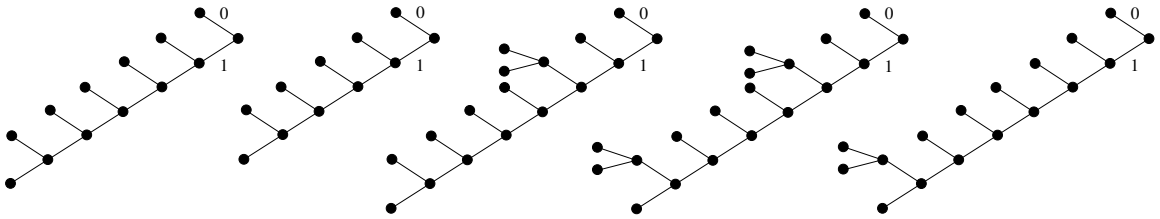


Figure C.2: The top $k = 5$ models obtained by k-BCT with $n = 1,000$ samples from the renewal-like chain. The prior probability of T_1^* is $\pi(T_1^*) \approx 1.2 \times 10^{-4}$ and its posterior is $\pi(T_1^*|x) \approx 0.0799$. The posterior odds $\pi(T_1^*|x)/\pi(T_i^*|x)$ for the next four models are approximately 1.25, 1.98 2.48 and 4, for $i = 2,3,4,5$, respectively. The total posterior probability of the top 5 models is approximately 0.2336.

Despite the larger data length, the best-BIC-VLMC again gives the depth-4 version of the true model. The default-VLMC and best-AIC-VLMC produce larger trees of depth 11, very similar between them but very different from the true model and the MAP tree; all of their leaves except one are not present in the true model. The long branches of depth 8-11 are clear examples of overfitting, resulting in poor BIC scores, while even the AIC score of the best-AIC-VLMC tree is within less than 0.1% of that of our MAP model.

The best-AIC-MTD and best-BIC-MTD both give $D = 5$ as the optimal depth, while the best-AIC-MTDg and best-BIC-MTDg both give $D = 4$. Once again, their scores are not competitive with those of the MAP model and the VLMC results.

A third order binary chain. Here we examine data generated from the 3rd order variable-memory chain in example V4 of the MTD paper (Berchtold and Raftery, 2002, p. 353). The model and its associated parameters are shown in Figure C.3.

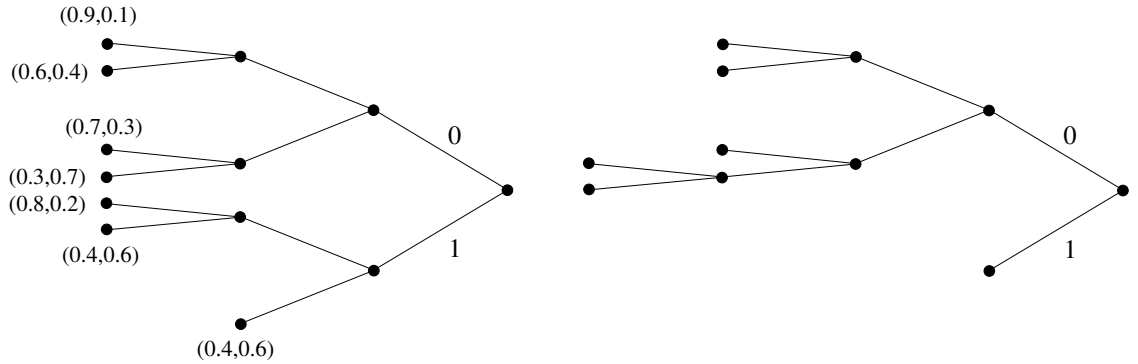


Figure C.3: True model and MAP tree model in the third order binary chain example. Left: True underlying model. Right: MAP tree model identified by the BCT algorithm from $n = 200$ samples.

• $n = 200$. The MAP model T_1^* identified by the BCT algorithm (with $D = 10$ and $\beta = 1/2$) from $n = 200$ simulated samples is shown in Figure C.3; its prior is $\pi(T_1^*) \approx 4.9 \times 10^{-4}$, and its posterior is $\pi(T_1^*|x) \approx 0.0363$. The top part of the true tree is correctly identified, the lower part is cropped at 1, and there is an extra branch of depth 4 that is not in the true model. The next four *a posteriori* most likely models are shown in Figure C.4.

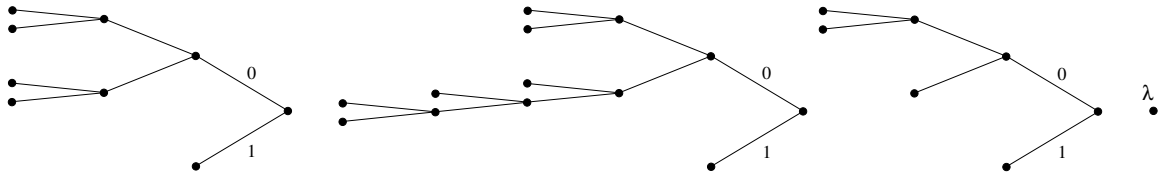


Figure C.4: The next four *a posteriori* most likely trees T_i^* identified by k-BCT with $n = 200$ samples from the third order binary chain. Notice that T_5^* is empty tree Λ consisting of just the root node λ. The posterior odds $\pi(T_1^*|x)/\pi(T_i^*|x)$ for $i = 2,3,4,5$ are approximately 1.27, 1.33, 2.53 and 2.78, respectively, and the sum of the posteriors of the top-5 models is ≈ 0.12.

The result of the best-BIC-VLMC is the same as T_2^*. Both T_1^* and T_2^* have very good and very similar AIC and BIC scores. The results of the default-VLMC and the best-AIC VLMC are shown in Figure C.5. They both have depth 6, and they bear little resemblance to the true model. Their AIC scores are good but not significantly better than the scores of T_1^* and T_2^*. Their BIC scores are rather poor, once again suggesting that the results of the default-VLMC and the best-AIC-VLMC overfit the data.

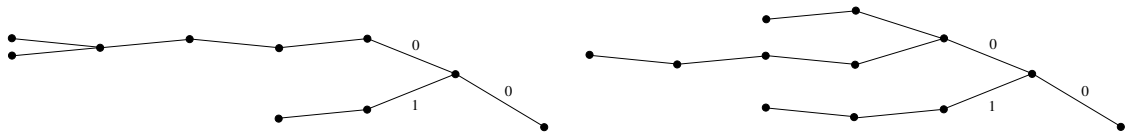


Figure C.5: The models produced by the default-VLMC (left) and the best-AIC-VLMC (right) with $n = 200$ samples from the third order binary chain.

The best-AIC-MTD and best-BIC-MTD both give $D = 3$, corresponding to a 3rd order chain which is not terribly different from the true model. Its BIC score is about 0.4% better than that of the MAP and the best-BIC-VLMC models, while its AIC is slightly worse than the other two. The best-BIC-MTDg gave $D = 0$, and the best-AIC-MTDg gave $D = 3$. In both cases of MTDg, the corresponding scores were not competitive with those of the other methods.

• $n = 1,000$. The MAP tree T_1^* produced by the BCT algorithm (with $D = 10$ and $\beta = 1/2$) from $n = 1,000$ samples is the true underlying model; its prior is $\pi(T_1^*) \approx 1.2 \times 10^{-4}$ and its posterior is $\pi(T_1^*|x) \approx 0.1065$. The next four *a posteriori* most likely models are shown in Figure C.6.

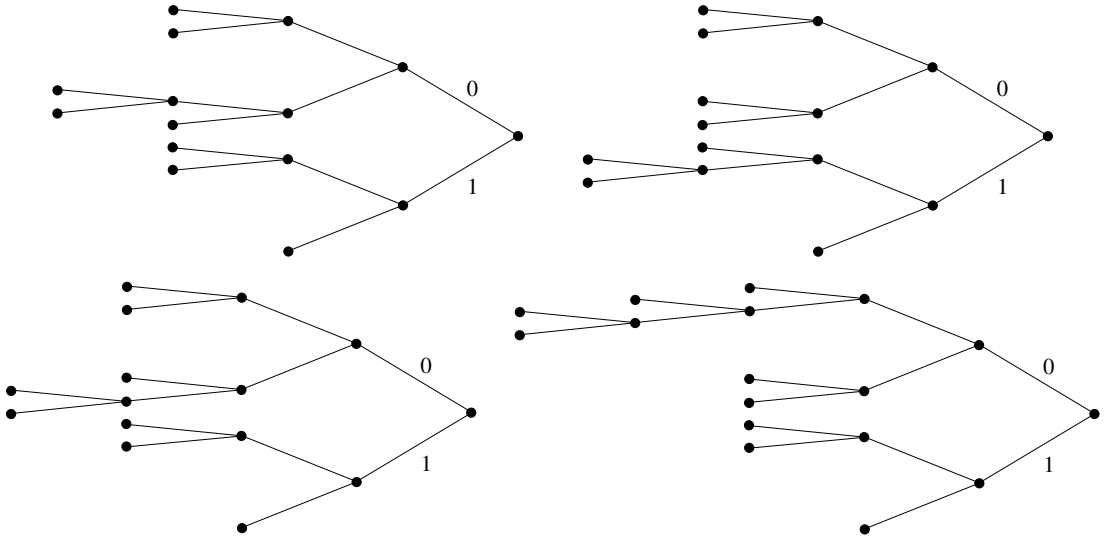


Figure C.6: The next four *a posteriori* most likely trees T_i^* identified by k-BCT with $n = 1,000$ samples from the third order binary chain. Their posterior odds $\pi(T_1^*|x)/\pi(T_i^*|x)$ for $i = 2,3,4,5$ are approximately 2.52, 3.96, 4.66 and 5.47, respectively, and the sum of the posteriors of the top-5 tree models is ≈ 0.2179.

The best-BIC-VLMC produces the true model as well, while the default-VLMC and best-AIC-VLMC produce very large trees of depth 18, which are not at all similar to the true model. They have good AIC scores (although not much better than the MAP and best-BIC-VLMC models), but their BIC scores are significantly worse.

The best-AIC-MTD and best-BIC-MTD give $D = 3$, which corresponds to a model with a BIC about 0.2% higher than that of the true underlying tree. The best-AIC-MTDg and best-BIC-MTDg also give $D = 3$, but the scores of the corresponding models are worse than those produced by the other methods.

Financial data. Here we consider the tick-by-tick price changes of the Facebook stock price during a six-and-a-half-hour-long trading period on October 3, 2016. The price change is recorded every time there is a trade; if the price goes down during the ith trade we set $x_i = 0$, if it stays the same we set $x_i = 1$, and if it goes up $x_i = 2$. This produces a ternary time series of length $n = 50,745$ observations.

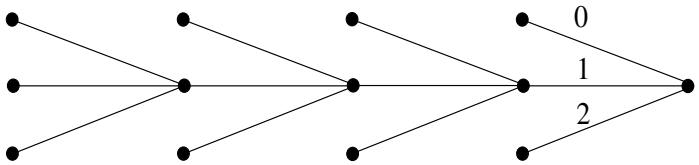


Figure C.7: The MAP tree model in the Facebook stock price dataset.

The MAP model T_1^* identified by the BCT algorithm with $D = 20$ and $\beta = 3/4$ is shown in Figure C.7. Its prior probability is $\pi(T_1^*) \approx 2.9 \times 10^{-4}$, and its posterior is $\pi(T_1^*|x) \approx 0.416$. Note that T_1^* admits an interpretation very similar to that of other tree models that were selected in various financial datasets considered in this thesis, including Example 2.1 of Section 2.5 and the financial examples studied in Chapter 6. In particular, in order to determine the distribution of the next value, you need to look as far back as necessary until you see a price change, or until you have looked four places back in the past.

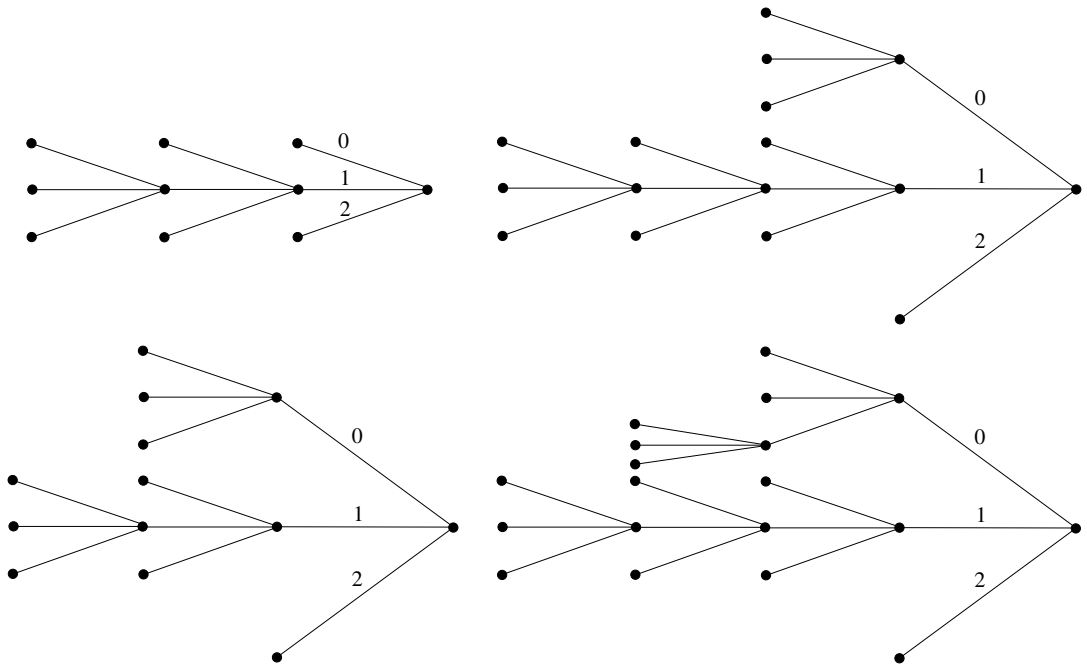


Figure C.8: The next four *a posteriori* most likely trees T_i^* identified by k-BCT in the Facebook stock price dataset. Their posterior odds $\pi(T_1^*|x)/\pi(T_i^*|x)$ are $1.017, 5.074, 5.161, 72.8$, for $i = 2,3,4,5$, respectively, and the sum of the posterior probabilities of the top-5 models is ≈ 0.993.

The next four *a posteriori* most likely models identified by k-BCT are shown in Figure C.8. The second most likely tree T_2^*, shown on the top left of Figure C.8, is the same as T_1^* but now pruned at depth 3; its posterior is $\pi(T_2^*|x) \approx 0.409$. Its form, together with the fact that the sum of the posteriors of the top-2 models is over 82%, provide further evidence of the above interpretation for the dependence structure present in financial data.

In this example, VLMC produces a model which is almost identical to the MAP tree, with the only difference being that VLMC gives the same parameter vectors θ_s to contexts $s = 1111$ and $s = 1112$. Their AIC and BIC scores are also essentially identical. Finally, the MTD produces a model of depth $D = 4$, with AIC and BIC scores only slightly worse than those of the MAP model, while the MTDg selects a model of order $D = 2$, whose corresponding AIC and BIC scores are much worse.

C.2 Change-point detection

Simulated data. The parameters of the four variable-memory Markov chains used for generating the simulated dataset of Section 5.2.2 are given below; the corresponding four tree models were also shown in Figure 5.3.

Model 1	Probability		
Context	0	1	2
0	0.3	0.4	0.3
2	0.5	0.3	0.2
10	0.2	0.5	0.3
11	0.1	0.4	0.5
121	0.7	0.2	0.1
122	0.4	0.2	0.4
1200	0.6	0.1	0.3
1201	0.3	0.5	0.2
1202	0.4	0.1	0.5

Model 2	Probability		
Context	0	1	2
0	0.4	0.5	0.1
2	0.4	0.4	0.2
10	0.4	0.2	0.4
11	0.2	0.4	0.4
12	0.6	0.1	0.3

Model 3	Probability		
Context	0	1	2
0	0.5	0.3	0.2
1	0.3	0.6	0.1
2	0.3	0.2	0.5

Model 4	Probability		
Context	0	1	2
λ	0.4	0.2	0.4

For this example, the posterior over the number of change-points and the change-point locations were presented in Figure 5.4 of Section 5.2.2, in the main text. In Figure C.9, here we also show zoomed-in versions of the MCMC histograms, showing the posterior of the location of each of the three change-points separately. The histograms were constructed using $N = 10^5$ MCMC samples obtained from the MCMC sampler of Section 5.1.2, with the first 10,000 samples discarded as burn-in.

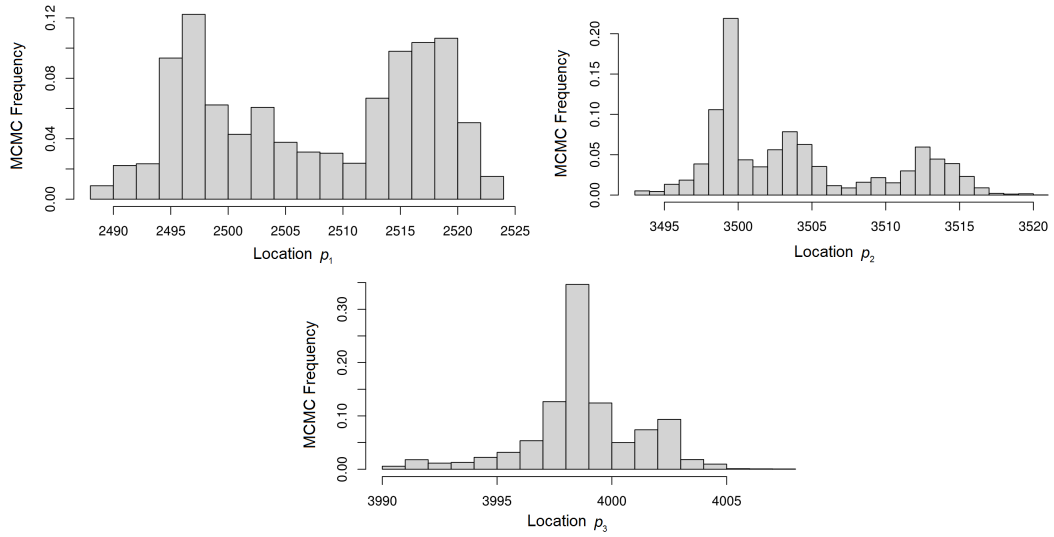


Figure C.9: MCMC histograms of the posterior of each change-point location in the simulated example of Section 5.2.2.

Bacteriophage lambda genome. In Figure C.10, we show zoomed-in versions of the MCMC histograms showing the posterior of the location of each of the four change-points separately for the bacteriophage lambda genome from Section 5.2.2.

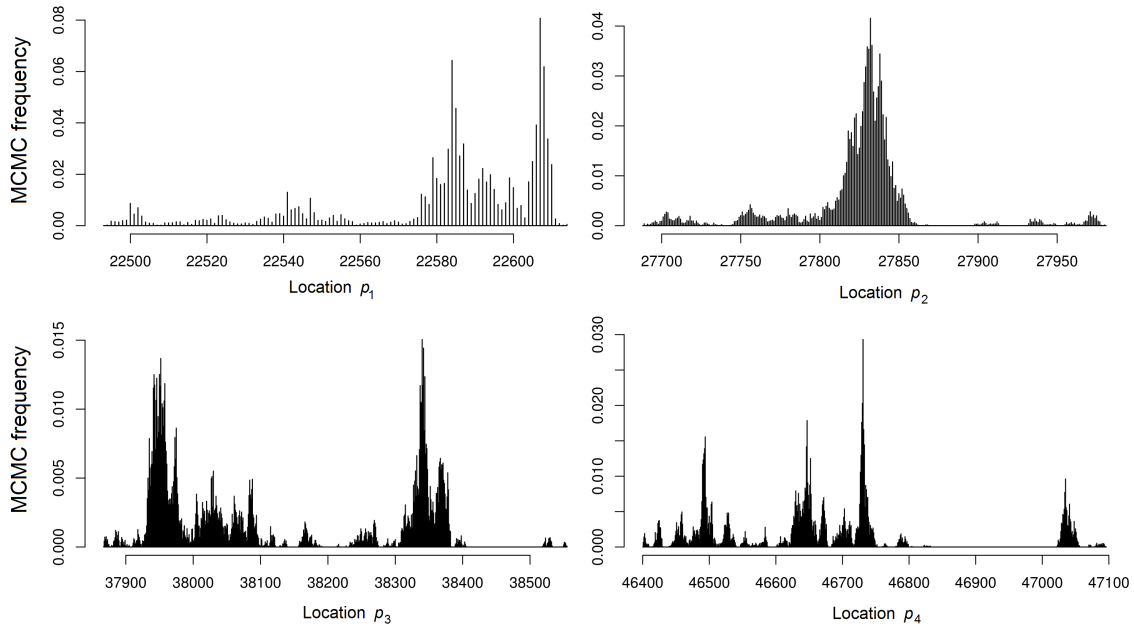


Figure C.10: MCMC histograms of the posterior of each change-point location in the bacteriophage lambda genome from Section 5.2.2.

El Niño occurrences. In Section 5.2.2, we studied a real dataset consisting of binary observations for the occurrences of El Niño events in the time period between 1525 and 2020. In Figure C.11 we show zoomed-in versions of the MCMC histograms, showing the posterior of the location of each of the two change-points separately.

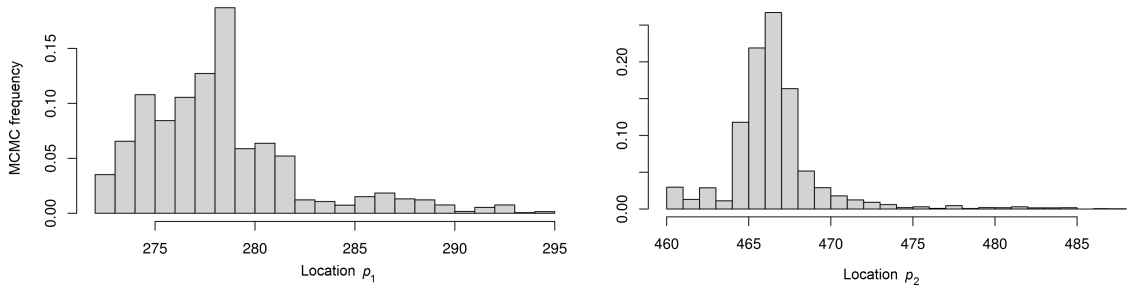


Figure C.11: MCMC histograms of the posterior of each change-point location in the El Niño example from Section 5.2.2.

Appendix D

Additional results with real-valued data

In this section we provide additional experimental results with real-valued time series. These include results with both versions of the general BCT-SSM model class considered in this thesis, namely, with BCT-AR and BCT-ARCH models. In addition, we provide a complete specification of the training details for all the methods used in the forecasting experiments of Chapter 6, and also report the corresponding empirical running times for each method.

D.1 Training details

D.1.1 BCT-AR experiments

Here we specify the training details for all the methods used in the forecasting experiments in Section 6.2.5. In all the examples the training set consists of the first 50% of the observations, and we allow updates at every timestep for all methods.

For the BCT-AR model, the complete training details are given in Section 6.2.3. There is no hyperparameter tuning from the data, the hyperparameters are just set to their default values. In particular, for the BCT prior we choose the default value $\beta = 1 - 2^{-m+1}$ and a maximum depth $D = 10$. We choose the AR order and the quantiser thresholds at the end of the training set by maximising the evidence, and then use the MAP tree model and parameters updated at every timestep, as explained in detail in the main text.

For ARIMA and ETS models we use the `R` package `forecast` (Hyndman and Khandakar, 2008); the corresponding functions `auto.arima` and `ets` are completely automated, so there are no parameters to specify here. For the NNAR model, we use the function `nnetar` that is also contained in the `forecast` package, and search over AR orders between $p = 1$ and $p = 5$. For the SETAR model, we use the `R` package `TSA` (Chan and Ripley, 2020), along with the commonly used conditional least squares method of Chan (1993). We search over AR orders between $p = 1$ and $p = 5$, and values of the delay parameter between $d = 1$ and $d = 5$. For

the MAR model, we use the `R` package `mix-AR` (Boshnakov and Ravagli, 2021), and try $K = 2$ and $K = 3$ components with AR orders between $p = 1$ and $p = 5$. For the MSA model, we use the `R` package `MSwM` (Sanchez-Espigares and Lopez-Moreno, 2021), which uses the EM algorithm for estimation, and try $K = 2$ and $K = 3$ hidden states with AR orders between $p = 1$ and $p = 5$.

For DeepAR and N-BEATS, we use the implementations available in the Python library 'GluonTS' (Alexandrov et al., 2020). As the computational cost per iteration differs for these methods, we use slightly different numbers of epochs and batches-per-epoch for each of them, in order to give similar empirical running times – which are still much higher than those of our methods; see below in Appendix D.3. For DeepAR we use 5 epochs with 50 batches/epoch, and for N-BEATS we use 3 epochs with 20 batches/epoch. In all cases we try AR orders between $p = 1$ and $p = 5$.

D.1.2 BCT-ARCH experiments

Here we specify the training details for all the methods used in the experiments of Section 6.3.5. Following Dellaportas and Vrontos (2007), in all forecasting experiments the size of the test set is $T = 130$ observations, which corresponds to half a year of working days. We allow for updates so that all models are re-trained at every timestep in the test set.

For the BCT-ARCH model we use binary trees with a quantiser threshold $c = 0$, in order to explicitly capture the leverage effect by distinguishing between positive and negative shocks. For the BCT prior we use the default value $\beta = 1/2$ corresponding to binary trees, and for the maximum tree depth and ARCH order take $D = p = 5$, corresponding to a full week of working days. Finally, we use $M = 10$ iterations for the scoring algorithm, which were always found to be enough in practice. In the forecasting experiments, for the predictive density we use the MAP tree model identified by the CBCT algorithm, together with the estimates of the ARCH coefficients from the scoring algorithm.

As benchmarks in the experiments we consider all the commonly used methods described in Section 6.3.4. For the family of ARCH models and their extensions, we use the implementation in the `R` package `rugarch` (Ghalanos, 2022). We use the automated function `ugarchforecast`, together with the model specifications `sGARCH`, `gjrGARCH`, and `eGARCH`, respectively, for the GARCH, GJR and EGARCH models. For the MSGARCH model we similarly use the `R` package `MSGARCH` (Ardia et al., 2019), and try $K = 2$ and $K = 3$ hidden states. For the SV model we use the `R` package `stochvol` (Kastner, 2016), which implements MCMC samplers for Bayesian inference. The function `predict.svdraws` is used to draw samples from the desired predictive density; where at every timestep in the test set we use 1,000 MCMC samples after a burn-in of 100 samples.

D.2 Model selection

D.2.1 BCT-AR experiments

Simulated data. The dataset `sim_1` is a simulated dataset consisting of $n = 600$ observations generated from a BCT-AR model with the tree of Figure 6.1, quantiser threshold $c = 0$, and AR order $p = 2$. The complete specification of this BCT-AR model was also given in Section 6.2.5 of the main text, as,

$$x_n = \begin{cases} 0.7\, x_{n-1} - 0.3\, x_{n-2} + e_n, & e_n \sim \mathcal{N}(0, 0.15), & \text{if } s = 1\text{: } x_{n-1} > 0, \\ -0.3\, x_{n-1} - 0.2\, x_{n-2} + e_n, & e_n \sim \mathcal{N}(0, 0.10), & \text{if } s = 01\text{: } x_{n-1} \leq 0,\ x_{n-2} > 0, \\ 0.5\, x_{n-1} + e_n, & e_n \sim \mathcal{N}(0, 0.05), & \text{if } s = 00\text{: } x_{n-1} \leq 0,\ x_{n-2} \leq 0. \end{cases}$$

In Section 6.2.5, the MAP tree model and parameters at the leaves were reported for this example, verifying the the BCT-AR inferential framework is very effective. Here, we also report the evidence $p(x|c, p)$ for a range of values of c and p. Although maximising the evidence is a very common, well-justified Bayesian practice (MacKay, 1992, 2003), here we report some values as a sanity check, to show that the evidence is indeed maximised at the true values of $c = 0.0$ and $p = 2$, and thus verify that our inferential procedure for choosing the quantiser and AR order is also effective.

Table D.1: Using the evidence $p(x|c, p)$ to choose the AR order and quantiser threshold.

	AR order p					Threshold c				
	1	2	3	4	5	-0.1	-0.05	0	0.05	0.1
$-\log_2 p(x\|c, p)$	533	**519**	526	531	535	558	539	**519**	555	577

The dataset `sim_2` is a simulated dataset consisting of $n = 500$ observations generated from a BCT-AR model with respect to the ternary tree in Figure D.1. The thresholds of the quantiser are $\{c_1 = -0.5, c_2 = 0.5\}$, and the AR order is $p = 1$. The complete specification of this BCT-AR model, is given by,

$$x_n = \begin{cases} 0.5\, e_n, & \text{if } s = 1,\ 01,\ 02,\ 20,\ 21, \\ 0.99\, x_{n-1} + 0.005\, e_n, & \text{if } s = 00,\ 22, \end{cases}$$

where $e_n \sim \mathcal{N}(0, 1)$.

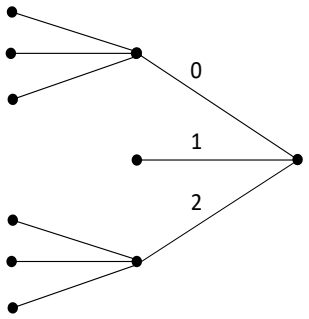


Figure D.1: Tree model of `sim_2`.

Finally, the dataset `sim_3` is a simulated dataset which consists of $n = 200$ observations generated from a SETAR model of order $p = 5$, given by,

$$x_n = \begin{cases} -0.1 + 0.9\, x_{n-1} + 0.9\, x_{n-2} - 0.2\, x_{n-5} + e_n, & \text{if } x_{n-1} > -0.2, \\ 0.2 + 0.1\, x_{n-1} + 0.9\, x_{n-5} + e_n, & \text{if } x_{n-1} \leq -0.2, \end{cases}$$

where $e_n \sim \mathcal{N}(0,1)$.

US unemployment rate. The dataset `unemp` is a real-world dataset, which consists of $n = 288$ observations of the quarterly US unemployment rate in the time period from 1948 to 2019. It is publicly available from the US Bureau of Labor Statistics (BLS), at https://data.bls.gov/timeseries/LNS14000000?years_option=all_years. The MAP BCT-AR model in this example is reported in Section 6.2.5 of the main text.

US Gross National Product. The dataset `gnp` is a real-world dataset, which consists of $n = 291$ observations of the quarterly US GNP from 1947 to 2019. It is available from the US Bureau of Economic Analysis (BEA), and can be retrieved from the Federal Reserve Bank of St. Louis (FRED) at https://fred.stlouisfed.org/series/GNP. Following Potter (1995), we consider the difference in the logarithm of the series, $y_n = \log x_n - \log x_{n-1}$.

For this dataset, the MAP BCT tree model is given in the main text, in Figure 6.2. It has depth $d = 3$, four leaves, $\mathcal{S} = \{0, 10, 110, 111\}$, and posterior 42.6%. The threshold of the binary quantiser selected using the procedure of Section 6.2.3 is $c = 0.2$, so that $s = 1$ if $y_{n-1} > 0.2$ and $s = 0$ if $y_{n-1} \leq 0.2$. The AR order selected is $p = 2$. The complete BCT-AR model, with its MAP estimated parameters is given by,

$$y_n = \begin{cases} 1.16 + 0.71\, y_{n-1} + 0.19\, y_{n-2} + 1.23\, e_n, & \text{if } s = 0, \\ 0.18 + 0.68\, y_{n-1} - 0.26\, y_{n-2} + 1.19\, e_n & \text{if } s = 10, \\ -1.05 + 1.40\, y_{n-1} + 0.19\, y_{n-2} + 1.04\, e_n & \text{if } s = 110, \\ 0.59 + 0.28\, y_{n-1} + 0.31\, y_{n-2} + 0.75\, e_n & \text{if } s = 111, \end{cases}$$

where $e_n \sim \mathcal{N}(0,1)$.

IBM stock price. The dataset `ibm` is a real dataset, which consists of $n = 369$ observations of the daily IBM common stock closing price, in the time period from May 17, 1961 to November 2, 1962. The dataset is taken from Box et al. (2015), and it is also available from the `R` package `fma` (Hyndman, 2020).

The MAP tree model fitted to this dataset is shown in the main text, in Figure 6.3. The complete BCT-AR model, with its MAP estimated parameters, is given by,

$$x_n = \begin{cases} 1.03\, x_{n-1} - 0.03\, x_{n-2} + 12.3\, e_n, & \text{if } s = 0, \\ 1.17\, x_{n-1} - 0.17\, x_{n-2} + 6.86\, e_n, & \text{if } s = 2, \\ -0.11\, x_{n-1} + 1.11\, x_{n-2} + 10.8\, e_n, & \text{if } s = 10, \\ 1.22\, x_{n-1} - 0.22\, x_{n-2} + 5.32\, e_n, & \text{if } s = 11, \\ 0.15\, x_{n-1} + 0.85\, x_{n-2} + 5.17\, e_n, & \text{if } s = 12, \end{cases}$$

where $e_n \sim \mathcal{N}(0,1)$.

D.2.2 BCT-ARCH experiments

Simulated data. In this section, some more extensive simulation experiments are presented, verifying that our inferential procedures are effective with data generated by BCT-ARCH models. As the effectiveness of our CCTW and CBCT algorithms was already tested with the BCT-AR model, the emphasis here is on illustrating that the approximations of (6.15) for the estimated probabilities are good enough in practice.

In general, this relies on two conditions. First, the maximum likelihood estimates of the scoring algorithm need to have converged; something which is easy to check by trying different initialisations and allowing for a large enough number of iterations M. Secondly, there need to be enough datapoints associated to each context s of T_{MAX} so that the Laplace approximations in (6.15) are close enough to the integrals of (6.4). This can be ensured by checking that the maximum depth D is small enough compared to the dataset size n[1]. In practice, values of $M = 10$ and $D = 5$ satisfy both the above conditions, and hence guarantee effective inference for datasets with size approximately in the range from $n = 1,000$ to $n = 10,000$ observations that we will be considering in the real-data section. This fact is illustrated in the simulation experiments presented below.

In the first simulated experiment, data are generated from a BCT-ARCH model where the underlying context tree is the binary tree T^* of depth $d = 1$, with leaves $\mathcal{S} = \{0,1\}$. First, we consider the evolution of the posterior over trees, $\pi(T|x)$, as more data from the model become available. The corresponding posterior of the true tree model, $\pi(T^*|x)$, is reported in Table D.2 for different values of the parameters M and D. Overall, it can be observed that with $M = 10$ and $D = 5$ our inferential procedures are effective.

[1] It is noted that this final condition would not be required with the MCMC approach of Chib (1995); Chib and Jeliazkov (2001), but at the expense of computational complexity.

Table D.2: Posterior of the true tree model, $\pi(T^*|x)$, as more data become available.

	$n = 1,000$	$n = 2,500$	$n = 5,000$	$= 10,000$
$D = 3, M = 10$	0.07	0.43	0.89	0.99
$D = 4, M = 10$	0.07	0.43	0.90	0.99
$D = 5, M = 10$	0.00	0.42	0.90	0.99
$D = 5, M = 100$	0.00	0.42	0.90	0.99

In specific, for all values of M and D, with $n = 1,000$ observations the MAP tree model is always the empty tree, corresponding to a single ARCH model. In this case, the posterior of the true tree is still small, as there are not enough data yet to support a more complex model. For $D = 5$, there is a small difference in $\pi(T^*|x)$ compared to $D = 3,4$, which might suggest that there are also not enough data yet for the Laplace approximations of (6.15) to be good. With $n = 2,500$ observations the MAP tree model is now the true tree, with a posterior probability of $\pi(T^*|x) \approx 0.42$, and with all values of D effectively giving identical results. With $n = 5,000$ and $n = 10,000$ observations, the posterior of the true tree becomes respectively 0.90 and 0.99, for all values of D. Increasing the number of Fisher iterations from $M = 10$ to $M = 100$ and trying different initialisations also gave identical results in all cases, suggesting that $M = 10$ iterations are enough for our purposes.

The true parameters of the complete BCT-ARCH model (left), and the estimates from $n = 5,000$ observations with $M = 10$ and $D = 5$ (right) are given below; note that this is a difficult example, as the ARCH models in the two regions are very similar. These results with our final choice of $M = 10$ and $D = 5$ are the ones shown in Section 6.3.5 of the main text.

$$\sigma_n^2 = \begin{cases} 0.10 + 0.20\, x_{n-1}^2 + 0.20\, x_{n-2}^2 \\ 0.10 + 0.20\, x_{n-1}^2 \end{cases}, \quad \widehat{\sigma}_n^2 = \begin{cases} 0.10 + 0.20\, x_{n-1}^2 + 0.16\, x_{n-2}^2, & \text{if } x_{n-1} \leq 0, \\ 0.10 + 0.21\, x_{n-1}^2 + 0.02\, x_{n-2}^2, & \text{if } x_{n-1} > 0. \end{cases}$$

As a second example, we consider the binary tree with depth $d = 2$ and leaves $\mathcal{S} = \{1, 01, 00\}$. The true model (left) and the model fitted from $n = 5,000$ observations with $M = 10$ and $D = 5$ (right) are given below, with the posterior of the true tree being $\pi(T^*|x) \approx 0.98$.

$$\sigma_n^2 = \begin{cases} 0.20 + 0.30\, x_{n-1}^2 + 0.10\, x_{n-2}^2 \\ 0.10 + 0.20\, x_{n-1}^2 + 0.10\, x_{n-2}^2 \\ 0.10 + 0.20\, x_{n-1}^2 \end{cases}, \quad \widehat{\sigma}_n^2 = \begin{cases} 0.20 + 0.30\, x_{n-1}^2 + 0.13\, x_{n-2}^2, & \text{if } s = 00, \\ 0.10 + 0.21\, x_{n-1}^2 + 0.14\, x_{n-2}^2, & \text{if } s = 01, \\ 0.10 + 0.20\, x_{n-1}^2 + 0.00\, x_{n-2}^2, & \text{if } s = 1. \end{cases}$$

Overall, it can be concluded that the concentration of the model posterior to the true tree, together with the convergence of the parameter estimates, both indicate that the BCT-ARCH inferential framework with $M = 10$ and $D = 5$ is very effective.

FTSE 100. The dataset `ftse` is a real dataset, consisting of $n = 7,821$ daily observations of the most commonly used UK-based stock market indicator, FTSE 100 (Financial Times Stock Exchange 100 Index), in a time period of thirty years up to 7 April 2023. It is available from Yahoo Finance, at https://finance.yahoo.com/quote/^FTSE/. The MAP BCT-ARCH model in this example is given in Section 6.3.5 of the main text.

CAC 40. The dataset `cac40` is a real dataset, consisting of $n = 7,821$ daily observations of the most commonly used French stock market index, CAC 40 (Cotation Assistée en Continu), in a time period of thirty years up to 7 April 2023. It is available from Yahoo Finance, at https://finance.yahoo.com/quote/^FCHI/. As in all examples in this section, we consider the transformed log-differenced time series, $y_n = 10\log(x_n/x_{n-1})$.

The MAP tree model in this example is given in the main text, in Figure 6.4. It has depth $d = 3$, four leaves, $\mathcal{S} = \{0, 10, 110, 111\}$, and a posterior of 63.5%. The complete BCT-ARCH model is given by,

$$\sigma_n^2 = \begin{cases} 0.01 + 0.16\, y_{n-1}^2 + 0.17\, y_{n-2}^2 + 0.24\, y_{n-3}^2 + 0.14\, y_{n-4}^2 + 0.09\, y_{n-5}^2, & \text{if } s = 0, \\ 0.01 + 0.00\, y_{n-1}^2 + 0.21\, y_{n-2}^2 + 0.15\, y_{n-3}^2 + 0.13\, y_{n-4}^2 + 0.16\, y_{n-5}^2, & \text{if } s = 10, \\ 0.00 + 0.08\, y_{n-1}^2 + 0.04\, y_{n-2}^2 + 0.32\, y_{n-3}^2 + 0.11\, y_{n-4}^2 + 0.13\, y_{n-5}^2, & \text{if } s = 110, \\ 0.00 + 0.06\, y_{n-1}^2 + 0.09\, y_{n-2}^2 + 0.04\, y_{n-3}^2 + 0.15\, y_{n-4}^2 + 0.07\, y_{n-5}^2, & \text{if } s = 111. \end{cases}$$

DAX. The dataset `dax` is a real dataset, consisting of $n = 7,821$ daily observations of the most commonly used German stock market index, DAX (Deutscher Aktienindex), in a time period of thirty years up to 7 April 2023. It is available from Yahoo Finance, at https://finance.yahoo.com/quote/^GDAXI/. As before, we consider the transformed log-differenced time series, $y_n = 10\log(x_n/x_{n-1})$.

The MAP tree model in this example is the same with the CAC 40 index (Figure 6.4), but now with a posterior of 48.6%. The estimated ARCH coefficients are also very similar, with the complete BCT-ARCH model given by,

$$\sigma_n^2 = \begin{cases} 0.01 + 0.14\, y_{n-1}^2 + 0.19\, y_{n-2}^2 + 0.22\, y_{n-3}^2 + 0.19\, y_{n-4}^2 + 0.12\, y_{n-5}^2, & \text{if } s = 0, \\ 0.01 + 0.00\, y_{n-1}^2 + 0.24\, y_{n-2}^2 + 0.19\, y_{n-3}^2 + 0.13\, y_{n-4}^2 + 0.16\, y_{n-5}^2, & \text{if } s = 10, \\ 0.01 + 0.02\, y_{n-1}^2 + 0.05\, y_{n-2}^2 + 0.28\, y_{n-3}^2 + 0.06\, y_{n-4}^2 + 0.10\, y_{n-5}^2, & \text{if } s = 110, \\ 0.01 + 0.00\, y_{n-1}^2 + 0.08\, y_{n-2}^2 + 0.04\, y_{n-3}^2 + 0.13\, y_{n-4}^2 + 0.14\, y_{n-5}^2, & \text{if } s = 111. \end{cases}$$

S&P 500. The dataset `s&p` is a real dataset, consisting of $n = 7,821$ daily observations of Standard and Poor's 500 Index (S&P 500), in a time period of thirty years up to 7 April 2023. It is available from Yahoo Finance, at https://finance.yahoo.com/quote/^GSPC/. Again, we consider the transformed time series, $y_n = 10\log(x_n/x_{n-1})$.

The MAP tree model in this example is given in the main text, in Figure 6.4. It has depth $d = 3$, five leaves, $\mathcal{S} = \{00, 01, 10, 110, 111\}$, and posterior 91.4%. The complete BCT-ARCH model is given by,

$$\sigma_n^2 = \begin{cases} 0.00 + 0.12\, y_{n-1}^2 + 0.43\, y_{n-2}^2 + 0.17\, y_{n-3}^2 + 0.29\, y_{n-4}^2 + 0.13\, y_{n-5}^2, & \text{if } s = 00, \\ 0.00 + 0.20\, y_{n-1}^2 + 0.05\, y_{n-2}^2 + 0.15\, y_{n-3}^2 + 0.18\, y_{n-4}^2 + 0.19\, y_{n-5}^2, & \text{if } s = 01, \\ 0.00 + 0.06\, y_{n-1}^2 + 0.27\, y_{n-2}^2 + 0.19\, y_{n-3}^2 + 0.16\, y_{n-4}^2 + 0.11\, y_{n-5}^2, & \text{if } s = 10, \\ 0.00 + 0.09\, y_{n-1}^2 + 0.10\, y_{n-2}^2 + 0.24\, y_{n-3}^2 + 0.14\, y_{n-4}^2 + 0.15\, y_{n-5}^2, & \text{if } s = 110, \\ 0.00 + 0.01\, y_{n-1}^2 + 0.14\, y_{n-2}^2 + 0.03\, y_{n-3}^2 + 0.20\, y_{n-4}^2 + 0.12\, y_{n-5}^2, & \text{if } s = 111. \end{cases}$$

D.3 Empirical running times

In this section, we report empirical running times for all the methods used in the forecasting experiments in Chapter 6. All experiments were carried out on a common laptop.

D.3.1 BCT-AR experiments

As discussed in in detail in Section 6.2.2, an important advantage of the BCT-AR framework is that the associated algorithms allow for very efficient sequential updates, making it very practical for online forecasting applications. This is in sharp contrast with DeepAR and N-BEATS, whose current implementation in GluonTS does not allow for incremental training, so the models are re-trained in each timestep from scratch; something which is computationally very costly as it involves gradient optimisation. The classical statistical approaches lie somewhere in-between the two extremes. They need to be re-trained at every timestep using the corresponding `R` package, but the cost required per timestep is lower than that of the ML methods; see also Makridakis et al. (2018b) for a relevant review comparing the computational requirements of ML versus statistical techniques.

The above remarks were verified in practice in our experiments, as it can be observed from the results of Table D.3, where empirical running times are reported for the forecasting experiments of Section 6.2.5. First, it is observed that because of its efficient sequential updates, the BCT-AR model clearly outperforms all the benchmarks in terms of empirical running times. Also, as discussed in Section 6.2.4, the MSA model has much higher computational requirements compared to the other classical statistical methods, as the presence of the hidden state process makes inference much harder; this was also verified in practice here. Finally, the only method that was found to achieve somewhat comparable performance with BCT-AR in terms of running times was ETS, but BCT-AR was found to perform much better in terms of MSE; see also Section 6.2.5.

Table D.3: Empirical running times in the BCT-AR experiments*

	BCT-AR	ARIMA	ETS	NNAR	DeepAR	N-BEATS	MSA	SETAR	MAR
`sim_1`	**7.4 s**	68 s	31 s	4.9 min	2.4 h	7.4 h	45 min	81 s	19 min
`sim_2`	**8.1 s**	61 s	23 s	3.2 min	2.1 h	6.3 h	24 min	98 s	2.5 min
`sim_3`	**2.4 s**	28 s	5.2 s	48 s	1.0 h	4.0 h	12 min	11 s	2.7 min
`unemp`	**3.1 s**	42 s	11 s	69 s	1.0 h	4.1 h	16 min	17 s	6.4 min
`gnp`	**2.2 s**	80 s	10 s	91 s	1.5 h	5.2 h	17 min	19 s	6.6 min
`ibm`	**4.6 s**	58 s	16 s	32 s	2.2 h	5.3 h	22 min	28 s	7.6 min

D.3.2 BCT-ARCH experiments

In Table D.4, empirical running times are reported for the forecasting experiments of Section 6.3.5. Similarly with the BCT-AR model, because of its efficient sequential updates and its low computational complexity, the BCT-ARCH model is also found to outperform all the alternatives in terms of empirical running times. The family of ARCH models and their extensions follow, as they need to be re-trained at every timestep using the `rugarch` package. Finally, the MSGARCH and SV models perform much worse in terms of running times, as in these cases the randomness present in the hidden state process makes inference much more challenging and expensive compared to the other approaches; see also Section 6.3.4.

Table D.4: Empirical running times in the BCT-ARCH experiments*

	BCT-ARCH	ARCH	GARCH	GJR	EGARCH	MSGARCH	SV
`ftse`	**4.1 s**	5.5 min	2.5 min	4.8 min	5.0 min	35 min	1.1 h
`cac40`	**4.8 s**	6.3 min	3.3 min	7.2 min	5.9 min	47 min	1.2 h
`dax`	**4.5 s**	6.9 min	3.0 min	5.9 min	6.2 min	39 min	1.0 h
`s&p`	**4.2 s**	5.2 min	2.6 min	4.6 min	4.7 min	43 min	1.2 h

*The empirical running times of all benchmarks could be reduced if the models were not re-trained at every timestep (i.e., as the corresponding software packages do not allow for sequential updates), but at the expense of prediction performance.

List of Figures

List of Tables

Nomenclature

Basic Notation

$\log$	Natural logarithm
$\log_2$	Binary logarithm
$\max,\ \max^{(i)}$	Maximum value, and ith largest value of a function (of a discrete argument)
$\partial/\partial\theta$	Partial derivative(s) with respect to θ (scalar/vector)
$\partial^2/\partial\theta^2$	Second-order partial derivative(s) with respect to θ (scalar/vector)
$O(\cdot)$	Big-O notation
$o(\cdot)$	Little-o notation
$P(\cdot)$	Probability distribution on a discrete space, such as a finite alphabet
$p(\cdot)$	Probability density on a continuous space, such as $\mathbb{R}$
X_i^j	Vector of random variables $(X_i, X_{i+1}, \ldots, X_j)$, for $i \leq j$.
x_i^j	Vector of observations $(x_i, x_{i+1}, \ldots, x_j)$, for $i \leq j$.
$\{X_n\}$	Discrete-time stochastic process
A^{-1}	Inverse of matrix A
$A^{\mathrm{T}},\ b^{\mathrm{T}}$	Transpose of matrix A / vector b
$\mathbb{N}$	Natural numbers
$\mathbb{R}$	Real numbers
$\mathbb{Z}$	Integers

Probability

$O_p(\cdot)$ Big-O in probability notation; stochastic boundedness

α_n α-mixing coefficients of a stationary and ergodic process $\{X_n\}$

δ_x Unit mass at point x

$\mathbb{I}(\cdot)$ Indicator function

$\mathbb{E}(\cdot)$ Expectation operator

$\mathbb{P}(\cdot)$ Probability measure

$\mathcal{N}(\mathbf{m}, \Sigma)$ Multivariate normal distribution with mean vector $\mathbf{m}$ and covariance matrix Σ

$\mathcal{N}(\mu, \sigma^2)$ Normal distribution with mean μ and variance σ^2

$\phi_\sigma(z)$ Density of a zero-mean Normal distribution with variance σ^2

$\pi(\cdot)$ Prior/posterior distribution

π Stationary distribution of Markov chain (alternate use)

$\sigma(X)$ σ-algebra generated by the random variable X

a.s. Almost surely

CLT Central limit theorem

$\mathrm{Dir}(\alpha)$ Dirichlet distribution with parameters $\alpha = (\alpha_0, \ldots, \alpha_{m-1})$

i.i.d. Independent and identically distributed

Inv-Gamma(τ, λ) Inverse-gamma distribution with parameters (τ, λ)

LIL Law of the iterated logarithm

MAP Maximum *a posteriori* probability

$\widehat{p}(\cdot)$ Empirical distribution

$\xrightarrow{\mathcal{D}}$ Convergence in probability/weak convergence of probability measures

$t_\nu(\mathbf{m}, P)$ Multivariate t-distribution with ν degrees of freedom, mean $\mathbf{m}$, and scale matrix P

$U(0,1)$ Uniform distribution on $(0,1)$

Information Theory

$H(X)$ Entropy of a discrete random variable X, in *nats*

$H(X,Y)$ Joint entropy of random variables X and Y

$H(X|Y)$ Conditional entropy of X given Y

$\bar{H}$ Entropy rate of a discrete-valued stochastic process $\{X_n\}$

$I(X;Y)$ Mutual information between random variables X and Y

$I(X;Y|Z)$ Mutual information between X and Y given Z

$D(p||q)$ Relative entropy between two probability mass functions p,q on the same alphabet

$d_{\chi^2}(p,q)$ χ^2-distance between p and q

$\mathcal{L}$ Log-loss in prediction; redundancy in data compression

Bayesian Context Trees

A Finite alphabet of size m; without loss of generality, taken as $A = \{0,1,\ldots,m-1\}$

m Size of alphabet A

n Number of observations/length of time series x

T Context tree model

D Maximum depth of tree models

$\mathcal{T}(D)$ Collection of all proper m-ary tree models T with depth no greater than D

$\pi_D(T;\beta)$ BCT prior on the model space $\mathcal{T}(D)$; sometimes simply denoted as $\pi(T)$

β Hyperparameter of the BCT prior, taking values in $(0,1)$; default is $\beta = 1-2^{-m+1}$

α Shorthand notation for $(1-\beta)^{1/(m-1)}$

$|T|$ Number of leaves of tree T

$L_D(T)$ Number of leaves of tree T at depth D

$s \in T$ Context s belongs to the set of leaves of tree T

θ Vector of parameters associated at the leaves s of tree T, $\theta = \{\theta_s\ ;\ s \in T\}$

λ Root node; $T = \{\lambda\}$ denotes the empty tree consisting of only the root node

sj Concatenation of context s and symbol j

T_{MAX} Tree constructed to contain all contexts of length D that appear in the time series

a_s Count vector for context s; its elements $a_s(j)$ are defined as the number of times that symbol $j \in A$ follows context s in the time series

$P_e(a_s)$ Estimated probability at node s; simple version for discrete-valued time series

$P_e(s,x)$ General version of the estimated probabilities, for the general BCT-SSM setting with real-valued observations

$P_{w,s}$ Weighted probability at node s

$P_{m,s}$ Maximal probability at node s

$P_{b,s}$ Branching probability at node s

T^* True underlying tree model, when data are generated by a BCT model in $\mathcal{T}(D)$

T_∞ Limiting tree model, when data are generated by a general stationary ergodic process

ℓ Number of change-points in a time series; the maximum possible number is $\ell_{\max}$

$\mathbf{p}$ Vector of locations of change-points in a time series

$x(j;\mathbf{p})$ jth segment of a time series x, as partitioned by the change-points, for $1 \leq j \leq \ell+1$

Q Quantiser from $\mathbb{R}$ to A; used to extract a discrete context from real-valued observations, in the BCT-SSM setting

p Order of AR/ARCH models in the corresponding BCT-AR/BCT-ARCH models

$\widetilde{\mathbf{x}}_{n-1}$ Vector of p previous values in a time series x, $\widetilde{\mathbf{x}}_{n-1} = (x_{n-1},\ldots,x_{n-p})^{\mathrm{T}}$

$\mathbf{z}_{n-1}$ Vector of p previous squared values in a time series x, $\mathbf{z}_{n-1} = (1,x_{n-1}^2,\ldots,x_{n-p}^2)^{\mathrm{T}}$

B_s Set of time indices i, such that the context of observation x_i in the times series x is s

$L_s(\theta_s)$ Log-likelihood of data with context s in the BCT-SSM setting; similarly, its maximiser is denoted $\widehat{\theta}_s$, and the corresponding expected information matrix $\widehat{I}_s = -\mathbb{E}\left(\frac{\partial^2 L_s}{\partial \theta_s^2}\right)$

$\mathcal{S}$ Set of relevant discrete states/leaves of context-tree model in the BCT-SSM setting

Acronyms / Abbreviations (Time series models)

AR Autoregressive model

ARCH Autoregressive conditional heteroscedasticity model

ARIMA Autoregressive integrated moving average model

CTF Conditional tensor factorisations

DeepAR Nonlinear autoregressive model based on recurrent neural networks

EGARCH Exponential GARCH model

ETS Exponential smoothing model

GARCH Generalised autoregressive conditional heteroscedasticity model

GJR Threshold GARCH model of Glosten, Jagannathan, and Runkle

HMM Hidden Markov model

MAR Mixture autoregressive model

MSA Markov switching autoregressive model

MSGARCH Markov switching GARCH model

MTD Mixture transition distribution model; MTDg is the multi-matrix MTD model

N-BEATS Neural basis expansion analysis for interpretable time series forecasting

NNAR Neural network autoregressive model

PST Probabilistic suffix trees

SMC Sparse Markov chains

SSM State space model

SV Stochastic volatility model

TAR Threshold autoregressive model; similarly, SETAR is the self-exciting threshold autoregressive model

VLMC Variable length Markov chains

Acronyms / Abbreviations (Bayesian Context Trees)

BCT Bayesian Context Trees (framework)

BCT Bayesian context tree algorithm (alternate use)

BCT-AR Bayesian context trees autoregressive model

BCT-ARCH Bayesian context trees autoregressive conditional heteroscedasticity model

BCT-SSM Bayesian context trees state space model

CBCT Continuous Bayesian context tree algorithm

CCTW Continuous context tree weighting algorithm

CTW Context tree weighting algorithm

k-BCT Top-*k* Bayesian context trees algorithm

Acronyms / Abbreviations (Datasets)

CAC 40 Cotation Assistée en Continu; major French stock market index

DAX Deutscher Aktienindex; major German stock market index

DNA Deoxyribonucleic acid; genomic DNA data are studied Chapters 2 and 5

FTSE 100 Financial Times Stock Exchange 100 Index; major UK-based stock market index

GNP Gross national product; the quarterly US GNP is studied in Chapter 6

IBM International Business Machines Corporation; its stock price is studied in Chapter 6

S&P 500 Standard and Poor's 500 Index; major US-based stock market index, and one of the most commonly followed indices worldwide

SARS-CoV-2 Severe Acute Respiratory Syndrome Coronavirus 2; virus responsible for the Covid-19 global pandemic, studied in Chapter 2

SV40 Simian virus 40; a commonly studied animal virus

BEA US Bureau of Economic Analysis; data source for US GNP

BLS US Bureau of Labor Statistics; data source for US unemployment rate

Acronyms / Abbreviations (Miscellaneous)

AIC Akaike information criterion

BIC Bayesian information criterion

BOCPD Bayesian online change-point detection

CART Classification and regression trees

EM Expectation-maximisation algorithm

GP Gaussian process

LSTM Long short-term memory neural network

LZ Lempel-Ziv (compression algorithm/entropy estimator)

MC Monte Carlo

MCMC Markov chain Monte Carlo

MDL Minimum description length principle

MH Metropolis-Hastings algorithm

ML Machine learning

MLE Maximum likelihood estimation

MSE Mean squared error

NN Neural network

PPM Prediction by partial matching

RNN Recurrent neural network

RW Random walk MCMC sampler

SGVB Stochastic gradient variational Bayes