Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

School of Computer Science and Statistics

Supervisor: Professor Simon Wilson

Bayesian Tree Regression within a Streaming Context

Michael Ferreira

November 7^{th}, 2023

A thesis submitted to the University of Dublin, Trinity College in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Declaration

I hereby declare that this Thesis is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at `http://www.tcd.ie/calendar`.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at `http://tcd-ie.libguides.com/plagiarism/ready-steady-write`.

Signed: _____ Date: _____

For my mother, Marylyn Ferreira (née Kirwan), who wrote our names in concrete.

Abstract

Regression in a statistical streaming environment. Explore either large amounts of data or data that is continually being generated in a meaningful way. The streaming setting is challenging because either the proportion of data to be analysed far exceeds the available resources or the rate at which the data is arriving and the timeliness of the inference on that data are at odds with each other.

Bayesian methods for streaming regression analysis have focused on using particle filtering and sequential Markov chains. Bayesian regression trees have been used as particles because they offer a tractable approach to nonlinear regression by providing conditional basis functions that can be both smoothed over and still allow for sudden changes in the data to be modelled. The Kalman filter, arguably the progenitor of SMC methods, epitomises the Bayesian methodology for analysis by using data to confirm beliefs which then become the prior beliefs for new data.

MCMC methods have been largely ignored in the statistical streaming setting because ergodic averaging over Markov chains requires that stationarity of the chains of sample measurements be established. Introducing new data invalidates the claim of stationarity requiring that new chains of measures be sampled to re-establish stationarity. What has not been shown is whether MCMC can be used in the streaming setting if one is willing to accept that, at least temporarily, the theoretical requirements for certainty of stationarity be set aside in favour of reaching a target distribution that is, for all intents and purposes, either the same as or very close to the "true" target distribution.

This document sets out to show that, using Bayesian regression trees to provide a collection of conditional filters, MCMC can be used in the streaming setting for nonlinear, nonstationary regression.

A tree filter based on the Kalman filter is developed. This initial stepping stone shows that the explanatory variables are only necessary to indicate a refinement of a partition created by the tree within which a filter provides an estimate and prediction for the level of the signal in that refinement. Thus there is no need to store neither explanatory variables nor observations because, by the Markov assumption, all histories of the processes are retained in the previous state of the latent process at the refinements and in the tree model. This fixed tree filter is then developed into an on-the-fly adaptive learning model that searches the space of tree models for possible models as new data is provided. It is shown that by using Markov chain Monte Carlo it is possible to get sufficiently close to the target distribution having only seen each new data point once. A single tree represents only a single chain limiting the search of the model space so an

ensemble of chains of tree measures is provided so that a more comprehensive search of the distribution of trees can be carried out. An approximation to the probability distribution of the trees is provided by this ensemble. A mixture of tree models over this distribution allows for tree model weighted predictions for the observations and estimates of the state along with their uncertainty estimates to be made on-the-fly.

Showing that MCMC can be used in the streaming setting opens up a whole gamut of MCMC methods for Bayesian statistical analysis that will broaden the scope of problems that could be tackled over large and streaming data sets. This method can be adapted to existing Bayesian tree regression methods and extended to cover variable selection. The independent nature of the trees and the fact that the algorithm has constant complexity with respect to the stream of data means that the size of the ensemble is only limited by available resources and is amenable to both parallel and concurrent computation. Almost any size problem can be explored using this method and, because the Kalman filter can handle vectors with ease, the dimension of the response is of concern only with respect to local (to the leaf filter) matrix manipulation. The model provides a method for autoregressive, on-the-fly Gaussian process regression but is also extendable to multi-output Gaussian process regression.

Acknowledgements

I am uncertain that I could overstate the support that I have been given by my supervisor, Professor Simon Wilson. To be trusted when I made so many mistakes is humbling. Never condescending nor unnecessarily critical even when faced with comments that make me cringe when I think back to them today, must have taken a stoic spirit and strength of will that, in nearly 50 years, I have yet to meet in another.

Jason Wyse was generous with his time and advice, enduring conversations in the coffee room that left me better off, if not himself. Thanks also to Arthur White and Brett Houlding (RIP) for the general chats and pointers for surviving a PhD. Myra O'Regan was another patient being with a love of data who offered sound guidance. I am also thankful to the SCSS support staff who put up with multiple requests for special IT enhancements.

And finally, to all my friends at Liffey Valley Athletics Club for the multiple commiserations and celebrations at the different stages of my failures and successes, I am truly grateful.

Contents

Abstract **iii**

Acknowledgements **v**

0.1 Notation xv

1 Introduction **1**

1.1 Motivation 1

1.2 Problem Statement 4

1.3 Proposition 6

1.3.1 Proposition Statement 6

1.4 Outline 8

1.5 Contributions 9

2 Background **11**

2.1 Introduction 11

2.2 Dynamic Function Approximation 12

2.2.1 Regression and Time Series 13

2.2.2 Filtering and Dynamic Modelling 20

2.3 Trees 27

2.3.1 Trees and Graphs 28

2.3.2 Trees in Statistics 30

2.4 Ensemble Modelling 37

2.4.1 Particle, Additive and Mixture Models 37

2.5 Streaming 44

2.5.1 Data Streams 45

2.5.2 Statistical Streaming 47

2.5.3 Streaming architectures 56

3 Methodology **63**

3.1 Introduction 63

3.2 Bayesian Methodology 64

3.2.1 Probability is a Belief 64

3.2.2 Coherence 65

3.2.3 Modelling . . . 68
3.3 Bayesian Methods . . . 76
3.3.1 Stochastic Processes . . . 76
3.3.2 Markov Chain Monte Carlo . . . 81
3.3.3 Sequential Monte Carlo . . . 84

4 A Regression Model for Streaming Data 87
4.1 Introduction . . . 87
4.2 Model Description . . . 89
4.3 Tree filter . . . 92
4.3.1 2-leaf Model . . . 95
4.3.2 The Probability Model . . . 98
4.3.3 The Marginal Likelihood of the Tree . . . 100
4.4 Calibration Study . . . 108
4.4.1 Introduction . . . 108
4.4.2 Method . . . 109
4.4.3 Results . . . 111
4.4.4 Conclusion . . . 118
4.5 Leaf model simulation studies . . . 120
4.5.1 2-leaf Simulation Study 1 . . . 121
4.6 Summary and Conclusion . . . 130

5 Tree Filter Ensemble 131
5.1 Introduction . . . 131
5.2 Tree Model Randomization and Sampling . . . 133
5.2.1 Tree Prior . . . 134
5.2.2 Modification of Allowable Coordinates . . . 135
5.2.3 Tree Prior Simulations . . . 137
5.2.4 Target Posterior . . . 137
5.3 Random Tree Demos and Simulations . . . 153
5.3.1 Modifying leaf parameters . . . 154
5.3.2 The 2 Leaf Model . . . 158
5.3.3 The CGM Model . . . 163
5.3.4 The Friedman Model . . . 167
5.3.5 Experimental Simulation Study . . . 170
5.4 Tree Ensemble Modelling . . . 171
5.4.1 Predictive Distributions . . . 172
5.4.2 Ensemble Model Demonstration . . . 174
5.4.3 Ensemble Model Comparison . . . 185
5.5 Summary and Conclusion . . . 191

6 Model Modifications **195**
6.1 Introduction . . . 195
6.2 Modifications . . . 195
6.2.1 Modification to Moves . . . 195
6.2.2 More proposals per Iteration . . . 202
6.3 Mackey-Glass Experiment . . . 208

7 Review and Conclusion **217**
7.1 Critical Review . . . 217
7.2 Future Work . . . 221

A1 Appendices **223**
A1.1 Derivation of the Multivariate Marginal Posterior . . . 223
A1.1.1 One dimensional marginal likelihood . . . 236
A1.2 Simulation Study 1.1 . . . 242
A1.2.1 Aim . . . 242
A1.2.2 Method . . . 242
A1.2.3 Results . . . 245
A1.3 Simulation Study 3.1 . . . 246
A1.3.1 Aim . . . 246
A1.3.2 Method . . . 246
A1.3.3 Results . . . 250
A1.3.4 Conclusion . . . 255

List of Figures

4.3.1 Tree and leaf filter output for the 2-leaf model. 97
4.3.2 Tree and Kalman filter output for the 2-leaf model. 97
4.3.3 Tree and leaf filter output for the 2-leaf model. 103
4.3.4 Individual leaf marginal scale parameter components $a_{t,b}$ and $1/a_{t,b}$ 105
4.3.5 Individual leaf marginal residual parameter component $d_{t,b}$ . . 106
4.4.1 Several Kalman filters over the same data, each with a different parameter specification. 109
4.4.2 114
4.4.3 Calibration of the tree marginal residual scale. 115
4.5.1 Ten example densities of the observed and latent distributions simulated from the 2-leaf tree model. 122
4.5.2 The cumulative MSE for 9 tree models. 123
4.5.3 The cumulative MSE for each leaf of the 9 tree models and the Kalman filter when compared to the data generated at each leaf of T_{sim}. 124
4.5.4 The variance and inverse of the scale parameter for each leaf of the 9 tree models and the Kalman filter. 125
4.5.5 Proportion of visits of MSE 126
4.5.6 Tree marginal for each of the experimental trees and for the Kalman filter 127
4.5.7 Leaf marginals for each of the leaves of the 9 experimental trees and for the Kalman filter. 128
4.5.8 Distribution of visits of the leaf marginal parameter $1/[a]_{t,b,b}$. 129
4.5.9 The residuals of the leaf marginals, $[d]_{t,b,b}$. 129

5.2.1 The tree generating process as a random walk with probabilistic reflective upper barrier. 138
5.2.2 Frequency of visit to each state conditional on the tree depth and parameters (α, β). 138
5.2.3 The expected size of a tree conditional on the tree depth and parameters (α, β). 139

5.2.4 Expected time spent in each state conditional on the tree depth and parameters (α, β). 139
5.2.5 148
5.2.6 152
5.3.1 155
5.3.2 156
5.3.3 157
5.3.4 Density estimates based on simulated and estimated parameters for the 2 leaf model. 159
5.3.5 Sequences of data simulated from a 2 leaf model and then estimated by a random tree model. 160
5.3.6 161
5.3.7 Estimation of the simulated stream of data including the error bounds. 162
5.3.8 163
5.3.9 Density estimates based on simulated and estimated parameters for the 5 leaf, CGM model. 164
5.3.10 Sequences of data simulated from the CGM model and then estimated by a random tree model. 165
5.3.11 Cumulative error between the CGM model and simulated stream and the Kalman Filter 165
5.3.12 Marginal terms of a single random tree model for the the CGM stream simulation. 166
5.3.13 Estimation of the simulated stream of data including the error bounds. 166
5.3.14 Density estimates based on simulated and estimated parameters for the Friedman model. 167
5.3.15 Sequences of data simulated from the Friedman model and then estimated by a random tree model. 168
5.3.16 169
5.4.1 178
5.4.2 179
5.4.3 Components of the tree marginal and the number of leaves. . 180
5.4.4 182
5.4.5 The CMSE between the simulated state and the estimated state as well as simulated observations and the predictions. 183
5.4.6 The $\log_2(K_t)$ of the chosen leaves for three trees: $1, 5, 10$ . . . 183

5.4.7 Comparison of estimated densities between Dynamic Tree model (DT), the Bayesian additive regression tree model (BART) and the proposed streaming regression model (BTRS) 191
5.4.8 192
5.4.9 Comparison of estimated densities between streaming Dynamic Tree model (sDT), (streaming) Bayesian additive regression tree model (sBART) and the proposed streaming regression model (BTRS) 192
5.4.10 193
5.4.11 A comparison of the cumulative mean square error between the BTRS model, the DT and sDT models as well as the BART and sBART models. 193

6.2.1 The estimated density when the only moves used are the change, grow and prune moves. 199
6.2.2 The sequence of estimates and predictions when only the change, grow and prune moves are used. 200
6.2.3 The estimated density when the only moves used are the change, grow and prune moves. 203
6.2.4 The sequence of estimates and predictions when only the change, grow and prune moves are used. 204
6.2.5 The estimated density when the only moves used are the change, grow and prune moves. 206
6.2.6 The sequence of estimates and predictions when only the change, grow and prune moves are used. 207
6.3.1 Estimated densities for three Mackey-Glass experiments . . . 210
6.3.2 A sample stream from the Mackey-Glass data generator . . . 211
6.3.3 The cumulative means square error for three experiments with the Mackey-Glass sample stream. 212
6.3.4 MCMC component output for Experiment 1. 213
6.3.5 MCMC component output for Experiment 2. 214
6.3.6 MCMC component output for Experiment 3. 215
6.3.7 MCMC analysis for three experiments of Mackey-Glass stream 216

A1.2.1 243
A1.3.1 247
A1.3.2 The density generated by the CGM tree model. 247
A1.3.3 Sequence of data simulated from the CGM model. 248
A1.3.4 251

A1.3.5 The cumulative means square error (CMSE) between the simulated data, the 5 Kalman filters and all of the trees for 3 experiments. 252
A1.3.6 A more detailed view of the cumulative means square error (CMSE) between the simulated data showing all of the trees for Experiments 1,5,3. 252
A1.3.7 The cumulative means square error (CMSE) between the simulated data and Trees 9,1,6,3 for all experiments. 253
A1.3.8 The distribution of the largest leaf number attained by the trees for Experiments 1,3,5,8 over all trees 1 to 9. 254
A1.3.9 The distribution of the largest leaf number attained by the trees for Tree 1,3,6,9 over all experiments 1 to 8. 255

List of Tables

4.4.1 Parameter values for several Kalman filters. 108
4.4.2 Parameter values used for calibration. 110
4.4.3 Bounds for the tree marginal scale parameter $a_{t,b}$ of F_b and W_b where $H_b = 1$ and $V_b = 1$. 111
4.4.4 Bounds for the inverse tree marginal scale parameter $1/a_{t,b}$ for values of F_b and W_b where $H_b = 1$ and $V_b = 1$. 112
4.4.5 Upper and lower bounds for the tree marginal scale parameter $a_{t,b}$ for values of V_b and W_b where $H_b = 1$ and $F_b = 0.95$. . . . 113
4.4.6 Upper and lower bounds for the inverse tree marginal scale parameter $1/a_{t,b}$ for values of V_b and W_b where $H_b = 1$ and $F_b = 0.95$. 113
4.4.7 Upper bound values for $1/a_{t,b}$ for extended values of F_b and W_b shown in Figure 4.4.2a. 114
4.4.8 Values for $1/a_{t,b}$ where table values are the proportional increases in values found by dividing each row of Figure 4.4.2b by is succeeding row. 114
4.4.9 Change in statistical values of y_t as a function of F_b, W_b for $H_b = 1, V_b = 1$ 116
4.4.10 Change in statistical values of y_t as a function of H_b, V_b $F_b = 0.95, W_b = 1$ for 116
4.4.11 Change in statistical values of y_t as a function of H_b, V_b $F_b = 0.5, W_b = 5$ for 117
4.4.12 Change in statistical values of y_t as a function of H_b, V_b $F_b = 0.05, W_b = 0.05$ for 117

5.4.1 Functions that modify the parameters as the trees change shape (grow or prune, grow-shift or prune-shift) for the CGM model 177
5.4.2 Parameters for the initial weak learners of a 10 tree ensemble. 177
5.4.3 Two measures of the differences between the the proposed distribution and the generated data. 184

6.2.1 Initial leaf parameters for the 10 tree ensemble. 196
6.2.2 Function settings that exactly match those of Section 5.4.2.3. 196

6.2.3 Function settings for Experiment 1 where only the noise variance parameter and observation matrix are modified as the tree changes. 197
6.2.4 Function settings for Experiment 3 where functions that modify F_b, W_b have been returned. 197
6.2.5 Function settings for Experiment 4 where the scale parameters for functions that modify F_b, W_b have been decreased. 198
6.3.1 Parameters with high signal-to-noise. 208
6.3.2 Parameters with low signal-to-noise. 209
6.3.3 Functions to modify parameters. 209

A1.2.1 Parameters for the tree simulation. Note the additional parameters G_b and u_b used to generate a constant mean at a fixed value for each of the leaves. 242
A1.2.3 The parameters for the Kalman filter estimation. 244
A1.3.1 Tree parameters ξ_T for each of 8 experiments. 248
A1.3.2 Leaf model parameters ψ_T for the 9 trees. 249
A1.3.3 Leaf parameter modification functions. These change the parameters as a function of the changes in tree shape. 250
A1.3.4 Number of iterations, CMSE and maximum tree leaf number for all trees and Experiments 1 to 4. 256
A1.3.5 Number of iterations, CMSE and maximum tree leaf number for all trees and Experiments 5 to 8. 256

0.1 Notation

Throughout it is assumed that for every distribution $\mathsf{P}(\cdot)$ a density, $\mathsf{p}(\cdot)$, exists so that these two notations can be used interchangeably.

Roman capital letters indicate random variables (unless otherwise noted) and lower case Roman letters indicate realisations of these random variables. For example, Z, Y are random variables and $Z = z$, $Y = y$ are the obtained (realised) values of these random variables.

Greek capital letters will indicate parameters (missing data values) unless otherwise noted. Θ is the largest set of parameters, it is the super set of all parameters in the ensemble of trees.

- $\Xi \subset \Theta$ is the set of parameters for all tree model functions T and $\xi_T \subset \Xi$ are the set of tree model parameters for a particular tree function T.
- $\Psi \subset \Theta$ is the set of parameters for all functions of every terminal node (also called leaf node) of tree function T. $\psi_T \subset \Psi$ is the set of leaf model parameters for a particular tree T and ψ_b, $b \in \{1 \ldots, K_T\} \subset \psi_T$ is the set of parameters for a model in a particular leaf b in tree T.

T is a binary regression tree model. Excluding the rules at the nodes then T can be described as a directed, acyclic graph (DAG), $T = G = (V, E)$, where V is a set of vertices and E is a set of edges. Every $v \in V$ has a parent node $\mathrm{P}(v)$ and a sibling node $\mathrm{S}(v)$. One exception is the root node R_T which has neither parent nor sibling but always has two child nodes $\mathrm{C}_l(v)$, $\mathrm{C}_r(v)$. L_T and I_T are a subsets of V such that $L_T \cap I_T = \emptyset$ and L_T forms a partition of G.

- The elements of L_T, also called parts, leaves or terminal nodes, have discrete index b. Leaves cannot have children but must have both a parent and a sibling. $\deg(b) = 1$, where $\deg(b)$ is the degree of node b. K_T is the size of the partition (number of elements) so that $|L_T| = K_T$.
- The elements of I_T, called the internal nodes of the tree T, have discrete index η. Every $\eta \in I_T$, excluding R_T, has a parent, two children and a sibling (which can be another internal node or a leaf node) and $\deg(\eta) = 3$. $|I_T|$ is the number of internal nodes in tree T and $|I_T| = \iota_T$.
- Each η is associated with a neighbourhood and two subtrees:
  - $\mathrm{N} = \{\mathrm{P}(\eta), \mathrm{S}(\eta), \mathrm{C}_l(\eta), \mathrm{C}_r(\eta)\}$ is the neighbourhood of η.
  - $\mathrm{P}(\eta)$ is the parent node of an internal node and $\mathrm{P}(b)$ is the parent node of a leaf node.

- $\mathrm{S_L}(\eta)$ and $\mathrm{S_R}(\eta)$ are the left and right subtrees of η rooted at $\mathrm{C}_l(\eta)$, $\mathrm{C}_r(\eta)$ the left and right children of η

Either η_R or $\mathrm{R_T}$ can indicate the root of a tree.

$\mathrm{A}(v) = \{\mathrm{P}(v), \mathrm{P}(\mathrm{P}(v)), \ldots, \mathrm{P}(\mathrm{P}(\ldots(\mathrm{P}(v)))), \mathrm{R_T}\}$ is the set of parents of parents of v until the root node is reached, called the ancestors of v. $\mathrm{P_A}(v)$ is the path of nodes described by the vertices and edges that link the members of $\mathrm{A}(v)$.

$d_v = \lceil \log_2(v) \rceil$ indicates the depth of a node in G where $d_{\mathrm{R_T}} = 0$. Then, for any given tree:

- $K_T \in \{2^{d_b}, \ldots 2^{d_b+1} - 1\}$ any leaf node b.
- $\iota_T = K_T - 1$

Index set t can be $t \in \mathbb{N}$ or $t \in \mathbb{R}$. Generally, for some function f, $f(t)$ represents a function of the index but this can be amended to be written as f^t to show that f is a function of all indices from $0 \ldots, t$. Similarly, f could be subscripted by t, f_t, to show that a f is a function of a particular t. In general, unless otherwise noted, $t \in \mathbb{Z}^+$.

$x(t) = (x_1(t), \ldots, x_p(t)) = x_t$ is a vector interchangeably called the input, covariate, independent or explanatory vector of variables. Each $x_j(t)$ is a deterministic function of the index set t. The outputs of functions $x_j(t)$, $j \in \{1, \ldots, p\}$ in general come from a mixed multi-set of possible outputs $\mathcal{X}^p = \mathcal{X}_1 \times \cdots \times \mathcal{X}_p$, such that $x_j(t)$ could be one of: $\mathbb{R}$, an interval value, $\mathbb{Q}$, a ratio value or $\{a, \ldots, z\}$, a mapping from categorical names or words to the letters of the alphabet. For the most part $x_j(t)$ will be considered a real value and, without loss of generality, normalised to be bounded between 0 and 1. $x^t = \{x_i\}_{i=1}^t$ is the set of input vectors indexed from $i = 1, 2, \ldots, t$. x_j^t is the sequence of a particular covariate $x_j \forall 1, \ldots, t$ and x_{ij} is the value of $x_j(i)$ at index i for covariate j.

Let $Y(t)$ be a random function on an index set t. $Y(t) \in \mathbb{R}^n$ is a vector of values called either the measurement or observation. $Y^t = \{Y_i\}_{i=1}^t$ is a collection of random values (a stochastic process) such that $\mathsf{p}\,(\,Y^t\,) = \mathsf{p}\,(\,Y_1 \leq y_1, \ldots, Y_t \leq y_t\,)$ represents the joint distribution for the observations over the index set t. $y(t)$, y_t are realizations of $Y(t)$, Y_t at the t^{th} index of the index set t. $y(i)$, y_i are the realizations of $Y(i)$, Y_i at indices between the initial index value, usually 1 in this document and t, the t^{th} index value. y^t is the realization of the collection Y^t up to index t so that $\mathsf{p}\,(\,Y^t = y^t\,)$ is a particular realization of the joint distribution of $Y(t)$ over index set t.

In a similar vane to $Y(t)$, $Z(t)$ is a random function such that $\mathsf{E}\,[\,Y(t) \mid x(t)\,] =$

$Z(t)$. $Z(t) \in \mathbb{R}^m$ is a vector of values called either the state, hidden or latent variable. $Z^t = \{Z_i\}_{i=0}^t$ is a stochastic process so that $\mathsf{p}\,(\,Z^t\,) = \mathsf{p}\,(\,Z_0 \leq z_0,\,\ldots,\,Z_t \leq z_t\,)$ represents the joint distribution for the latent state over the index set t. $z(t)$, z_t are realizations of $Z(t)$, Z_t at the t^{th} index of the index set t. $z(i)$, z_i are the realizations of $Z(i)$, Z_i at indices between the initial index value, usually 0 for the latent signal, in this document and t, the t^{th} index value. z^t is the realization of the collection Z^t up to index t so that $\mathsf{p}\,(\,Z^t = z^t\,)$ is a particular realization of the joint distribution of $Z(t)$ over index set t.

A data multiset is represented by $D(t) = \{x(i), y(i)\}_{i=1}^t$, $x(i) \in \mathcal{X}$, $Y(i) = y(i) \in \mathbb{R}^n$, $i \in \{1,\,\ldots,\,t\}$. A data point is represented by $d(i) = (x(i), y(i)) \in \{1,\,\ldots,\,t\}$ and $d(t) = (x(t), y(t))$ is the t^{th} realization of the data.

1 Introduction

1.1 Motivation

The motivation behind this research is real-time data analysis. That this is a statistical problem has an intuitive appeal: sensory input to an actuator is a realization of a set of possible inputs that, conditional on the context of the transducers, obtain and the result is some observed phenomenon with some error in measurement. The actual or true state of the sensors and their context is unknown and the observations might be occluded or confounded due to interaction effects; other exogenous factors affecting input phenomena; separation between sensor and actuator or, quite likely, some combination of these. Further, the phenomena acting on the sensors (inputs) might be changing resulting in changes in the observations (outputs) and this may be happening in a partially nondeterministic manner at random time intervals.

Thus the sample is a sequential multiset of observations from a population of input sources, large and complex enough so complete measurement is impossible. The task, given an instance of the sample of observed outputs is to estimate the current true state of the system and to predict future outputs based on this current state.

The reason for pursuing this task is that if one has better information with measured uncertainty within a time frame that is close to the time that phenomena are produced then the actions taken as a result of this information could be better than not having had these estimates and predictions from the most recent epoch. Bifet, Gavald, et al. (2018) present a list of current and future applications including exploring potentially useful "hidden" data, monitoring of sensory data from the "Internet of Things", healthcare, disaster management, resource management etc.

Bifet, Gavald, et al. (2018) also list a set of constraints that arise from the streaming data setting, the foremost being that it is not possible to know exactly the

number of data points in the sample. This is a restriction from the point of view of the usual approach to statistical analysis from a sample data set but it is also an advantage because to measure the uncertainty of a stream of data requires that the number of samples, N, is decoupled from the analysis. This extends the possible applications of streaming data analysis to data sets of extremely large depth N and width, the number of covariates p.

The recursive nature of the Bayesian approach to probability and statistics has a form that is also intuitively appealing: if the prior over a set of parameters is updated by the likelihood to produce a posterior distribution over those parameters then that updated posterior is surely a prior for the parameters that will be updated by future data to produce the next posterior over those parameters and so on *ad infinitum*. The problem with this intuition is that to calculate the posterior distribution with each new round of data requires that the evidence of all the data accumulated so far is taken into account. The calculation of the marginal of the data is the bugbear of Bayesian analysis but has largely been resolved by sampling approaches to integral calculations such as Markov chain Monte Carlo and Sequential Monte Carlo.

Sequential Monte Carlo (SMC) or particle filtering is the current preferred method of updating estimates and making predictions when new data has been obtained. This approach has been shown to converge to the correct posterior distribution but for best results still requires several runs over the data set and other modifications to avoid degeneracy. Markov chain Monte Carlo (MCMC) has generally been eschewed for streaming analysis because convergence of the chain is only guaranteed if the samples can be repeatedly drawn from a fixed data set. However, MCMC has a whole host of associated methodologies that would be a boon to streaming data analysis if it could be shown that the fact that MCMC does not necessarily converge at each instant that new data is added is of negligible consequence.

This is the task that this thesis sets out to show: using Bayesian regression trees of the type designed by Chipman et al. (1998) can Markov chain Monte Carlo be used with streaming data to provide accurate estimates of the state of nature even if the target distribution changes before full convergence of the chain?

Clearly this is a hard problem, not only because of the requirements of MCMC but the nature of the stream of data: separation between source and analyst; non-stationarity of observations and context; non-linearity and sudden changes between inputs and outputs as well as the time variance of observations. The complexity of the setting requires some additional restrictions to be state before

the problem statement is set out.

Firstly, as pointed out by Bifet, Gavald, et al. (2018) and Gama (2010) streaming data analysis is firmly within the data mining or "knowledge discovery" realm. It is largely an exploratory tool and hence inference based on uncertainty measurements may only be temporally and temporarily relevant. Caution should be taken in extending inference beyond the temporal context but this depends also on the type of analysis being performed.

Secondly, the proposed modelling solution will be a stream of data in itself and requires that there are the computational resources available for the type of statistical analysis proposed in this document. The streaming data problem is a problem of its day: it is only recently (in the last decade or so) that the ubiquity of relevant technology and the communication bandwidth as culminated in the possibility of analysing data in close to real-time. This has lead to the desire, feasibility and ultimate usefulness of this type of analysis without which this research, while not redundant, would have much more limited interest.

Thirdly some base assumptions are needed to begin to frame the problem. These assumptions stem from nature of the problem and the first two points previously raised: prior to the availability of the data, in fact, prior to this type of technology even being envisaged, statistical analysis has been founded on the assumption that the sample is represented by a finite data set. This finite multi-set, according to L.J. Savage in his book *The Foundations of Statistics* "makes no formal reference to time" (Savage 1972, p. 10) and that events are "timeless". In Savage's context temporal properties can be used to describe events, for example time can be used as an index for ordering (as will be done in this document), but the events themselves are not particular to the temporal context of their analysis: measuring heights now or in two centuries time may make a difference to the outcome of the measurements but the samples (and to a large extent the sampling processes) themselves will not be affected by the disparity in time.

The statistical streaming setting challenges this view point in several ways. It is assumed that:

1. the sample is a stream of data and hence the (multi-) set of data is unbounded in the positive direction of the ordering (time) index. Thus the number of data points N to be analysed is not known and assumed random;

2. the sources of the data are separated from the analyses of the data, for example by a lossy communication network, and there is limited control over the data sources (although not over the selection of the (potentially large

number) of sources). This restricts the sample to a rough-and-ready format and changes to these source measurements (and methods of measurement) might be out of the hands of the analyst (or client).

3. the lossy communications network and source separation might disguise the temporal and spatial properties of both the inputs and observations.

Thus, unlike in Savage's context, the timeliness of the data and the sampling context are relevant to the analysis because of the nature of the sources, their output and the need to include calendar time to frame the analysis. The first point covers this to some extent but this additional variation and complexity is worth mentioning because it adds to the challenges presented by the streaming setting and hence to evaluation and potential application of the proposed methods.

Guided by this motivation and some assumptions the following section will present the problem statement.

1.2 Problem Statement

Let $Y(t) \in \mathcal{Y}$ be a random function on an index set $t = \mathbb{Z}^+$. t represents the current value of the index set and i represents a value of the index set between its initial value zero and its current value t. The index $t = 0$ is called the initial index and is not observed. Each $Y(t) \in \mathbb{R}^n$ is a vector of values called either the measurement or observation and results from the interaction of some natural phenomenon on a sensor of some kind, implying that the observations (and inputs) are available in digital format. It is assumed that $Y^t = \{Y(t)\}_{i=1}^t$ is a nonstationary process which is exchangeable conditional on the stream of known explanatory variables $x^t = \{x(i)\}_{i=1}^t$.

Let $x(t) = (x_1(t), \ldots, x_p(t))$ be a vector of known digital inputs also called covariates, independent or explanatory variables where each $x_j(t) = x_{tj}$ is the result of a deterministic function of the index set t. The outputs of functions $x_j(t)$, $j \in \{1, \ldots, p\}$ come from a mixed multi-set of possible outputs $\mathcal{X}^p = \mathcal{X}_1 \times \cdots \times \mathcal{X}_p$, such that $x_j(t)$ could be one of: $\mathbb{R}$, an interval value, $\mathbb{Q}$, a ratio value or $\{a, \ldots, z\}$, a mapping from categorical names or words to the letters of the alphabet. For the most part $x_j(t)$ will be considered a real value and, without loss of generality, normalised to be bounded between 0 and 1.

It is assumed that at each t inputs $x(t) = x_t$ explain $Y(t) = Y_t$ in some functional way using $f(\cdot, t)$: $Y_t = f(x_t, t)$. This equivalence relation is not without some error, $\varepsilon(t) = \varepsilon_t$, usually due to a multitude of factors such as measurement error, model uncertainty and incorrect or incomplete specification of explanatory

variables. In the streaming setting the model $f(\cdot, t)$ is itself a function of t because the relationship between Y^t and x^t may also be evolving. Similarly, the error process $\varepsilon^t = \{\varepsilon_t\}_{i=1}$ is an evolving process and the model $f(\cdot, t) = f(\cdot, \varepsilon_t, t)$ is also a function of the error.

There are additional data, call these $\{\Theta(t)\}_{i=0}^t$ which could be processes, that are used to help $\{f(\cdot, t)\}$ describe the relationship between Y_t and x_t at each t but these data are unobserved. The type and structure of these data, called parameters, are based on guesses, beliefs or judgements by the analyst or as a result of the particular modelling choices by the model designer. The central task of the statistical analysis proposed in this paper is to infer the state of nature of the streaming process using the model, functional assumptions supported by the data and accumulated evidence. This state of nature is represented by a latent process $Z^t = \{Z(t)\}_{i=0}^t,\ Z^t \in \Theta$. If the process Z^t is removed from the set of parameters Θ then $\Theta(t)$ represents a set of parameters that can change at some index t but is not, in this document, considered a process such as the random variables Y^t, Z^t, ε^t or the deterministic functions x^t.

In summary the problem to be solved, as described above, is the following:

Definition 1.1 (Problem Statement)**.**

$$Y^t = f(x^t,\, \Theta(t),\, \varepsilon^t, t). \tag{1.1}$$

A set of real-valued, observed and measured random data Y^t are, with some error process ε^t, functionally related via $f(\cdot, t)$ to some deterministic inputs x^t and some additional data $\Theta(t)$. The function, $f(\cdot, t)$, which relates the inputs and outputs, is also uncertain. The task is to predict current and future outputs, $Y^{t+k}, k \in \mathbb{Z}^+$, and estimate or predict the latent observations Z_t using on an adaptive model, $f(\cdot, t+k), k \in \mathbb{Z}^+$.

The rest of this chapter is structured in the following way: the immediate sequel will state a proposition for solving the statement in Definition 1.1. It will present a streaming regression model using the Bayesian methodology and Markov chain Monte Carlo sampling to estimate the state of nature of the observed process and make predictions on both the observed and latent processes. This will be followed by a section that sets out the structure of this thesis in full. The final part of this chapter will state the contributions of this thesis.

1.3 Proposition

The proposed method of understanding and analysing data in a real-time setting is to provide a model for the stream of data which is a stream of data itself. The methodological framework chosen is that of Bayesian probability and statistics which incorporates sampling as a means of accumulating evidence.

1.3.1 Proposition Statement

The proposed solution to Equation (1.1), in the subjective Bayesian context, is to consider the joint distribution of the random processes, in this case Y^t, Z^t and samples from $f(\cdot, t)$, via tree models $T(x^t, \Theta(t), \varepsilon^t, t)$, conditional on the independent variables x^t and some assumed known (or at least deterministically changing) parameters $\Theta(t)$: $\mathsf{P}\,(\,T,\, Y^t,\, Z^t \mid x^t, \Theta(t)\,)$. However this probability model is still too broad an approach because of the sheer number of possible models and parameters. To simplify things it will be assumed that:

1. $f(\cdot, t)$ is approximated by $\hat{f}(\cdot, t) = \mathcal{F}(\cdot, t)$ where $\mathcal{F}(\cdot, t)$ is an ensemble function of tree models $T(x^t, \Theta(t), \varepsilon^t, t)$ where each $T(\cdot)$ is function that uses $x_t \in \mathcal{X}$ at each discrete t to assign $Y_t = y_t$ to a subset of the partition of $\mathcal{X}$ using some parameters.

2. The set of $\Theta(t)$, now called the ensemble parameters, can be partitioned in the following way:

   (a) Let $\Xi(t) \subset \Theta(t)$ be the set of parameters that will be used for providing additional data about the tree models $T(\cdot)$ such that $\xi_T(t) \subset \Xi(t)$, also called tree parameters, are the subset of parameters for model $T(\cdot)$.

   (b) Let $\Psi(t) \subset \Theta(t)$ be the set of parameters necessary for providing additional data for a set of models, $\{g(Z^t, \Psi(t), t)\}$, called leaf models, that are assigned to each of the models $T(\cdot)$. The parameters indicated by $\psi_T(b, t) = \psi_b \subset \Psi(t)$, also called leaf parameters, are the subset of $\Psi(t)$ assigned to the b^{th} part of the partition of $\mathcal{X}$ generated by $T(\cdot)$. ψ_T is the superset of these leaf parameters associated with a particular model $T(\cdot)$ which can also be called the tree leaf parameters or the leaf parameters of tree T.

Note that $\Xi(t) \cap \Psi(t) = \emptyset$: the former are specifically parameters for $T(\cdot)$ and the latter are specifically parameters for $g(\cdot)$. The size of the ensemble, $|\,\mathcal{F}\,|$ is a parameter of $\Theta(t)$, the ensemble parameters.

The following model is proposed as a solution to the problem statement in Defi-

nition 1.1:

Proposition 1.1 (A Streaming Regression Model)**.**

$$\begin{aligned} Y^t &= f(x^t,\, \Theta(t),\, \varepsilon^t, t) \approx \hat{f}(x^t,\, \Theta(t),\, \varepsilon^t, t) = \mathcal{F}(x^t,\, \Theta(t),\, \varepsilon^t, t) \\ &= \sum_{\forall T \in \mathcal{F}} T(x^t,\, \Theta(t),\, \varepsilon^t, t)\, \mathsf{P}\left(T \mid Y^t, x^t,\, \Theta(t),\, \varepsilon^t, t\right) \\ &= \sum_{\forall T \in \mathcal{F}} T(g(Z^t, \psi_T(t), t),\, x^t,\, \varepsilon^t, \xi_T(t))\, \mathsf{P}\left(T \mid Y^t, g(Z^t, \psi_T(t), t),\, x^t,\, \varepsilon^t, \xi_T(t)\right) \end{aligned} \tag{1.2}$$

for every $t \in \mathbb{Z}^+$ where:

- *$\hat{f} = \mathcal{F}$ is an ensemble function with a distribution generated by the models $T(\cdot)$ where $\mathsf{P}(T \mid \cdot)$ is the posterior probability of each model $T(\cdot)$;*
- *$T(g(Z^t, \psi_T(t), t),\, x^t,\, \varepsilon^t, \xi_T(t))$ are random tree models. Each is a conditional indicator function for model $g(\cdot)$.*
- *$g(Z^t, \psi_T(t), t)$ is an intermittent Kalman filter model to estimate the latent process $Z^t \in \mathcal{Z}$ at each leaf of T from the observations Y^t assigned to at least one of the models $g(\cdot)$ by the covariate vector x_t at each t.*
- *$\mathsf{P}(T \mid Y^t, g(Z^t, \psi_T(t), t),\, x^t,\, \varepsilon^t, \xi_T(t))$ is the posterior probability for the tree model and hence the generator of the ensemble distribution.*

Thus, conceptually, the proposal in this document is to provide an ensemble of tree models each of which are able to indicate or point to a set of filters one of which, conditional on the covariate vector at each t, choose a Kalman filter to estimate the level of the state variable. The ensemble attempts to cover the space of possible models over which Y^t might be explained by x^t and, by randomly adapting the tree models using MCMC proposal moves, the tree models can evolve and search the space of possible models. The filters infer the "true" state of nature conditional on the tree model, the covariate x_t and the observed $Y_t = y_t$ at each discrete t. The tree models are probabilistic and statistical (chosen based on observed and latent data) so that it is possible to provide predictions for future outputs via a probability weighted sum of the random basis functions provided by the trees. Showing that this family of models presented in Proposition 1.1 is a solution to the problem statement in Definition 1.1 is the central thesis of this document.

In summary, this document will present the use of an ensemble of tree models within the Bayesian statistical paradigm as a method for solving nonstationary, nonlinear, time-variant stochastic processes in a streaming setting. The section

that follows this will outline how this will be achieved.

1.4 Outline

The problem as presented in the previous section is not a new problem but, given the technological advances, it has now become possible to not only look at old problems in a new way but also to consider new types of problems. The purposes of this section are to provide a structure for approaching the problem so that the genesis of the problem and hence the thesis and its contributions can be substantiated.

Chapter 2 will provide a review of the problem as it is currently understood as well as present the state-of-the-art in how this or similar problems are solved. The chapter will begin by introducing regression and time series analysis and proceed to describe various developments of the Kalman filter and its relationship to the Bayesian framework. The link between regression analysis, times-series analysis and state-space analysis will be established. Hidden Markov models will be discussed as these form a link between state-space estimation and tree models. Trees will be introduced as graphs and their use in partitioning subspaces for statistical analysis will be presented. The section will conclude with a discussion on streaming, its relationship to other fields in computer science and will then focus on streaming data analysis with particular attention given to tree modelling in this setting.

Chapter 3 provides a summary of the main aspects of the Bayesian methodology. This is a necessary chapter because it is the Bayesian approach for updating inference on unknown data within a coherent framework that was the initial inspiration behind this thesis, not-to-mention that this framework provides the theoretical structure upon which this thesis rests. Also presented here will be current methods for solving complex marginal integrals via sampling, specifically Markov chain Monte Carlo and sequential Monte Carlo methods.

Chapter 4 presents the tree filter which is the stepping stone to full nonstationary analysis in the streaming setting. This chapter will elaborate on the model from Proposition 1.1 providing greater clarification for the assumptions made so far and some additional assumptions necessary for showing the model's functionality and validity. The tree filter marginal distribution will be developed and a calibration study for the leaf model parameters produced. This is necessary for both tree and ensemble setup and replaces parameter estimation at the leaf models. Simulation studies will be provided to demonstrate and support properties of the model.

Chapter 5. This is a continuation of the previous chapter and develops the tree filter into a tree filter that can evolve in a streaming setting. This section proceeds to show how a collection of random tree models is mixture model over the distribution of trees and how this mixture is used to perform prediction for the state and observation processes. The chapter will present simulation studies for the prior in the streaming setting and the individual tree models over three different simulated models. The ensemble model will be demonstrated and compared to two other Bayesian Models, "Dynamic trees for learning and design"(Taddy et al. 2011) and "BART: Bayesian additive regression trees"(Chipman et al. 2010) in a static setting where both are known to converge to the true target distribution of the posterior of the tree model. These models will then be used to perform a comparison in a (pseudo-) streaming setting again where they both will be able to converge to the posterior. The section will conclude with comments on the comparison of the models.

Chapter 6 presents some minor modifications to the proposed model. The aim of these model modifications are improve the performance of the model. Demonstrations and simulation studies in this section will be compared with their counterparts in the previous two sections. The estimation and prediction for the Mackey-Glass differential equation is shown.

1.5 Contributions

The contributions of this these broadly follow the chapter outline of the previous section:

- The first contribution is an adaptation of the Intermittent Kalman filter to a tree filter. The tree filter provides an opportunity to filter a complex process or signal by assuming that the support of the signal can be better specified through explanatory variables but without storing neither covariates nor responses. The work in this paper shows how the tree filter out performs the Kalman filter when it is assumed that there is more than one process generating the signal.
- The second contribution is to adapt the random tree model generation process of Chipman et al. (1998) to the streaming setting. A considerable difficulty in performing analysis in the streaming setting is that there is no finite data set which limits choices of the modeller to the provided sample. Adaptations to the tree model include allowing for a infinite number of threshold values and any number of covariates. The tree model is decoupled from the size of the data set so that the computational complexity is

constant in the number data points.

- The third contribution is to combine the all the tree models into an ensemble of models to approximate the tree model space. Each of the tree models is Markov chain so that the ensemble is a set of Markov chains working in parallel to approximate the target distribution which is the "true" distribution of the tree model at each iteration t. That is, at each t each T is unlikely to converge to the target posterior at $t+1$ but by having $|\mathcal{F}|$ Markov chains the target posterior is sampled from more often thus replicating the (assumed) independent samples of the MCMC process. Combining these samples as components in a mixture model for estimating the parameters of the target distribution at each t brings the streaming approach using MCMC close to both the SMC approach in Taddy et al. (2011) and MCMC approach in Chipman et al. (2010) in the static and (pseudo-) streaming setting.
- The final contribution of this thesis is to provide methods for adapting the raw MCMC approach to bring the proposed model closer to the target distribution. These methods can also be used to improve the performance of the model in the streaming setting.

2 Background

2.1 Introduction

The problem to be solved in Definition 1.1 has been approached in Section 1.3.1 as portmanteau of a mixture model, tree models and Kalman filters each of which is a successful and thoroughly developed field in its own right. Before the proposed model can be explained in detail in Section 4.3 and Chapter 5 it is necessary to provide the reader with some background on how these individual approaches to modelling were developed and how or why they they have been selected to form the structure around the model of Proposition 1.1. Furthermore, this composite of models has been placed in the statistical data streaming setting; an area of study that is not necessarily new in its original guise of statistical process control but has become more attractive to contemporary research due to advances in technology and a concurrent growth, at least commercially, in the availability of streaming data sources. Each of the sections that follow this introduction will provide a brief introduction to each of these fields; some of the existing and state-of-the-art modelling approaches used in these fields and a description of how some of their features will be used in the proposed model.

Section 2.2 will show the links between regression and dynamics. The goals of this section are to present the basic idea of regression, its link to time-series and dynamic modelling and then to focus on the intermittent Kalman filter (Sinopoli et al. 2004) which is the approach adopted for modelling sequential data. There are some essential assumptions and stipulations that relate to both dynamic modelling and control theory and these will be expanded upon. The Bayesian approach will be emphasised and there will be will be a brief mention of stochastic differential equations, autoregressive Gaussian processes and jump diffusions as these form part of the broader framework of the proposed model but they are not essential to the presentation of the model in Chapter 4.

Section 2.3 will present a brief general description of trees as graphs and then it will concentrate on trees in a statistical setting. A summary of the development

of the statistical approach to tree modelling with a particular focus on regression will be provided. Using trees for dynamic modelling has also been done before by Taddy et al. (2011) in the framework of particle filtering, which is an alternative to the Markov chain Monte Carlo method that this proposed model uses.

Ensemble approaches to modelling take different forms including the already mentioned particle approach. A successful ensemble approach is the Random Forest (Breiman 2001) and its Bayesian counterpart, BART (Chipman et al. 2010). Mixture models are another well founded approach to covering a complex space of models and it is this approach that will be used to combine the tree models of Equation (1.2).

The final section of this chapter will present the state-of-the-art in the approach to statistical analysis of streaming data. Section 2.5 will present a short introduction to the general streaming setting of which statistical analysis of data streams is a small part. The data streaming setting will be introduced and some necessary modelling assumptions based on this paradigmatic description will be adopted. Many of the current streaming data methods are based on the classical or frequentist setting. This is in part due to the perceived intractability of the Bayesian methodology, in particular Markov chain Monte Carlo, one notion that this thesis will challenge. The section on streaming methods will start with Very Fast Machine Learning (VFML) (Hulten and Domingos 2003), a tool kit for data mining which uses trees to quickly summarise data, and progress on to the development of adaptive Hoeffding trees (Bifet and Gavaldà 2009) and their variants. These methods form a part of, and much of the basis for, Massive Online Analysis (MOA) (Bifet, Gavald, et al. 2018) a system of online machine learning techniques.

2.2 Dynamic Function Approximation

Regression is a form of function approximation that relies on observed past data to learn the form of a function. It is hoped that this functional form can then be a way to predict unobserved or future outcomes. In the statistical streaming setting the past data is arriving almost at the same time that inference on the model, estimation and prediction from the model and model adjustment and evaluation need to occur. This is a challenge that is hard if not impossible to fulfil without some strong modelling assumptions. This section shows how some of these models, modelling assumptions and requirements have been derived and how they have been used for static ordered data analysis. The first subsection will briefly introduce regression and time-series analysis with a focus on autoregression. The

second section will be focused on the Bayesian approach to dynamic and ordered data analysis with a focus on filtering and dynamic models. This subsection will terminate with a detailed look at the intermittent Kalman filter as developed by Sinopoli et al. (2004).

2.2.1 Regression and Time Series

2.2.1.1 Regression

Regression or function approximation was initially developed as a means of solving an under- or over-determined system of linear equations (Stigler 1986). Minimum least squares approximation uses the partial derivatives of the sample residual difference between a measured but random value, call this a dependent variable, response or observation $Y_i = y_i \in \mathbb{R}$, and some other values that are known and not random, call these covariates, inputs or independent variables $X = (x_{ij})$, to find some weights, typically denoted $\beta = \beta_j$ that provide solution coefficients to the system:

$$y_i = \beta_0 + x_{i1}\beta_1 + \cdots + x_{ij}\beta_j + \cdots + x_{ip}\beta_p,\ i \in \{1, \ldots, N\} \tag{2.1}$$

The form of the system of equations in Equation (2.1) when generalised (viz. Equation (2.18)), has the same form as Definition 1.1 and Equation (1.2) where f is a linear function. In the static form the expected value or average of the sample data is assumed to be linear:

$$\mathsf{E}\left[\, y_i \mid \beta, X \,\right] = \beta_0 + x_{i1}\beta_1 + \cdots + x_{ij}\beta_j + \cdots + x_{ip}\beta_p,\ i \in \{1, \ldots, N\} \tag{2.2}$$

and that this is enough to provide a solution to the system in Equation (2.1) with some error σ^2 between the left and right hand sides of the system. The error of the residuals or deviations is assumed to have a Gaussian distribution and so,

$$r_i = y_i - (\beta_0 + x_{i1}\beta_1 + \cdots + x_{ij}\beta_j + \cdots + x_{ip}\beta_p),\ \sim N(0, \sigma_2), \tag{2.3}$$

leads to the familiar form of the linear regression model,

$$Y \sim N(\boldsymbol{X}\beta, \sigma^2), \tag{2.4}$$

using the linearity properties of the Gaussian distribution.

As shown in Draper and H. Smith (1981) and Hastie, Tibshirani, and J. Friedman (2001), the system of residual gradients (RSS) in Equation (2.3) can be written

as a function of weights β,

$$RSS(\beta) = (\boldsymbol{y} - \boldsymbol{X}\beta)^T(\boldsymbol{y} - \boldsymbol{X}\beta)$$

which can be used, by partial differentiation with respect to the parameters β, to find a unique fixed point for $\hat{\beta}$:

$$\frac{\partial RSS}{\partial \beta} = -2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\beta) = 0 \tag{2.5}$$

so that

$$\hat{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{2.6}$$

The estimate for the variance of the estimated $\hat{\beta}$ can be obtained from Equation (2.5) and is given by:

$$\mathsf{Var}\left[\,\hat{\beta}\,\right] = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\sigma^2 \tag{2.7}$$

where

$$\hat{\sigma}^2 = \frac{1}{N-p-1}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.8}$$

is an unbiased estimator based on the data and the fitted values

$$\hat{\boldsymbol{y}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{2.9}$$

The statistical task is to provide a statement about the range of values within which the estimated model parameters $\hat{\beta}$ and $\hat{\sigma^2}$ could be found. Point estimates provided by the above methods are convenient but of little use because changing the data changes the values. The classical approach to statistical inference on model eq. (2.1) is to assume that in fact this is the "true"[1] or "correct" model and then based on the distributional assumptions, test whether these initial assumptions are the correct assumptions (see Hastie, Tibshirani, and J. Friedman (2001, pp. 47–49)).

The general form of a Bayesian regression problem is:

$$\mathsf{p}\,(\,\theta \mid \boldsymbol{y}, \boldsymbol{X}\,) = \frac{\mathsf{p}\,(\,\theta \mid \boldsymbol{X}\,)\,\mathsf{p}\,(\,\boldsymbol{y} \mid \theta, \boldsymbol{X}\,)}{\int_\theta \mathsf{p}\,(\,\boldsymbol{y} \mid \boldsymbol{X}\,)\,d\theta}$$

[1] "true" is placed in inverted commas here and elsewhere because all models are wrong, G. E. P. Box (1976), and these quotes are intended raise the awareness of this fact.

where the task is to learn about the posterior distribution of θ based on likelihood of the parameter $\mathcal{L}(\theta) = \mathsf{p}\,(\,\boldsymbol{y} \mid \theta, \boldsymbol{X}\,)$ supported by the evidence, $\int_\theta \mathsf{p}\,(\,\boldsymbol{y} \mid \boldsymbol{X}\,)\,d\theta$. The conditioning variables $\boldsymbol{X}$ are assumed known and hence complete so that the conditional distribution of $\boldsymbol{y} \mid \boldsymbol{X}$ can be analysed to learn about θ (A. Gelman et al. 2013, p. 354). This approach to inference is the approach taken in this thesis and more about it will explained in Chapter 3.

In the standard linear model the parameters that need prior distributional statements are $\theta = (\beta, \sigma^2)$. The non-informative approach is to assume that

$$\mathsf{p}\left(\,\beta, \sigma^2\,\right) \propto \sigma_{-2}$$

which is the same as assuming that $(\beta, \log\sigma)$ are jointly uniform (A. Gelman et al. 2013, p. 355). The method used to make posterior statements about these parameters is to first determine the conditional posterior of $\beta \mid \sigma$ and then to consider the marginal distribution for σ_2 via the factorization $\mathsf{p}\,(\,\beta, \sigma^2 \mid y\,) = \mathsf{p}\,(\,\beta \mid \sigma^2, y\,)\,\mathsf{p}\,(\,\sigma^2 \mid y\,)$. So, conditional on σ_2,

$$\beta \sim N(\hat{\beta}, \mathsf{Var}\left[\,\hat{\beta}\,\right]) \tag{2.10}$$

and then

$$\mathsf{p}\,(\,\sigma_2 \mid y\,) \sim \text{Inv-}\chi^2(n-p, \hat{\sigma}^2) \tag{2.11}$$

A necessary assumption for regression modelling in the Bayesian setting is conditional exchangeability which intuitively means that, conditional the $x_i = \sum_{j=1}^{p} x_{ij}$ the order of the y_i in the sample does not matter. More will be said about this in Section 3.2.3.1. Another assumption necessary in the above linear regression model is that the deviation of y_i from x_i, $r_i = \sigma$, is the same for every realisation in the sample and hence in the population.

The above model in Equation (2.4) is simplistic in the sense that it assumes that all variances are equal for every $i, i \in \{1, \ldots, N\}$ and that there are no correlations between $i, j, i \neq j$. Extending that model to include correlations and unequal variances requires the specification of a variance-covariance matrix $\Sigma_y \in \mathbb{R}^{N \times N}$ which means that

$$y \sim N(\boldsymbol{X}\beta, \Sigma_y) \tag{2.12}$$

and

$$\hat{\beta} = (\boldsymbol{X}^T \Sigma_y^{-1} \boldsymbol{X})^{-1} \boldsymbol{X}^T \Sigma_y^{-1} \boldsymbol{y}. \tag{2.13}$$

with

$$\mathsf{Var}\left[\, \hat{\beta} \,\right] = (\boldsymbol{X}^T \Sigma_y^{-1} \boldsymbol{X})^{-1} \tag{2.14}$$

because for every i, j it is necessary to consider how i influences j in the estimates of the posterior parameters (see A. Gelman et al. 2013, pp. 370–371).

To make predictions from the model in Equation (2.12) let $\tilde{X}$ be a collection of new covariates without associated observations and let $\tilde{y}$ be a vector of predictions of interest. Then it is necessary to specify a joint variance-covariance matrix for the old observations and the new predictions. That is, it is necessary to specify $\Sigma_{\tilde{y}}$ which explains how the predictions might vary together and it is necessary to specify how the these predictions might vary together conditional on the known data with variance-covariance Σ_y (A. Gelman et al. 2013, p. 371).

Let

$$\begin{pmatrix} y \mid \boldsymbol{X}, \theta \\ \tilde{y} \mid \tilde{\boldsymbol{X}}, \theta \end{pmatrix} \sim N \left[\begin{pmatrix} \boldsymbol{X}\beta \\ \tilde{\boldsymbol{X}}\beta \end{pmatrix}, \quad \begin{pmatrix} \Sigma_y & \Sigma_{y,\tilde{y}} \\ \Sigma_{\tilde{y},y} & \Sigma_{\tilde{y}} \end{pmatrix} \right] \tag{2.15}$$

be the joint distribution Gaussian distribution between the current data y and the predictions $\tilde{y}$ conditional on the current data $\boldsymbol{X}$, the current parameters θ and the new covariates $\tilde{\boldsymbol{X}}$ Then it can be shown that the conditional distribution of the predictions based on the current data is Gaussian with parameters:

$$\mathsf{E}\left[\, \tilde{y} \mid y, \beta, \Sigma_y \,\right] = \tilde{\boldsymbol{X}}\beta + \Sigma_{\tilde{y},y} \Sigma_{\tilde{y}}^{-1} (y - \boldsymbol{X}\beta) \tag{2.16}$$

$$\mathsf{Var}\left[\, \tilde{y} \mid y, \beta, \Sigma_y \,\right] = \Sigma_{\tilde{y}} - \Sigma_{\tilde{y},y} \Sigma_{\tilde{y}}^{-1} \Sigma_{y,\tilde{y}} \tag{2.17}$$

The result in Equations (2.16) and (2.17) is a powerful one as it used in the Bayesian formulation of the Kalman Filter by Meinhold and N. D. Singpurwalla (1983), for Gaussian processes in Rasmussen and Williams (2006) and for stochastic differential equations in Särkkä and Solin (2019).

The form of the model in Equation (2.4) is extremely flexible and a relevant extension to the linear model is to assume that the vector of covariates is vector of functions:

$$\mathsf{E}\left[\, Y \mid x \,\right] = \mu(x) = \sum_{f=1}^{\mathcal{F}} \beta_f h_f(x) \tag{2.18}$$

where each of the $h_f(x)$ is a called a basis function model. More about this modelling approach will be explained in Section 2.4. This section will now present time series modelling and in particular auto-regressive modelling as an extension to regression over ordered indices.

2.2.1.2 Time-series

Time series analysis involves modelling a sequence of data that is dependent on the data ordering. In this section, and for most of this document, the ordering principle is time in the positive direction and this is denoted t. The data in a time-series model is considered a single realisation of a stochastic process. The aim of time-series analysis is to understand the properties, for example autocovariance, variance and mean of the process. The canonical reference on classical time-series modelling is *Time Series Analysis: Forecasting and Control* (G. E. P. Box et al. 2008) and most of the material in this section is drawn from this reference. Section 2.2.2 will present the Bayesian approach to forecasting and control as presented in *Bayesian Forecasting and Dynamic Models* (Harrison and West 1999). This will also include the Kalman filter which is a form of the linear filter which forms the basis of classical time series modelling.

Consider Equation (2.1) but modify it by indexing the system over t rather than i, letting the response $y_t = z_t$, each of covariates be a previous, and hence known, response value, $x_t, j = z_{t-j}$, $j \in \{1, \ldots, p\}$ and let $\beta_j = \phi_j$ be the parameter that describes the weight of the contribution of the previous observed values to the current observed value:

$$z_i = \phi_1 z_{t-1} + \cdots + \phi_j z_{t-j} + \cdots + \phi_m z_{t-p},\ i \in \{1, \ldots, N\}. \tag{2.19}$$

Further, let each observed value z_t be subject to a random Gaussian shock, $z_t \sim N(0, \sigma^2)$, at each t. Then one has the autoregressive model of order p from G. E. P. Box et al. (2008, p. 9).

Introducing the backshift operator B such that $\mathrm{B}z_t = z_{t-1}$ then Equation (2.19) can be written as:

$$\begin{aligned} &\phi(\mathrm{B})z_t = a_t \quad \text{where} \\ &\phi(\mathrm{B}) = 1 - \phi_1\mathrm{B} - \phi_2\mathrm{B}^2 - \ldots \phi_p\mathrm{B}^p \text{ and} \\ &a_t = N(0, \sigma^2). \end{aligned} \tag{2.20}$$

Stationarity of a stochastic process is a statement about the distributional properties of the process. More will be said about this in Section 3.3.1.1 but in its

strongest form a stationary process requires that the mean, variance and auto-covariance of the process do not change over all t. For Equation (2.20) to be stationary requires that $|\phi| < 1$ (G. E. P. Box et al. 2008, p. 10).

Another useful and popular model is the moving average model of order q,

$$\begin{aligned} z_t &= a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \cdots - \theta_q a_{t-q} \\ &= \theta(\mathrm{B}) a_t \quad \text{where} \\ \theta(\mathrm{B}) &= 1 - \theta_1 \mathrm{B} - \theta_2 \mathrm{B}^2 - \ldots \theta_q \mathrm{B}^q. \end{aligned} \tag{2.21}$$

The random observation at t is deemed to be a finite linear sum of weighted random shocks $a_t, \ldots, a_{t-q}$. In G. E. P. Box et al. (2008, p. 10) it is shown that by repeated substitution of $z_{t-j} = \phi z_{t-j-1}$ say m times that there is an equivalence between the moving average and autoregressive models.

The combination of both the moving average and autoregressive models leads to an ARMA(p,q) model:

$$\begin{aligned} &z_t = \phi_1 z_{t-1} + \cdots + \phi_p z_{t-p} - a_t - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q} \text{ or} \\ &\phi(\mathrm{B}) z_t = \theta(\mathrm{B}) a_t \end{aligned} \tag{2.22}$$

which is a stationary model with wide application usually of order less than 2 for both p and q.

For a nonstationary model define the generalised autoregressive operator

$$\varphi(\mathrm{B}) = \phi(\mathrm{B})(1 - \mathrm{B})^d \tag{2.23}$$

where $\phi(\mathrm{B})$ is as before. For the autoregressive model it was stated that to model a stationary process requires that $|\phi| < 1$ which is equivalent to stating that the roots of Equation (2.20) must lie outside of the unit circle. If roots of Equation (2.20) lie within the unit circle an explosive or divergent process is the result of the model (G. E. P. Box et al. 2008, p. 95). In Equation (2.23) d represents the roots Equation (2.20) that lie on the unit circle.

The ARIMA model of order (p, d, q) is defined as a model that can represent a homogenous, nonstationary process via:

$$\varphi(\mathrm{B}) z_t = \phi(\mathrm{B})(1 - \mathrm{B})^d = \theta(\mathrm{B}) a_t$$

or equivalently,

$$\phi(\mathrm{B})w_t = \theta(\mathrm{B})a_t \text{ where } w_t = (1-\mathrm{B})^d z_t = \nabla^d z_t \tag{2.24}$$

and so

$$w_t = \phi_1 w_{t-1} + \cdots + \phi_p w_{t-p} - a_t - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q}$$

Thus, by choosing polynomial of order d, the autoregressive component of the model is differenced across d time intervals. In the discrete approach described here this is akin to differentiating the time series if $d = 1$.

All of the above models can be represented as a linear filter. Let

$$\begin{aligned} z_t &= \mu + a_t + \psi_1 a_{t-1} + \psi_2 a_{t-2} + \ldots \\ &= \mu + \psi_(\mathrm{B})a_t \quad \text{where} \\ \psi(\mathrm{B}) &= 1 + \psi_1 \mathrm{B} + \psi_2 \mathrm{B}^2 + \ldots \end{aligned} \tag{2.25}$$

Here, μ represents a constant level for the model and, again the a_t are assumed random Gaussian shocks. $\psi_(\mathrm{B})$ is called the transfer function of the filter (G. E. P. Box et al. 2008, p. 9). If the $\sum_{j=0}^{\infty} |\psi_j| < \infty$ then the linear filter model is called stable. This is a concept that is tied into the ideas of observability, controllability, detectability and stabilizability that are necessary for the adjustment and control of control models that form part of the necessary assumptions for the proposed approach.

Taking $\psi_(\mathrm{B}) = \phi_{-1}(\mathrm{B})$ and the autoregressive model of Equation (2.20) is recovered. Similarly, let $\psi_(\mathrm{B}) = \phi^{-1}(\mathrm{B})\theta(\mathrm{B})$ and the ARMA model of Equation (2.22) follows and finally it can be seen by looking at Equation (2.24) that again if $\psi_(\mathrm{B}) = \phi^{-1}(\mathrm{B})\theta(\mathrm{B})$ then $w_t = \psi(\mathrm{B})a_t$. Thus the linear filter is a powerful way to model data that has correlations over an ordered index.

The section that follows will describe the Kalman filter (Kalman 1960), a state-space model of which the above models are special cases. This will end with the Intermittent Kalman filter (Sinopoli et al. 2004) and will include a description of stability and observability and controllability.

To bring these models into the state-of-the art they will be represented in other form called a Gaussian process. It will be shown that autoregressive Gaussian processes are a natural extension of the above models.

Also of importance is the choice of model parameter F_t in the Kalman filter. In the static time-series approach there are several ways of estimating this parameter

but in the streaming modelling setting a judicious choice of parameter, for at least some duration, reduces the complexity of the streaming model. Särkkä and Solin (2019) provide several alternatives for choosing F and also show how the Kalman filter is form of autoregressive Gaussian process. References to current papers on choosing relevant kernels and multi-output processes will be provided. A contribution of this paper is perform the filtering (and smoothing) solution to these types of models in real-time.

2.2.2 Filtering and Dynamic Modelling

The aim of filtering is to make inference about "the state of nature" based on some observations $y^t = y_1, \ldots, y_t$; $y_t \in \mathbb{R}^n$ (Meinhold and N. D. Singpurwalla 1983). Let the "state of nature" be represented by a latent process $\{Z\}_{i=0}^t = Z^t = Z_1, \ldots, Z_t$; $z_t \in \mathbb{R}^m$, a collection of random variables indexed by a set t that increases in regular steps (is a discrete ordered set $\mathbb{Z}^+$). In the classical filtering setting the process that generates the observations (the observation equation)

$$y_t = Hz_t + v_t; \qquad v_t \sim \mathsf{N}(0, V) \tag{2.26}$$

is a linear function of the state of nature which is disturbed by some Gaussian error or noise process v_t with $\mathsf{E}[\, v_t \,] = 0$ and $\mathsf{Var}[\, v_t \,] = V \in \mathbb{R}^{n \times n}$ so that $Y_t \sim \mathsf{N}(Hz_t, V)$. $H \in \mathbb{R}^{n \times m}$ is called the observation matrix and relates the state process to the observation process. The unobserved or latent process that underlies the observed process is also described by a linear equation (the state equation)

$$z_t = Fz_{t-1} + w_t; \qquad w_t \sim \mathsf{N}(0, W) \tag{2.27}$$

that has Gaussian driving process w_t with $\mathsf{E}[\, w_t \,] = 0$ and $\mathsf{Var}[\, w_t \,] = W \in \mathbb{R}^{m \times m}$ so that $Z_t \sim \mathsf{N}(Fz_{t-1}, W)$. The driving process w_t is assumed to be independent of the noise process and to have independent increments ($w_s \perp w_t$; $\forall s \in t, s < t$). $F \in \mathbb{R}^{m \times m}$ is called either the state matrix or the transition matrix and relates the current state of nature to the previous state.

Meinhold and N. D. Singpurwalla (1983) proceed to relate the recursive estimation process of the Kalman filter (Kalman 1960) to the Bayesian modelling formalism

$$\mathsf{p}(\, Z_t \mid y_t \,) \propto \mathsf{p}(\, Z_t \mid y_{t-1} \,)\, \mathsf{p}(\, y_t \mid Z_t, y_{t-1} \,) \tag{2.28}$$

so that the distribution of the state of nature at time t is the posterior distribution of the state process proportional to the likelihood of the data seen up to time t and

the prior distribution of the current state of nature given the observations prior to index t. A key assumption here is the Markov assumption: all information necessary to transition to random state Z_t is contained in the previous "known" state z_{t-1}. However, z_{t-1} is never actually observed in the filtering case so its state of nature is described by distribution $\mathsf{p}\,(\,Z_{t-1} \mid y_{t-1}\,) \sim \mathsf{N}(\hat{\mu}_{t-1|t-1}, \Sigma_{t-1|t-1})$, which is the posterior of the previous state of nature. To get from $\mathsf{p}\,(\,Z_{t-1} \mid y_{t-1}\,)$ to $\mathsf{p}\,(\,Z_t \mid y_{t-1}\,)$ it is necessary to make a prediction for Z_t which, using the state model equation and standard theory about the Gaussian distribution, has the form:

$$\mathsf{p}\,(\,Z_t \mid y_{t-1}\,) \sim \mathsf{N}(\hat{\mu}_{t|t-1} = F\hat{\mu}_{t-1|t-1},\ \Sigma_{t|t-1} = F\Sigma_{t-1|t-1}F^T + W). \tag{2.29}$$

To get to the distribution for the current state it is necessary to update the estimate of the predicted level of the state $\hat{\mu}_{t|t-1}$ to an estimate of the current level $\hat{\mu}_{t|t}$ and this requires assessing the likelihood of the current level based on new data y_t that has been acquired.

As well as predicting Z_t it is not unreasonable that one might want to predict Y_t before the new data has arrived. Based on the model in Equation (2.26) a prediction for $\hat{y}_t$ is $HF\mu_{t-\hat{1}|t-1}$ but this too has some random error, call this $\varepsilon_t = H(Z_t - F\mu_{t-\hat{1}|t-1})$, a function of Z_t which, because the ε_t are assumed independent, has distribution

$$\varepsilon_t \sim \mathsf{N}(H(Z_t - F\mu_{t-\hat{1}|t-1}), V). \tag{2.30}$$

Meinhold and N. D. Singpurwalla (1983) provided this reasoning to show that to reach the the posterior distribution of the state of nature

$$\mathsf{p}\,(\,Z_t \mid y_t, y_{t-1}\,) = \mathsf{p}\,(\,Z_t \mid y_{t-1}\,)\,\frac{\mathsf{p}\,(\,\varepsilon_t \mid Z_t, y_{t-1}\,)}{\int_{\forall Z_t} \mathsf{p}\,(\,\varepsilon_t \mid Z_t, y_{t-1}\,)\,\mathsf{p}\,(\,Z_t \mid y_{t-1}\,)\,dZ_t} \tag{2.31}$$

one must consider the likelihood of the error at the current index over the accumulation of the errors or residuals from $t = 0$, the index of initial guess for the state of nature which has distribution $Z_0 \sim \mathsf{N}(\mu_0, \Sigma_0)$.

A more direct way to obtain the update of the level parameter $\hat{\mu_{t|t}}$ and its variance, $\Sigma_{t|t}$, is by considering the joint distribution of the error and the state, both of which have a Gaussian distribution. Notice that in Equation (2.31) the likelihood term, is a conditional function of the random state. Meinhold and N. D. Singpurwalla (1983) show that the likelihood term independent of the state variable can be obtained by exploiting the correspondence that results from the well

known property of the Gaussian distribution which is that if two independent conditional distributions are Gaussian the joint distribution of these conditionals is Gaussian[2]. The first conditional Gaussian is the density in Equation (2.29) and the second comes from the likelihood because, as mentioned previously, the errors are considered independent at each index t so that the conditional distribution of ε_t independent from Z_t is $\mathsf{N}(0, V + H\Sigma_{t|t-1}H^T)$ hence

$$\mathsf{p}\left(\begin{bmatrix} Z_t \\ \varepsilon_t \end{bmatrix} \,\middle|\, y_{t-1}\right) \sim \mathsf{N}\left(\begin{bmatrix} \mu_{t|t-1} \\ 0 \end{bmatrix}, \begin{bmatrix} \Sigma_{t|t-1} & \Sigma_{t|t-1}H^T \\ H\Sigma_{t|t-1} & V + H\Sigma_{t|t-1}H^T \end{bmatrix}\right) \tag{2.32}$$

is the joint distribution of the prediction and the likelihood. This joint distribution provides the posterior, which is conditional on both the likelihood and the prior, as

$$\mathsf{p}\left(\, Z_t \mid \varepsilon_t, y_{t-1} \,\right) \sim \mathsf{N}\left(\hat{\mu}_{t|t-1} + \Sigma_{t|t-1}H^T S^{-1}\varepsilon_t, \Sigma_{t|t-1} - \Sigma_{t|t-1}H^T S^{-1} H\Sigma_{t|t-1}\right) \tag{2.33}$$

where

$$S = \left(V + H\Sigma_{t|t-1}H^T\right) \tag{2.34}$$

and

$$K = \Sigma_{t|t-1}H^T S^{-1} \text{ is the Kalman Gain.} \tag{2.35}$$

Therefore the equations to update the estimate of level and its variance are:

$$\hat{\mu}_{t|t} = \hat{\mu}_{t|t-1} + \Sigma_{t|t-1}H^T S^{-1}\varepsilon_t \tag{2.36}$$

$$\Sigma_{t|t-1} = \Sigma_{t|t-1} - \Sigma_{t|t-1}H^T S^{-1} H\Sigma_{t|t-1} \tag{2.37}$$

Meinhold and N. D. Singpurwalla (1983) proceed to point out that the Kalman gain (Equation (2.35)) is equivalent to β of Equation (2.1) if one were to consider Z_t as the dependent variable and ε_t as the dependent variable: "Thus Kalman filtering can also be viewed as the evolution of a series of regression functions of $[Z_t]$ on ε_t, at times $0, 1, \ldots, t-1, t...$; the evolution stems from a learning process involving all the data"(Meinhold and N. D. Singpurwalla 1983, p. 126).

In the original paper by Kalman (1960) the eqs. (2.36) and (2.37) were proposed as solutions to Wiener problem (Kalman 1960, p. 40) and this was achieved by

[2]The converse is not true. Two jointly distributed random variables need not have independent Gaussian distributions. This would only be the case if the covariance terms were zero.

solving the observation and state equations as an convex optimization problem for a discrete linear dynamic system. Indeed, Kalman (1960) shows that because linear regression produces an orthogonal projection of the observation into the covariate space, the estimate of this level is the optimal minimiser of the squared error loss function $L(\mu_t) = \mathsf{E}\left[(Z_t - \mu_t)(Z_t - \mu_t)^T \right] = \Sigma_t$. Kalman and Bucy (1961) update the original paper for continuous indices and provide a "nonlinear differential equation of the Riccati type" (Kalman and Bucy 1961) for the covariance of the state estimate. This implies that there are asymptotic results that can be applied to Kalman filtering which is one of its huge advantages: for a linear dynamic system of the type described by Equations (2.26) and (2.27) not only is $\hat{\mu}_{t|t}$ the optimal estimator of the level of the system but the total variation of system up to small perturbations can be completely determined.

In Humpherys et al. (2012) the Kalman update equations are derived via Newton's method for root finding emphasising the relationship between differential equations and the Kalman filter. This paper also shows a linear system in its full from $t = 0$ to t and from this form it is easy to see the relationship between the Kalman filter and the time series models of Section 2.2.1.2. Indeed, in G. E. P. Box et al. (2008) and Shumway and Stoffer (2017) the state estimation forms a part of time series analysis and the Kalman filter features prominently in these.

The Kalman filter is ubiquitous as a technique in state estimation and has inspired many approaches to solving sequential problems beyond that of linear dynamical systems. The extended Kalman filter considers a first order Taylor approximation to a nonlinear function rather than a linear function for either or both the state or observation equations but for most part retains the Gaussian assumption. The origins of the extended Kalman filter are not clear but it is mentioned as having originated from the Kalman filter via Battin (1964). The unscented Kalman filter (Wan and Van Der Merwe 2000) improves upon the extended Kalman filter by creating a (small) set of sample points around the point of interest with aim of locally approximating the function of the state variable rather than using the linearization of the function as the extended Kalman filter does. These sample points are called sigma points and these points are propagated at each prediction by some transition function of the state variable. A weighted sum of the sigma points provides the prediction estimate and similarly the prediction covariance. For the update step a new set of sigma points and weights is generated and these are propagated through some observation function. These new points provide an empirical estimate of the updated state and covariance function.

Not only has Kalman filtering inspired techniques for filtering it has also served as motivation for statistical techniques beyond that of engineering. Particle filtering

(Gordon et al. 1993) is arguably the basis behind another technique for sample based integration called Sequential Monte Carlo (Doucet et al. 2001). More directly related to the Kalman filter are the methods of Harrison and West (1999) in which, in its most general form, uses a design matrix as the observation matrix and the state equation is used to propagate the change in the regression coefficients over time. Harrison and West (1999) also provides a link between Bayesian dynamic modelling, classical time series analysis (G. E. P. Box et al. 2008) and the techniques of Goldstein and Wooff (2007).

Of particular interest is the Intermittent Kalman filter (Sinopoli et al. 2004) which extends the Kalman filter to jump diffusions. This will be presented in the subsection that follows.

2.2.2.1 The Intermittent Kalman Filter

Motivated by the lack of reliability across wireless networks the Intermittent Kalman Filter (Sinopoli et al. 2004) is designed to explore the implications on control of, for example, an autonomous vehicle when the observations have stochastic arrival rate due to the stochastic nature of the communication channel. Sinopoli et al. (2004) uses the discrete linear Kalman filter as a starting point but modifies it to the case where the parameters of the model, in particular the Kalman gain and covariance matrix of the state estimator, are related to the random nature of the communication channel. Assuming that the observations arrive as a Bernoulli process with parameter $0 < \lambda < 1$ they study the asymptotic average of the state covariance matrix and show that, depending on the eigenvalues of the transition matrix F of Equation (2.27) and the structure of the observation matrix H of Equation (2.26), there is a critical value λ_c for the probability of an update by an observation of the filter below which the state error covariance matrix is unbounded. If the probability is $\lambda_c < \lambda < 1$ and provided that the properties of stabilizability and detectability are met then state error covariance is always finite.

Define the arrival of an observation to update the Kalman Filter as a binary random variable γ_t with probability $\mathsf{p}(\gamma_t = 1) = \lambda_t$, $\gamma_s \perp \gamma_t \, s \neq t$. Then the observation noise can be modelled as:

$$\mathsf{p}(v_t \mid \gamma_t) = \begin{cases} \mathsf{N}(0, V), & \text{if } \gamma_t = 1 \\ \mathsf{N}(0, \sigma^2 I), & \text{if } \gamma_t = 0. \end{cases} \tag{2.38}$$

The approach of Sinopoli et al. (2004) is to create a dummy observation when there is no observation (hence the variance σ^2) and to find the limit as $\sigma \to \infty$.

Summarising and redefining the Kalman filter equations from Equations (2.29), (2.36) and (2.37) they show that for the intermittent Kalman filter the prediction and update level and variance estimates are:

$$\hat{\mu}_{t|t-1} = F\hat{\mu}_{t-1|t-1} \tag{2.39}$$
$$\hat{\Sigma}_{t|t-1} = F\hat{\Sigma}_{t-1|t-1}F^T + W \tag{2.40}$$
$$\hat{\mu}_{t|t} = \hat{\mu}_{t|t-1} + \hat{\Sigma}_{t|t-1}H^T(H\hat{\Sigma}_{t-1|t-1}H^T + \gamma_t V + (1-\gamma_t)\sigma^2 I)^{-1}\varepsilon_t$$
$$\hat{\Sigma}_{t|t-1} = \hat{\Sigma}_{t|t-1} - \hat{\Sigma}_{t|t-1}H^T(H\hat{\Sigma}_{t-1|t-1}H^T + \gamma_t V + (1-\gamma_t)\sigma^2 I)^{-1}H\hat{\Sigma}_{t|t-1}$$

Taking the limit of the second two equations as $\sigma \to \infty$ provides

$$\hat{\mu}_{t|t} = \hat{\mu}_{t|t-1} + \gamma_t K_t \varepsilon_t \tag{2.41}$$
$$\hat{\Sigma}_{t|t-1} = \hat{\Sigma}_{t|t-1} - \gamma_t K_t H\hat{\Sigma}_{t|t-1} \tag{2.42}$$

where

$$K_t = \hat{\Sigma}_{t|t-1}H^T(H\hat{\Sigma}_{t-1|t-1}H^T + V)^{-1} \text{ is the Kalman gain as before.} \tag{2.43}$$

Notice that the covariance between the state estimate components is now estimated rather than a known differential equation. This results from the random observation updates. The prediction equations are unaffected by the random update at each iteration but clearly affected in the future by the recursive nature of the filter.

Sinopoli et al. (2004) modify the algebraic Riccati equation to contain the random variable γ_t and proceed to prove the convergence conditions for this modified Riccati equation. They show that for some special cases, including the scalar case, that if H is invertible and the Kalman Gain $K = FH^{-1}$ then the convergence condition for Equation (2.42), the minimum probability of an update, $\lambda_c = 1 - 1/\alpha^2$ where $\alpha = \max_i |\,\mathrm{u_i}\,|$ and u_i is an eigenvalue of F.

Mo and Sinopoli (2008) compute the value of λ_c and show that under certain conditions on the state and observation matrices, a lower probability of arrival of the process than stipulated in Sinopoli et al. (2004) is possible. For parameter estimation and adaptive control Kar et al. (2012) explore a random Riccati equation that would result from random draws of the state estimation matrix F. They are able show that it is possible to calculate the statistics of a stationary process in this case.

This section has provided a very brief summary of regression and linked this to time series analysis, filtering and intermittent filtering. The Kalman filter is far

more wide reaching than has been displayed here. The approach of both Harrison and West (1999) and Goldstein and Wooff (2007) are fields in their own right and this cursory touch on their contribution to the field does not do them justice. The adaptation of filtering to particle filtering by Gordon et al. (1993) will be broached again in Section 3.3.2 as it forms the basis for Sequential Monte Carlo methods currently used for streaming analysis using trees by Taddy et al. (2011) an alternative approach to the proposed MCMC approach of this thesis.

2.2.2.2 Autoregressive Gaussian processes

Another view on the Kalman filter is as a stochastic differential equation (Särkkä and Solin 2019). The approach is to consider two stochastic processes, $\{Y_t\}$ and $\{Z_t\}$ where the former is an observed process and the latter is not, analogous to the observation and latent state equations above. The statistical objective is to infer the predictive distributions for the latent and observed processes. In this form of model it is often the case that the state is not observed but is expected to follow a physical process. The state transition matrix F_t will typically be of a known form and the state noise process is seen as a Brownian process that drives (forces) the state process through a linear function $Lz_t,\ L \in \mathbb{R}^{r\times m}, z \in \mathbb{R}^m$.

Let the state equation be represented as an Itô stochastic differential equation (Särkkä and Solin 2019, pp. 45, 198):

$$dz = f(z,t)dt + L(z,t)d\beta \tag{2.44}$$

and then the observation equation is modelled as (Särkkä and Solin 2019, p. 198):

$$du = h(z,t) + d\eta \tag{2.45}$$

where $Y_t = \dfrac{\mathrm{d}u}{\mathrm{d}t}$ and $v = \dfrac{\mathrm{d}\eta}{\mathrm{d}t}$.

Equations 2.44 and 2.45 represent a continuous-time state-space model (Särkkä and Solin 2019, p. 199). However, $\{Y_t\}$ is often sampled at discrete but irregular time points which leads to the continuous-discrete state-space model. Discretization invariance, where the performance of the calculations is independent of the sampling times and of the calculation time steps, of the Kalman filter is anther useful attribute of the Kalman Filter.

The reason this is introduced here is that Särkkä and Solin (2019) show the link between Gaussian processes and differential equations which leads to autoregressive Gaussian processes.

2.2.2.3 Hidden Markov Models

Before finishing off this section hidden Markov models (HMMs) (Baum and Petrie 1966) will be briefly described. HMMs are another field that extends well beyond the aspects mentioned here but it is introduced because it is a generalization of both filter modelling and tree modelling and hence unites these two approaches via graphical models (Jordan 1997). The focus here will be on the showing that the Kalman filter and intermittent Kalman filter are HMMs. The link between trees and HMMs forms a part of the main body of the thesis but in Section 2.4 the link between HMMs and mixture/basis function modelling, of which trees are an example, is provided.

If $Z^t = \{Z_1, \ldots, Z_t\}$ is a Markov process and for each Y_t in $Y^t = \{Y_1, \ldots, Y_y\}$, Y_t is assumed independent of $Y_{t-1} \mid Z_t$ then Y^t is called a hidden Markov model or HMM. The term hidden arises because only the process Y^t is observed and the process Z^t is considered to be the true underlying process that drives the observations. Consider the Equations (2.26) and (2.27) and one can immediately see that the state equation is the driving equation for the hidden Markov model that is the observation process.

The next section will consider tree modelling in the statistical setting.

2.3 Trees

Binary trees used in statistical analysis are "a child of the computer age"(Breiman et al. 1984): a method used to for applied problem solving due to their descriptive capabilities. A more formal description of the binary tree model used in this document, also called a full binary tree, is that it is a directed acyclic graph where each node has exactly two children. It is a rooted graph that has either a single node, the root, or each node is itself a full binary tree. Thus a full binary tree is recursive structure made up of weak learners: a single node that provides a boolean choice and two terminal nodes.

The tree model used here is not a graphical model in the sense of Maathuis et al. (2018) because the nodes do not represent random variables. The model used in this document is a statistical model in at least two senses: it summarises the population by providing a set of subsets of summaries of the sample data and the tree model is itself a sample of a population of possible tree models. All tree models from ID3 (J. R. Quinlan 1986), CART (Breiman et al. 1984), random forests (Breiman 2001) and boosting (which uses weak learners) (Robert E. Schapire 1990) and before achieve the first sense of a tree model as a statistical

object, that is as an method for summarising data. Both the first and second sense are achieved by the Bayesian approach to tree modelling as presented by Chipman et al. (1998), Taddy et al. (2011), Chipman et al. (2010) and their derivatives. The modelling approach used in this document uses these latter Bayesian approaches as a jumping off point.

The subsection that follows will briefly present tree models as graphs with the intention of introducing some notation that will be used in subsequent sections. A summary of the notation can be found in Section 0.1. The next section will present the Bayesian approach to tree modelling and introduce the basic ideas of their approaches which are foundation of this approach. A brief summary of the state-of-the-art in Bayesian tree modelling will also be provided.

While ensembles of trees form a part of tree modelling in general descriptions of these models will postponed to Section 2.4 because this fits in with the development of the composite model: filter $\rightarrow$ tree $\rightarrow$ ensemble of trees.

2.3.1 Trees and Graphs

A graph, $G = (V, E)$, is a pair of two sets where V is a set of vertices, $\{v_k : k \rightarrow \mathbb{N}\}$, and E is a set of pairs $\{e_{k,l} = (v_k, v_l) : (v_k, v_l) \rightarrow V \times V\}$. If the pairs $e_{k,l}$ are ordered for all k, l then the graph is a directed graph (DG) otherwise it is an undirected graph (UG).

$S = G = (V, E)$ is a full binary tree. Define a root tree as a single root node (vertex) $v_R = \mathrm{R}_T$ with exactly two child nodes (vertices) $\mathrm{C}_l(v)$, $\mathrm{C}_r(v)$ also called terminal nodes. A full binary tree is a recursive construction of root trees where every subtree is itself a root tree. Every $v \in V$ has a parent node $\mathrm{P}(v)$ and a sibling node $\mathrm{Sib}(v)$. L_S and I_S are a subsets of V such that $L_S \cap I_S = \emptyset$ and these form a partition of S.

If $\mathrm{S_R}$ is the root of a full binary tree then

$$\mathrm{A}(v) = \{\mathrm{P}(v), \mathrm{P}(\mathrm{P}(v)), \ldots, \mathrm{P}(\mathrm{P}(\ldots(\mathrm{P}(v)))), \mathrm{S_R}\}$$

is the set of parents of parents of v until the root node is reached, called the ancestors of v. $\mathrm{P_A}(v)$ is the path of nodes described by the vertices and edges that link the members of $\mathrm{A}(v)$. Each v is associated with a neighbourhood $\mathrm{N}(v) = \{\mathrm{P}(v), \mathrm{Sib}(v), \mathrm{C}_l(v), \mathrm{C}_r(v)\}$.

A statistical tree model T_R begins with a root tree S_R that is mapped to a sample space, defined by the sample data set $D = (y_i, x_{i,j}), i, 1 \ldots N, j, \ldots p$, in the following way. $v \in I_S$ is assigned a rule which is a covariate label $x_j \in \mathcal{X}$

and a value, c_j from the known multiset x_j, called a threshold, such that D is partitioned by the pair, $\text{rule}_\eta = (x_j, c_j)_\eta, \eta \in I_T$ into $D = (D_l, D_r)$ each called parts or leaves of the partition. I_T is the set of internal vertices I_S with an associated rule for each $v \in I_S$. The total number of possible rules for a root tree model is $p \times N$, where N is the number of i in D. $v \in L_S$, on the other hand, is assigned some function, called a leaf model, $g(\cdot)$, that has as its domain the part of $\mathcal{X}$, D_l or D_r, formed by the rule at η_R. The codomain of $g(\cdot)$ is the range of values of $g(y_i \mid \text{rule}_\eta)$ that are assigned to each of the parts. L_T is the set of leaf models and these are indexed by b.

By, "to partition" it is meant that, based on some rule_η, D is divided into two multisets, call them D_l and D_r such that $D_l \cap D_r = \emptyset$ conditional on the basis coordinate (covariate) x_j. "A partition" is a set of parts, a collection of multisets, where each part is called a leaf. A leaf consists only of the data assigned to it by the tree model. For example when creating a root tree model with a single node, irrespective of the size of p, the model at each b is $y = h(x_j), j \in 1, \ldots, p, b \in \{1, 2\}$.

In a similar way to a full binary tree S, a tree model T is a recursive model built up over the sample data by choosing a particular leaf node $b \in L_T$ changing it to an internal node by removing the model $g(\cdot, b)$, assigning a rule from I_T, to the new internal node and then assigning two models to the two new leaves. The analogues of the binary tree notation for the tree model are as follows:

- The elements of I_T, called the internal nodes of the tree model T, have discrete index η. $|I_T|$ is the number of internal nodes in tree T and $|I_T| = I_T$.
- The elements of L_T, have discrete index b. K_T is the size of the partition (number of elements) so that $|L_T| = K_T$.
- $\mathrm{T_R}$ is the root of the tree model,
- $\mathrm{P}(\eta)$ is the parent of an internal node and $\mathrm{P}(b)$ is the parent of a leaf node.
- $\mathrm{A}(b)$ are all the ancestor nodes of the leaf b so that $\mathrm{P_A}(b)$ is the path from the root of the tree to the leaf node b and similarly for $\mathrm{P_A}(\eta)$ and $\mathrm{A}(\eta)$
- Each $\eta \in I_T$ is associated with a neighbourhood $\mathrm{N}(\eta) = \{\mathrm{P}(\eta), \mathrm{Sib}(\eta), \mathrm{C}_l(\eta), \mathrm{C}_r(\eta)\}$.

A full binary tree (hence a tree model) has the following properties:

- Let $d_v = \lceil \log_2(v) \rceil$ indicate the depth of a node in G where $d_{\mathrm{S_R}} = 0$. Then, for any full tree:
  - $K_T \in \{2d_v + 1, \ldots 2^{d_v+1} - 1\}$ so that $\min(K_T) = 2d_v + 1$, $\max(K_T) =$

$2^{d_v+1} - 1$;

- $I_T = K_T - 1$ so that, in total, a full binary tree has $2K_T - 1$ nodes.
- The size of a tree model, $|T|$ is measured by the number of leaf nodes so that $K_T = |L_T| = |T|$.

- If the size of $|D| = N$ then N_b is the number of data points that would be located in the subset of the data D_b assigned to leaf node b by the model T. Similarly, N_l is the number of nodes in the left subtree of some internal node and N_r the number of nodes in the right subtree.

The tree models that will be the focus of this paper are regression trees hence every observation $y_i \in \mathbb{R}^n$. For sake of simple computation, each covariate space $x_j \in \mathcal{X}$, also called a coordinate basis, will be assumed to be continuous in $(0, 1)$. It should also be noted that, in general, a tree with with more than one spit at a node can be represented as a (larger) binary tree.

The section section will focus on tree models in the statistical setting. Every tree model can be described by the recursive form given above what makes them different is are ways that the rules are chosen, the method by which one chooses a particular model or set of models and the model assigned to each of the leaves of the tree.

2.3.2 Trees in Statistics

Tree modelling provides flexibility, nonlinearity, multiple dependent models, interpretability (if desired) in regression modelling (Denison, C. C. Holmes, et al. 2002). All tree models, referred to as trees, have the recursive structure in common with difference being in how rules are assigned and which leaf models are chosen. Another fact that binds all tree models is that they are strongly conditional on the data set. That is, it is possible to design a tree that perfectly describes each observation $y_i \in D$. However, this tree is practically useless from a statistical point of view as it does not generalise to a population. Each model, in the case where $N_b = 1$ has zero degrees of freedom so that a mean would not make sense, let alone a variance. Thus different approaches to tree modelling are defined, firstly, by (arbitrarily) choosing a minimum number of data points to be allocated to a leaf model. This is called a stopping rule. Secondly, to choose a method of removing leaf models so that the tree model is more general. This is called tree pruning. Finally, a method of choosing between tree models is necessary because no tree model is unique, even tree models that completely partition a data set might have more than one representation. Thus what defines

the tree modelling approach is about how one chooses rules, models and then chooses between models or sets of tree models.

2.3.2.1 Classical tree models

The Iterative Dichotomiser 3 (ID3) (J. R. Quinlan 1986), originally designed to solve classification problems, provides an iterative approach using a random window function over the training data set. Choosing appropriate thresholds is based on a combination of an information theoretic approach performing a ξ^2 test on the hypothesis that the attribute is independent of the class of objects being classified. C4.5 (J. Quinlan 1993) is an upgrade to the ID3 approach considers systems of learners.

The approach of Breiman (1969), again originally aimed at classification, is to choose covariates via "goodness of split" criterion based on an impurity measure. Initially, the tree was stopped from growing if maximum change in impurity was less than some threshold value for all possible splitting values in the data set. This was shown to be unsatisfactory because, ultimately, the more the tree was split, the better the responses were classified because the re-substitution estimate, the sum of the costs of misclassifying a response, always improves as the tree splits in Breiman (1969, p. 29). Changing the stopping rule did not help but growing a tree to its fullest capacity (every data point allocated to a single node or are "too small" (Breiman 1969, p. 58)) and then retroactively pruning and comparing trees via cross-validation using the misclassification rate of the tree proved to provide trees that better generalized to test samples.

A difference between this approach that of J. R. Quinlan (1986) is that Bayes' Rule is used to determine the misclassification rate at each node. This lead to the definition of the Bayes' rule for trees as the tree model, and its associated partition, that minimises the expected loss at each tree node so that the best tree is the one that minimises the sum of expected loss of each node (Breiman 1969, pp. 262–267). In the regression case the Bayes' rule is the mean at each leaf of the partition and risk minimizing value is the variance of the mean at each leaf. Thus the best tree over the dataset is the one where the mean value chosen by the predictors at each leaf minimises sum of the variances of the leaf nodes. Breiman (1969, pp. 317–321) provides some consistency results over empirical distributions for tree models. Suppose that the data set is large enough, that both the data and partition sets are considered random variables, the variance of the observations are finite and, importantly, that the set of partitions are linearly independent subsets of Euclidean m-dimensional space. Then in the limit the over large N, the probability that the set of predictive models $D_N(x)$ will have error greater

than ϵ is zero. Further, the sequence of tree models (predictor functions $d(x)$), $\{d_N\}$ will be risk consistent is the sense that the risk of the predictors will equal the risk of the optimal predictor defined by the Bayes' rule for the predictor.

Multivariate Additive Regression Splines (MARS) (J. H. Friedman 1991), are developed as method to overcome some of the issues of regression in tree models, specifically the discontinuity at the leaf boundaries, by using adaptive regression splines. This model uses a weighted sum of basis functions to approximate a stepwise regression selection of local models. Each local model is a positive piecewise function $h_m(x)$ and for a given $h_m(x)$, the weights or coefficients of the model are chosen by standard linear regression (Hastie, Tibshirani, and J. Friedman 2001). As this model typically over-fits it is pruned according to a generalized cross validation criterion (Hastie, Tibshirani, and J. Friedman 2001, p. 325).

2.3.2.2 Bayesian tree models

There are Bayesian versions of some of the models presented in the previous section, namely Bayesian MARS (BMARS)(Denison, Mallick, et al. 1998b), A Bayesian CART Algorithm (Denison, Mallick, et al. 1998a) (BCART) and of particular interest as already mentioned, Bayesian CART Model Search (BCARTMS) (Chipman et al. 1998).

The BMARS model adapts the MARS model by placing a prior over a parameter vector θ^k where k is the number of basis functions and θ is the parameter vector for the components of the MARS model, namely, the knots t, the predictor T, the basis coefficient functions a, the sign of the basis function s and the variance of the model σ^2. The joint probability model is $\mathsf{p}\left(k, \theta^k, y\right) = \mathsf{p}\left(k\right)\mathsf{p}\left(\theta^k \mid k\right)\mathsf{p}\left(y \mid k, \theta^k\right)$ and the aim is to find the target posterior distribution $\mathsf{p}\left(k, \theta^k\right) = \mathsf{p}\left(k \mid y\right)\mathsf{p}\left(\theta^k \mid k, y\right)$. The number of basis functions is changing the dimension of the models so search via MCMC requires the use of a reversible jump algorithm (Green 1995) . There are 3 possible model changes at each sample of the MARS model, C changes t_i, the knot position for the i^{th} basis function, B is the birth of a knew basis function and D is the death of an existing basis function. Each has associated with it a transition probability so that the acceptance ratio of the MCMC approach includes the proposal odds of the new model parameters.

The approach to BCART (Denison, Mallick, et al. 1998a) is very similar to that of the BMARS model. Here the prior probability is Poisson(λ), $k > 0$, over the number of terminal nodes k. The probability model and posterior have exactly the same form as the BMARS model but now $\theta^k = (s_i^k, s_i^{var,k}, s_i^{rule,k})$ is over s_i^k,

the number of internal nodes, $s_i^{var,k}$, the number of predictors and $s_i^{rule,k}$) the number of threshold values in the k^{th} model. In the regression case the model at each leaf is Gaussian, $N(\alpha_k, \sigma_k^2)$ and as the leaves are independent the likelihood is the product of these models where the the number of data points at each of the k leaves is a minimum of 5 and a maximum defined by the tree model and the data set.

Chipman et al. (1998) describe the approach of Breiman et al. (1984) as a "sophisticated procedure for finding good models rather than a fully Bayesian analysis,". Their approach is not too dissimilar to the approach of Denison, Mallick, et al. (1998a) in the sense that objective of the model search is to find the target posterior distribution of the tree model $\mathsf{p}(T \mid Y, X)$ via an MCMC search. To find a prior for the tree model Chipman et al. (1998) propose sampling from a tree prior process rather than sampling over the conditional distributions of the number of internal nodes, predictors and thresholds. The likelihood for this model is defined over the independent leaf models where each leaf has some function $f_b(Y \mid \theta)$ which is the same approach to leaf modelling as Breiman et al. 1984. The prior for each leaf model depends on the form of $f_b(Y \mid \theta)$ but in the regression case either the mean is considered the random parameter at each leaf and the variance is assumed known or both are assumed random. The prior is then chosen to be conjugate to the Gaussian likelihood so that the tree can be marginalised over to form the tree likelihood. The posterior for the tree model is then $\mathsf{p}(T \mid X, Y)$. Of particular interest is the generation of the prior and the model search method used by Chipman et al. (1998) and these will be further detailed.

The prior is implicitly defined via "tree generating stochastic process"(Chipman et al. 1998, p. 937) as follows: Tree size decreases as function of β and d_η, the

Algorithm 2.1: CGM Prior Algorithm

Result: A binary tree structure including covariate values and threshold values.

```
Set T to be the root node
while psplit and splitting rules available do
    Calculate: p_split = α(1 + d_η)^{-β}
    if p_split then
        Uniformly draw x_j from the available covariates
        Uniformly draw c_j from the available threshold values
        Create two child nodes;
    end
end
```

depth at node η. Note that only splits that are available in data set and are

allowable, based on the ancestral tree structure of the new split node, are to choose a new rule. Further, this model assumes that "every dataset is necessarily finite"(Chipman et al. 1998, p. 939) which is true in a very general sense but the finiteness of the dataset may mean that the multiset of data is extremely large or that the dataset will be finite at some point but not necessarily is finite now when the analysis is necessary or desired. There is also, in this model, a minimum number of data points that must be allocated to each leaf node. This can only be checked against a known number of data points.

The tree prior is a sampling process of tree models developed by Chipman et al. (1998), the algorithm for which is provided in Algorithm 2.1. This algorithm can be reformulated as

$$\mathsf{p}\,(\,T\,) = \prod_{\eta \in I_T} \mathsf{p}\,(\,\textit{SPLIT}\,)\,\mathsf{p}\,(\,\textit{RULE}\,) \prod_{b \in K_T} (1 - \mathsf{p}\,(\,\textit{SPLIT}\,)) \tag{2.46}$$

where

$$\begin{aligned} \mathsf{p}\,(\,\textit{SPLIT}\,) &= \alpha(1+\delta_\eta)^{-\beta} \\ \mathsf{p}\,(\,\textit{RULE}\,) &= \mathsf{p}\,(\,x_j\,)\,\mathsf{p}\,(\,c_j \mid x_j\,) \end{aligned} \tag{2.47}$$

and

$$\mathsf{p}\,(\,x_j\,) = \mathsf{U}(\textit{ALLOWABLE } x_j) \tag{2.48}$$

with

$$\mathsf{p}\,(\,c_j \mid x_j\,) = \mathsf{U}(\textit{ALLOWABLE } c_j \mid \textit{ALLOWABLE } x_j) \tag{2.49}$$

In the above equations $x_j \in \mathcal{X}\,, j = 1, \ldots, p$ is one of the specified covariates. In the BCARTMS model $c_j \in (min(x_j), max(x_j))$ is defined by the minimum and maximum of the covariates available in the data set and hence is a discrete distribution and invariant over the dataset unlike the model by BCART where $(min(x_j), max(x_j))$ is defined by a grid of possible values. The term *ALLOWABLE* refers to the additional qualifications that are imposed on the model. In BCARTMS these rules are:

1. x_j must have an available value at the chosen η. For example, if x_j is binary and it is already used in any $\mathrm{P}_A(\eta)$ then x_j is not allowable at η ;
2. if the pair (x_j, c_j) force the tree to have an empty node then the pair is not allowable at η. (although some other c_j with the x_j might be allowable, if such a c_j exists in the dataset.);

3. if the choice of a pair (x_j, c_j) means that there are less than 5 data points (y_i, x_i) in some terminal node b then the pair is not allowable.

The term *ALLOWABLE* is explained in Section 3.3.2.2 and refers to a configuration of nodes in some graph G that the chain is allowed to transition to when sampling the Markov Chain.

In her masters thesis, Jolicoeur-Martineau (2016)[3], mentions the dependency of the prior, not only on the data set but also on the size of the data set. This is a possible concern in the streaming setting, because over time the influence of the prior sampling process on the choice of model may wane.

Other authors such as Wu et al. (2007) and Pratola (2016) have suggested alternative ways of modifying the tree because, for example, the swap has been found to be ineffective (Pratola 2016). Further reasons include to guarding against model degeneracy, poor chain mixing and long sampling times.

The posterior of tree takes the form:

$$\mathsf{p}\,(\,T \mid X, Y\,) \propto \mathsf{p}\,(\,Y \mid X, T\,)\,\mathsf{p}\,(\,T\,) \tag{2.50}$$

where the constant of proportionality cannot be calculated directly because of the size and complexity of the tree model space. However, using a version of the Metropolis-Hastings algorithm, the tree model space can be sampled for a discrete set of models. A sequence of tree models, designed to form a Markov Chain of trees, $T^0, T^1, \ldots$ is generated by Algorithm 2.2. The grow move is to randomly select and split a uniformly chosen leaf using the rules in Algorithm 2.1. The prune alternative randomly selects an internal node and subsumes the child partitions although this only works for internal nodes that have at least two children that are leaves. The change step reassigns a splitting rule according to Algorithm 2.1 but note that it only chooses covariates from those that are available based on the data set. The swap rule swaps the splitting rules of internal parent-child pairs.

The term *ALLOWABLE* will be fully explained in Section 3.3.2 but for now it means that before a change to the tree model can be evaluated, the proposed tree must be able to accept at least 5 data points at each of the new child nodes. This means that the choice of rule, if the move is a grow move, is dependent on the data set and on the ancestral path $\mathrm{P}_A(b^*)$ between the root and $b*$ the node chosen to be changed. If the move is a swap or change move, the change in the internal

[3]This is in French and the English interpretation comes from the brief notes on her blog mentioned along with the reference.

Algorithm 2.2: Chipman et al. 1998 Stochastic Search,

Input: Initialse a root tree, T^0 with a single root node

while $\mathsf{p}(T \mid X, Y)$ *not stationary* **do**
  Change T^* from the existing tree using one of the grow, prune, swap or change moves outlined below
  if *The move is allowable* **then**
    continue
  else
    Delete proposed tree and propose another move
  Pass the data through the tree and calculate the leaf and tree marginal likelihoods
  Calculate $\mathsf{p}(T^*)$ and $\mathsf{p}(T^i)$
  Find the transition kernels $q(T^i, T^*), q(T^*, T^i)$
  Calculate:

$$\alpha(T^i, T^*) = min\left\{1\ , \frac{q(T^*, T^i)}{q(T^i, T^*)}\ \frac{p(Y \mid X, T^*)p(T^*)}{p(Y \mid X, T^i)p(T^i)}\right\} \tag{2.51}$$

  if $\alpha(T^i, T^*) < U(0, 1)$ **then**
    $T^{i+1} = T^*$
  else
    $T^{i+1} = T^i$

structure of the tree must be such that every leaf in the new subtrees of η^*, the internal node to be changed, must be able to accept 5 data points. A prune move is nearly always approved because this is the sum of data points in two leaves which is greater than 5 by design but can only be applied if both children of η^* are terminal nodes.

Thus the search for better tree models is local to the existing model, always has increasing posterior probability when accepted and is conditional on the finite set of data. CGM caution against the temptation to select as single tree as the 'best' by choosing a tree with the highest probability.

The term "while $\mathsf{p}(T \mid X, Y)$ not stationary" in Algorithm 2.2 is intentionally vague because the number of iterations necessary to achieve the required stationary distribution is unknown. Further, as Chipman et al. (1998, p. 914) point out, the tree models converge quickly to a local mode so that there are long waiting times (run times) between steps between modes of the posterior. To deal with this problem the chain is repeatedly restarted at a single node tree and then it rapidly converges again around a local model.

Alternative search methods have been proposed by Wu et al. (2007) because it has been found that the local nature of the moves required several restarts (Chipman et al. 1998) and because the convergence of the search process was slow. Pratola (2016) has also developed additional evolution steps based on the same reasoning.

2.4 Ensemble Modelling

A single tree model is limited in its usefulness because of the strong dependency between the covariate at the root node and the covariates at the nodes in the left and child subtrees (Denison, C. C. Holmes, et al. 2002). The consequence of this dependency being that every leaf model has its likelihood dominated by one covariate and a single threshold value. The interactions from other covariates at nodes in the subtrees can only describe partial effects on a single main effect at the root. Having very large trees could, perhaps, provide greater specificity for the analysis by supporting many possible models but the costs of this are the increase in complexity of the tree model, the difficulty in model parameter specification and, the more partitions there are the larger the number of data points that are required.

A collection of models provides a better approach for estimation and prediction by combining the efforts of simpler models (Hastie, Tibshirani, and J. Friedman 2001). Due to the problems described above for single tree models there are many examples of ensembles of trees and basis functions. These include J. H. Friedman (2001), Breiman (2001), Freund and Robert E Schapire (1997), Chipman et al. (2010), Taddy et al. (2011), among many others that form a collection of models to either average over, add together or vote for. There is even a form of the Kalman filter called the Evensen and Leeuwen (2000) that uses a collection of state vectors to approximate nonlinear, non-Gaussian dynamic functions.

Of particular interest in this paper are the approaches of Chipman et al. (2010), Taddy et al. (2011) and another form of combining models called finite mixture modelling (A. Gelman et al. 2013; W. Gilks et al. 1995) which is a sum of model components typically weighted by the probability of that component.

2.4.1 Particle, Additive and Mixture Models

Briefly mentioned in Section 2.2.2 was the paper by Gordon et al. (1993) which presents the bootstrap filter, a method of filtering that creates random samples of the density of interest and propagates these samples using a filter and an importance weight of each of the particles (samples). This approach has given

rise to the Sequential Monte Carlo (SMC) method which will be described more fully in Section 3.3.2. Taddy et al. (2011) uses the SMC method where each of particles are trees thereby, in a sense, creating an ensemble of trees that cover a range of possible models for each instance of the data.

Breiman et al. (1984) developed the CART model (Section 2.3.2.1) and then proceeded to develop random forests (Breiman 2001) which has been very successful in the machine learning field. In a similar way, Chipman et al. (1998) developed the Bayesian approach to random tree generation and then developed the BART model (Chipman et al. 2010) which, like "Boosting" (Freund and Robert E Schapire 1997), is an ensemble of weak learners that approximates a function by summing over many models each with a small contribution to overall model fit. BART uses an MCMC method called Bayesian Backfitting (Hastie and Tibshirani 2000) which is a form of Gibbs sampling.

Mixture modelling is a parametric technique for approximating a "true" density by summing over a weighted set of standard densities (W. Gilks et al. 1995, p. 241). These densities or components form a functional basis for the approximate model and the weights of the components are typically probabilities. The components of the mixture model do not necessarily have any particular meaning and are used to provide support to different parts of the target density. While the main objective is often to improve estimation and prediction in some cases model component inference can provide insight into the problem.

Let $f(x|\theta)$ be a density, for the sake of argument, from the exponential family of models. Let $g(x)$ the "true" density that is the target of the model and $\hat{g}(x)$ be an approximation to that density. Then the mixture model is

$$g(x) \approx \hat{g}(x) = \sum_{i=1}^{k} \pi_i f(x \mid \theta) \tag{2.52}$$

where $\sum_{i=1}^{k} \pi_i = 1$ and k is large enough (but finite) so that each component does not carry too much weight. The goal in the Bayesian setting is to estimate the posterior distribution which is proportional to prior distribution of the component weightings π_i, the model parameters θ_i and the likelihood of these parameters given the density f and the data x:

$$\begin{aligned} &\mathsf{p}\left(\pi_1, \ldots, \pi_k,\ \theta_1, \ldots, \theta_k \mid x_1, \ldots, x_n\right) \propto \\ &\qquad \mathsf{p}\left(\pi_1, \ldots, \pi_k\right) \prod_{i=1}^{k} \mathsf{p}\left(\theta_i \mid \psi_i, \lambda_i\right) \prod_{j=1}^{n} \left(\sum_{i=1}^{k} \pi_i f(x_j \mid \theta_i) \right) \end{aligned} \tag{2.53}$$

where ψ_i and λ_i are constants that occur in the exponential family of models.

The issue with this approach is that there are k^n terms that require evaluation which even modern computational hardware would be difficult to achieve for any data set of small to moderate size. Thus a typical approach is to implement a Gibbs sampler that exploits a hierarchical modelling structure that augments the data set by adding a vector of missing data, integers $z = (z_1, \ldots, z_n)$, that assign to each x_i the component in $1 : k$ that x_i belongs to. The reason for doing this is that, supposing z were known, it would be easy to assign each $f(\cdot)$ to a particular subset of components so that the likelihood and posterior are now

$$\mathsf{p}\,(\, x_1, \ldots, x_k, \mid z_1, \ldots, z_n\,) = \prod_{j:z_j=1} f(x_j \mid \theta_1) \;\ldots\; \prod_{j:z_j=k} f(x_j \mid \theta_k)$$

and the posterior is

$$\begin{aligned}\mathsf{p}\,(\, \pi_1, \ldots, \pi_k,\ \theta_1, \ldots, \theta_k \mid x_1, \ldots, x_n\,) \propto \pi_1^{\alpha_1+n_1-1} \;\ldots\; \pi_1^{\alpha_k+n_k-1} \\ \mathsf{p}\,(\, \theta_1 \mid \psi_1 + n_1\bar{x}_1, \lambda_1 + n_1\,) \;\ldots\; \mathsf{p}\,(\, \theta_k \mid \psi_k + n_k\bar{x}_k, \lambda_k + n_k\,)\end{aligned}$$

where α_i arises from the assumption of a Dirichlet prior over the model components, $n_i = \sum_{j=1} I(z_j = 1)$ and $n_i\bar{x}_i = \sum_{j:z_j=i} x_j$. This is now a product of k terms and much more manageable. In W. Gilks et al. (1995), Chapter 24 C. Robert provides a general Gibbs sampling algorithm and also some examples of binomial mixture and a normal mixture.

Also raised are interesting theoretical consequences of introducing the missing data structure to the mixed model. Sampling from the missing data structure and then the model parameters π, θ produces two Markov chains. Since the missing data structure is finite it has a finite state space and hence this allows the properties of the finite Markov chains such as geometric convergence, $\phi-$mixing and a central limit theorem to be applied from the missing data chain to the parameter chain. These issues are not discussed in this document beyond being mentioned here and where they might be applied to the proposed model but W. Gilks et al. (1995) provide references and an introduction to these concepts.

An extension of the mixture modelling approach is for hidden Markov models (HMM). As mentioned in Section 2.2.2 the Kalman filter is a form of HMM where the random distributions Y_t of the observation process Y^t are assumed independent of each other but dependent on the latent process Z^t. Each Z_i is dependent on only Z_{i-1} by the Markov assumption thus from the mixture point of view the observations x_i have density $f(x \mid \theta_{z_i})$ and belong to the hidden state Z_i with probability $\pi_{z-1,i}$, which depends on the previous state Z_{i-1}. To get the mixture distribution one must marginalise out the latent data and, according to

C. Robert (W. Gilks et al. 1995), the convergence properties of the Gibbs sampler still hold.

Next is presented two examples of ensemble tree modelling. The first is BART (Chipman et al. 2010). This is an example of an additive model in the Bayesian setting which uses Markov chain Monte Carlo. Alongside, Taddy et al. (2011) which is an example of a particle model that uses trees, BART is used as a comparative model in subsequent chapters. The Taddy et al. (2011) model has been adapted to the streaming setting (Anagnostopoulos and Gramacy 2013) whereas BART has not but it uses the Sequential Monte Carlo method rather than MCMC.

2.4.1.1 BART

Bayesian additive regression trees (BART) (Chipman et al. 2010) is a sum-of-trees model where each T is a BCARTMS model of Section 2.3.2.2. This model is described by

$$Y = \left(\sum_{j=1}^{r} g(x; T_j, M_j) \right) + \epsilon, \qquad \epsilon \sim N(0, \sigma^2) \tag{2.54}$$

where $M_j = \{\mu_{1j}, \ldots, \mu_{K_T,j}\}$ is the set of parameter values associated with each T_j. The function $g(\cdot)$ assigns to each new x_i an element of M determined by the splitting rules that form part of the parameter vector μ_k. The expectation of the response y_i given the new data x_i, $E[y_i|x_i]$, is the sum of all the assignments of x_i to μ_{kj} by $g(\cdot)$. Note that each observation is assumed to have an additive Gaussian error that is independent of $g(\cdot)$.

The aim of the prior in BART is to regularise the trees so that one tree does not dominate the prediction. The prior formulation for each tree is exactly that of Section 2.3.2.2 with the additional assumption of independence of the r trees:

$$\begin{aligned} \mathsf{p}\left((T_1, M_1), \ldots, (T_r, M_r) \right) &= \left[\prod_j \mathsf{p}\left(T_j, M_j \right) \right] \mathsf{p}\left(\sigma \right) \\ &= \left[\prod_j \mathsf{p}\left(M_j \mid T_j \right) \mathsf{p}\left(T_j \right) \right] \mathsf{p}\left(\sigma \right) \end{aligned} \tag{2.55}$$

where $\mathsf{p}\left(M_j \mid T_j \right) = \prod_k \mathsf{p}\left(\mu_{kj} \mid T_j \right)$ is a model at each of the j leaf nodes in each of the r trees. This model can be a constant model, linear model etc. Assumptions of model (leaf) and tree independence and those of the BCARTMS approach simplify the expression of the prior to $\mathsf{p}\left(T_j \right), \mathsf{p}\left(\mu_{kj} \mid T_j \right)$ and $\mathsf{p}\left(\sigma \right)$ and a few hyper-parameters, about which the authors are quite specific, the

intention being to limit the size of the trees so that each is a 'weak learner' as in the Adaboost algorithm (Freund and Robert E Schapire 1997).

The prior on the $p(\mu_{kj}|T_j)$ is assumed to be conjugate normal hence the prior on $\mathsf{E}[\,Y \mid X\,]$ is the sum of r $N(\mu_\mu, \sigma_\mu^2)$'s which implies this prior is $N(r\mu_\mu, r\sigma_\mu^2)$. The authors then continue to create a data based shrinkage of the individual trees by rescaling and shifting the data y in each leaf of each tree to achieve:

$$\mu_{kj} \sim N(0, \sigma_\mu^2) \text{ where } \sigma_\mu = 0.5/k\sqrt{r}. \tag{2.56}$$

The main data based assumption is that under the sum-of-trees approach the mean is highly likely to fall within $[y_{min}, y_{max}]$. This leads to choosing $r\mu_\mu - t\sqrt{r} = y_{min}$ and $r\mu_\mu + t\sqrt{r} = y_{max}$ where in this case t is a predetermined value, for example 2 giving a roughly 95% credible interval about the mean.

The σ prior is the conjugate prior under the Inv-$\chi^2(\nu, \lambda)$ distribution. The choice of hyperparameters here is specified such that over dispersion and over-fitting are avoided. The value of r is also specified in the BART paper and it is recommended that, at least for a starting off the model, the number of trees be 200. The paper notes that all of these hyperparameters could be learnt but that the additional computational requirements may not be worth it.

The back-fitting MCMC algorithm for searching the posterior of the forest of trees relies on noting that the conditional distribution $\mathsf{p}\left(T_j, M_j \mid T_{(j)}, M_{(j)}, \sigma, y\right)$ depends on $(T_{(j)}, M_{(j)}, y)$ through the residual:

$$R_j \equiv y - \sum_{l \neq j} g(x; T_l, M_l). \tag{2.57}$$

Thus to sample each of the (T_j, M_j) r times is the same as r draws from $(T_j, M_j) \mid R_j, \sigma$ for every $j = 1, \ldots, r$. In other words, $R_j = g(x; T_j, M_j) + \epsilon$ and, because the structure of the individual trees is the same as that in Chipman et al. (1998), we have the same posterior, $\mathsf{p}(\,T_j \mid R_j, \sigma\,)$, as in Chipman et al. (1998) for each j. Each draw for each tree then follows the Metropolis-Hastings method of Chipman et al. (1998) to get a sample of trees grown locally around the j^{th} tree. This is followed by a draw from the normal distribution for each of the tree's μ_{kj}s which is needed for subsequent draws of T_j.

After a burn-in number of draws, the sequence of $f^* = \sum_{j=1}^{r} g(\cdot; T_j^*, M_j^*)$ draws, $f_1^*, \ldots, f_M^*$, can be considered as an approximate sample from the distribution $\mathsf{p}(\,f \mid y\,)$. Thus the usual statistics, such as the mean or median of the f_m^* as well as the variance and percentiles for construction of interval estimates, can be calculated. This ensemble method also provides insight into the different marginal

effects of the x on the trees via the partial dependency of one or more predictors, x_s, against their complement x_c:

$$f(x_s) = \frac{1}{n}\sum_{i}^{n} f(x_s, x_{ic})$$

where x_{ic} is the i^{th} observation of x_c in the data.

Another interesting inferential application is variable selection involving the monitoring of the frequencies of the components of the design matrix as the number of trees r is made smaller (than the suggested 200 trees). If $v_i \equiv \frac{1}{M}\sum_{m=1}^{M} w_{ik}$ where w_{ik} is the proportion of splitting rules that use the p^{th} component of X then as r gets smaller, the repeated sampling of the trees causes the explanatory variables to compete for entry to be used as internal node rules.

In Bleich et al. (2014) information about important variables can be included for assessing the real effect of particular covariates in high-dimensional settings. In Hernández et al. (2018) high dimensional data (data sets in excess 5000 covariates) is developed for BART using Bayesian model averaging. Linero (2018) also tackles the problem of a large number of covariates but this time using a Dirichlet "hyperprior" over the proportions of the splits chosen from the covariate at each node. In a different vein, Mohammadi et al. (2020) consider reversible jump approach to BART with the intention of improving mixing of the algorithm because one weakness of tree modelling is the tendency to focus on one very successful tree so that other rival models are not allowed to develop. There are other adaptations to this popular method for Bayesian ensemble learning but none have yet provided a solution to BART in a streaming (or N independent) setting.

2.4.1.2 Trees in dynamic search

Taddy et al. (2011) propose growing trees in a sequential manner where the tree is a representation of the model state in the state-space paradigm. Each tree is also a particle in a Sequential Monte Carlo (SMC) setting. This model allows for a sensible local search for sample trees in a manner similar to Chipman et al. (1998) but also allows for a more global search of the posterior space of trees by resampling trees in manner similar to that of mixture models as described by Carvalho et al. (2010).

T_t is a tree which consists of set of partition rules at time t. A tree is evolved from $T_{t-1} \to T$ via:

$$\mathsf{p}\left(T_t \mid T_{t-1}, D_t\right) \propto \mathsf{p}\left(y_t \mid T_t, x_t\right) \mathsf{p}\left(T_t \mid T_{t-1}, x_t\right) \tag{2.58}$$

which is the posterior for each particle where D_t is all the data (x_t, y_t) up to time t. This posterior is used to calculate the posterior predictive distribution:

$$\begin{aligned} \mathsf{p}\left(y_{t+1} \mid T_t, x_{t+1}, D_t \right) &= \mathsf{p}\left(y_{t+1} \mid x_{t+1}, D_t^{\eta(x_{t+1})} \right) \\ &= \int \mathsf{p}\left(y_{t+1} \mid x_{t+1}, \theta \right) \mathsf{p}\left(\theta \mid D_t^{\eta(x_{t+1})} \right) d\theta \end{aligned} \tag{2.59}$$

where $D_t^{\eta(x_{t+1})}$ includes only the data of node $\eta(x_{t+1})$ which was selected by the new data x_{t+1}. Thus the predictive distribution for the new data x_{t+1} depends only on the leaf that is selected by x_{t+1} for each tree.

Algorithm 2.3 provides the SMC sampling approach used by Taddy et al. (2011) and in Section 3.3.2 more detail on the general SMC approach will be provided.

Algorithm 2.3: Particle Learning for Posterior Simulation.

1. $\{T_{t-1}^{(i)}\}_{i=1}^N \sim p(T_{t-1}|D_{t-1})$ is a particle approximation to the posterior where at t_0 each particle is empty.
2. Pass data to the tree until the conditions are met such that child nodes will have minimum data for calculation of sufficient statistics $S_{t,\eta}^{(i)}$ should a grow move be selected[4].
3. Once these conditions are met, to resample draw particle indices $\{\zeta(i)\}_{i=1}^N$ using the predictive probability as weights:

$$p(\zeta(i) = i) \propto p(y_t|x_t, T_t^{(i)}) = p(y_t|x_t, S_{t,\eta(x_t)}) \tag{2.60}$$

4. Set $T_{t-1}^{(i)} = T_{t-1}^{\zeta(i)}$
5. The propagation stage is to update the particles with a sample from the posterior distribution of the tree including the new data, $\mathsf{p}\left(T_t \mid T_{t-1}, D_t \right)$. This is done by partition moves similar to those of the BCARTMS model.
6. Update the estimates of the sufficient statistics $S_{t,\eta_{x_t}}$

The purpose of this method is partially based on a criticism of Chipman et al. (1998) which is that in order to explore the posterior space an improbable number of grow/prune moves would be required to radically alter the single tree structure. Having many trees (particles) allows one to pick the tree with highest posterior at time t to make a prediction. The tree at the previous time t may not be the best tree at time $t+1$ but, given the data, some tree that is best at time $t+1$ will give the most probable prediction given the new x_{t+1}. However, it can be noted from Equation (2.59) that the predictive distribution is only calculated via the parameters of the leaf selected by x_t. If each tree is a representation of the state in a dynamic state-space model then $|T| - 1$ leaves (partitions) are not selected at each time t and it may be incorrect to assume that parameters of those leaves do not also evolve as t progresses. In other words, as new information arrives,

the state should evolve as a whole not only in one partition.

Tree models are not a panacea for conditional modelling and even in the incarnation of CART (Breiman et al. 1984) limitations to the tree modelling approach are raised, for example, the limitation of single threshold values at nodes; misinterpretation of the tree if one variable is masked by another; a bias towards uneven partitioning of the data set and that growing a tree is has similar limitations to one stepwise procedures in linear regression (Breiman et al. 1984, pp. 39, 41, 42). Despite these problems and others, a founding idea behind using trees, is the operational approach that " 'honesty' is more critical than 'optimality' " (Breiman et al. 1984, p. 43) so that a tree is regarded as successful if it useful and practical under prediction.

This brings to an end the brief summary of tree modelling in Bayesian setting. While this section has not attempted to be comprehensive because this is such a large and well covered topic it has introduced the main ideas and background that supports the approach that is used in this paper. The next section will look at streaming modelling approaches with a particular focus again on regression and trees.

2.5 Streaming

Streaming data is an extremely broad term that is, in this document, going to be divided into two main threads: data streams that occur in computer science and streams of data in the statistical setting which is focused on generating (possibly) evolving statistical summaries of phenomena as recorded at source by transducers or collected in a repeated and ordered manner from a source of unknown depth.

Statistical streaming is a small sub-branch of streaming data use and analysis. The statistical streaming "field" has become more prevalent largely due to the rise in machine learning and artificial intelligence along with the associated software and hardware capabilities. However the idea of analysing streams of data, beyond the time-series approaches mentioned above (the distinction will be made more clear in Section 2.5.2) is not new and according to G. Box and Luceño (1997), statistical analysis of data and control of engineering and manufacturing processes go hand-in-hand. What has changed is that computational power has vastly increased so that analysis in "real-time" or "on-the-fly" is possible because the resulting summaries, as they are available, are possibly meaningful and in some way actionable.

In Section 2.5.2 some clarity will be provided on terminology that is used in

this document. Before that a very brief discussion of data streams in computer science will be provided. After these two brief sections the background on existing streaming regression, specifically focused on tree based methods, will be provided. The section will be rounded off with the introduction of two methods of streaming architectures which serve as infrastructure to support streaming data analytics.

2.5.1 Data Streams

In computer science a stream could be an input/output stream, a bit stream or any flow of data within the deterministic structure of a computational device such as the stream of data between monitor and graphics card or the streams of data manipulated by the scheduler. These types of stream are typically handled in manner that is appropriate to their form. The variability of data in these types of streams, "perfect communication over an imperfect, noisy communication channel"(MacKay 2002, p. 3) is handled in information theory and involves minimal loss over some communication channel that is related to the entropy of an ensemble of bits that the channel is capable of handling.

The Nyquist-Shannon theorem governs the conversion of an analogue signal (such as that produced by a real-world phenomenon) to a digital signal. Suppose that W is the highest frequency in a signal. Then the signal can be fully determined from its values at a set of discrete sample points that have distance $\Delta t = 1/(2W)$ between them (MacKay 2002, p. 178). This implies that a sufficient sample rate, a rate for maintaining the fidelity of a signal, is anything greater than $2W$ samples per second. Thus if this sample rate is achieved then, for all intents and purposes, the phenomenon that produced the signal and the recording of the signal are considered the same.

In the above sense a data stream is a digital signal, a sequence of bits (0-1 pulses) that is passed over some channel (a wire, PCB, optic cable, e.t.c.). A particular sequence of bits might represent present an identifiable object. For example a particular known sequence of exactly 512^2 bits can be used to present a black and white image where each bit represents a pixel. Thus some data, a number or image or word, is a collection of bits (a 0-1 sequence). However, for the purposes of communication between transmitter and a receiver, this signal must be formatted and processed into packets of data so exactly the right sequence can be decoded an interpreted at the receiver. These packets would typically appear in network communication which is the next form of streaming that is addressed.

To borrow from several applications, streaming is the process of data stream processing (*Amazon Kinesis - Streaming* 2021,*Confluent* 2021,*Apache Kafka* 2021).

A data stream in this sense is composed of a continuous sequence of records of events that could be created at any location, typically on an electronic device like a sensor, mobile phone, webpage, database etc. and sent over a communications network (a collection of transmitters and receivers) which could be other sensors, actuators, databases, cloud servers, mobile phones etc. A stream of data is typically processed sequentially and incrementally on a record-by-record basis in near "real-time" sometimes without storing the data. In most cases the act of stream processing entails being able to react to the information in the stream and to be able to make requests or queries of the data in the stream as the need arises. This type of stream of data is typified by the network of servers and clients known commonly as the internet.

A network is a group of nodes interconnected by links. In telecommunications this is typically a packet-switching network which acts to group data into packets that is then utilised by application software. There are many types of network including the familiar ARPANET and the current internet. All networks require a network protocol to facilitate rules, syntax, semantics and synchronisation of packets sent of the network. Typical protocols include the Internet Protocol (IP), the Transmission Control Protocol (TCP), the User Datagram Protocol, the Stream Control Transmission Protocol and the Common Industrial Protocol. The Network time protocol (NTP) is a common method for synchronising computer systems over unreliable networks (Kozierok 2005).

Not only does a network generate data about itself, for instance failures of packet deliveries, demand, latency etc. but the packets of data may need to be preprocessed (for example optimally coded or compressed), post-processed (interpreted, standardised, recoded etc) and also analysed in some ways for example on the fidelity of the received packet, timing, routing etc. Although streaming data analysis could be used in this setting much of the structure of protocols, regulation and deterministic choices control this data so that this is not the type of data that typically falls into the statistical streaming setting.

Another type of setting for streaming data is in real-time distributed computing. A real-time system is one where the behaviour of system and the outputs of the system depend on both logical and temporal correctness (Kopetz 2011, p. 2). A real-time computer system is one part of a larger whole and changes with time as does any dynamic system. A distributed system is one where computational nodes of the computer system are connected by a network. A cluster may consist of several nodes. What defines a "real-time" system from a network of servers and clients is that the "real-time" system, distributed or otherwise, must be expected to perform certain functions that will define it as a "real-time" system and that

can be used to evaluate the design and operation of the system. There is a time model for this type of system and deadlines that can be soft, firm or hard depending on the nature of the consequences if the deadline is not met (Kopetz 2011, p. 3).

These three types of streaming data setting are not exclusive but what distinguishes them, at least partially, from the statistical streaming setting is the relationship between the system and the user. In all of the above systems there is little or no need for human involvement barring design, implementation and use. They are automatic systems that do not (usually) provide for inference on the system itself (other than for fault checking) and are not aimed at collating and coordinating data that is exogenous to the system. That is, they are, for the most part, homogenous systems that support other exogenous processes (such as changes in phenomena, input from users, faults etc.) but do not permit the streams of data themselves to be selected, analysed, evaluated, changed, evolved and compared by the user. This act of choosing sources, collating them, analysing them, inferring from them and modifying them is the domain of statistical streaming which is the topic of the next section.

2.5.2 Statistical Streaming

The statistical streaming setting falls under the umbrella of "Big Data", a broad term that covers not only the analysis of data but the architecture to support the analysis, evaluation and visualization of the data and analysis and the collection and storage of the data pre and post analysis (Bifet, Gavald, et al. 2018, p. 8). Data mining is a subfield of machine learning, computer science and statistics that seeks to find patterns in large amounts of data. Alternatively called knowledge discovery or, perhaps a bit worryingly knowledge management, the idea is to provide insights into usually very large sets of data that could not be analysed by traditional means. Implicit in this setting is caution over the inference that is made based on these "discoveries" but bearing mind the sheer quantity of data currently available (Bifet, Gavald, et al. 2018, p. 7) there are clear advantages to having algorithms that can either search for patterns in these data lakes or process the data as it is being created to create summaries that not only reduce the amount of data that is stored but can provide useful insights into the data in "real-time" that can help to improve business decision making, healthcare, climate management, disaster management, computer and network security and others (Bifet, Gavald, et al. 2018, p. 9).

The proposed approach in this document falls under the use-case: analysing streams of data to calculate summaries of continuous outcomes. Hoped for ap-

plications might be in any setting where regression is usually applied with the added benefit of being independent of the size of the data set. Thus a central assumption that stems from the definition of big data (Bifet, Gavald, et al. 2018, p. 3) is that either the data set is too large to be managed with current analytical and statistical algorithms or the the data is being generated on a continuous basis so that the amount of data to be processed is not known. For all intents and purposes these two cases are the same: the number data points is big enough to be considered unknown.

Thus a key assumption in this document is that N, the amount of data to be processed or the size of the data set, is a random variable. There is no interest in learning N but its randomness implies that there is no known end to the data set so the idea of constructing tables or matrices of data fir subsequent analysis would not make sense. The fact that N is random and potentially very large means that traditional idea of a data set representing a sample of a population is challenged. Further, if N is considered countable rather than finite then the data set is not a compact multiset but, at least in the forward direction, a countably infinite set and potentially always changing.

One of the main contributions of this thesis is to attempt to apply Markov chain Monte Carlo to the streaming setting. This means challenging the necessary assumption of a bounded and unchanging set of points over which to integrate by sampling. By challenge it is meant that the necessary assumptions are retained but a method is proposed that allows for new data to be incorporated into the current MCMC chain while avoiding the need to rerun the analysis from the start that would be the usual approach if the sample data were altered. This will be discussed further in Section 3.3.2 and developed throughout the main body of the thesis.

The terms “real-time” or “on-the-fly” are often associated with streaming data analysis. There is no consensus on a definition for “real-time”. The phrase seems to indicate that it has something to do with what is “real” but from engineering to process control; computer science to economics there seems to be a difference version of “real-time” suited to each domain, to the realities involved in each of the subject areas rather than to wall-time or space-time or universe time or whatever time may actually be. It is not unreasonable then to expect statistics to be any different.

The term “on-the-fly” is more appropriate because is removes the notion that the analysis of data is happening at exactly the same time as the event are occurring. This term means that analysis is performed as and when the data are available

and when the actions that must result from the decisions need to be taken. This does not preclude the idea that one can collect a sequence of summaries of data, indeed this is almost the definition of statistical streaming, and then at some future point use this data in further analysis, discard it or take some action because of it. "On-the-fly" suggests that the data is transitory and that, to some degree, the rate of analysis is related to the transitory nature of the data. From now on the inverted commas for real-time and on-the-fly - used to suggest their impreciseness - will be dropped.

The focus of the approach of Bifet, Gavald, et al. (2018) is on big data stream mining. Their approach provides a base that supports the framework for the methods of streaming data analysis. Here are outlined some of their "axioms" of stream mining. These axioms are provided by Bifet, Gavald, et al. (2018, p. 35) which are a summary of the several passages in Gama (2010, pp. 4, 6, 7) that more or less state similar requirements:

1. Only one pass is allowed on the stream.
2. The processing time must be low.
3. Memory must be low ... certainly sub-linear in the length of the stream.
4. The algorithm must be able to provide answers at any time.
5. ... they are nonstationary data sources.

It is clear that these axioms are necessitated by the desire to have have the results of the analysis available as quickly as is reasonable and also take cognisance the potentially unlimited sample size of the data.

A classifier or regression function has the following minimal functional requirements (Bifet, Gavald, et al. 2018, p. 12):

- It must receive a set of covariates without an associated response and make a prediction based on the current model.
- It must receive a response from the past and use it to adjust the model for training.

In the usual machine learning setting training a model happens at any arbitrary time before the current data requires analysis or before predictions are made. However in the streaming setting training and prediction must be able to happen in temporal proximity to each other using the above "axioms" to achieve the minimal functional requirements. Therefore there is a simplified prediction/training cycle in streaming analysis due to the complexities of the data which are: asyn-

chronicity, buffer size limitations and the variation in data point load (too much or too little complete data). Therefore, in the Massive Online Analysis (MOA) approach the following assumptions are made (Bifet, Gavald, et al. 2018, p. 13):

- There is an observation for every labelled instance (set of covariates).
- The complete data points arrive in order so that both the covariates and response to those covariates arrive before the next set of (response, covariates) = data point arrives.

An example of a typical streaming loop is then as follows:

1. Get a set of complete covariates x_t.
2. Make prediction $\hat{y}_t = f(x_t)$ from the current model.
3. Receive observation y_t.
4. Use (y_t, x_t) to update f and $(\hat{y}_t, y_t)$ to update the statistics of the model performance.

As pointed out by the authors this approach is too simple because it ignores delayed, uncoordinated and missing data. It is advertised as appropriate for the comparison of streaming algorithmsBifet, Gavald, et al. 2018, p. 13.

The above approach to the receipt, coordination and output of data in the regression setting is the one that will be adopted here. This is for the sake of simplicity and clarity of exposition because the focus is on showing that Bayesian regression using MCMC is possible and effective in the streaming data setting.

2.5.2.1 Massive online analysis (MOA)

The reason for the focus on this particular approach is that this framework seems to be the most currently implemented framework across several streaming architectures and there does not seem to be a unified approach to Bayesian statistical streaming. It also includes the use of classical decision tree methods and has the longest historial reference to statistical streaming leading back to VFML (Domingos and Hulten 2000). It is worth noting that most of the algorithms presented in the Massive Online Analysis (MOA) approach are focused on classification rather than regression.

Ignoring resource allocation, the defining characteristic of the classical streaming approach, dealing with queries over non-stationary distributions, is the central aim of classical streaming techniques. To this end Bifet, Gavald, et al. (2018) define a changing or evolving stream as when the underlying distribution of the data in the stream changes. To evaluate an algorithm over these changing streams they

suggest considering "the 'average case' performance of an algorithm by averaging over all possible streams according to their probability under the generating distribution"(Bifet, Gavald, et al. 2018, p. 68). The Markovian assumption is made but also criticised because some streams display bursts of activity of correlated events. This is also bourne out by Xiangheng Liu and Goldsmith (2004) when considering packetloss and bursts of data. A simplifying assumption is that whatever correlations exists they are only over a short enough duration so that over a long enough data stream the Markovian assumption is valid.

The two paragraphs below present concept drift and ADWIN (Bifet, Gavald, et al. 2018, p. 79). The first is one aspect of nonstationarity others being varying variance, step changes, bursty data etc. The second is a main method of dealing with change due to model drift (when the model is no longer suitable for the data being analysed) used by many of the models presented in Sections 2.5.2.2 and 2.5.2.3.

Concept drift: Concept drift, the idea that the stream distribution changes with time, is specified as a sudden change; a gradual change; a global (over the whole stream) change or partial (part of the stream) change; recurrence (such as seasonality); regular but not periodic distortion such as the ebb and flow of traffic and real versus virtual change where the former case concerns $\mathsf{P}(Y \mid X)$ changing in response to X and the latter only $\mathsf{P}(X)$ changes (Bifet, Gavald, et al. 2018, pp. 69–70). Three change management strategies are suggested:

1. Adaptive estimators: Many model builders maintain a statistical quantity and combine the values into a single model. Naïve Bayes (a probabilistic classifier) is an example because it keeps count of values and combines them.
2. Change detectors: A change detection algorithm works in parallel to the model builder and detects (and reports) changes in the model.
3. Ensemble methods: Complex classifiers are built out of some selection of simple classifiers that may be focused on different aspects of the data stream.

Several estimators are suggested for tracking changes including the exponentially weighted moving average and the unidimensional Kalman filter (Bifet, Gavald, et al. 2018, pp. 73–74). Suggested automatic change detectors involve (hypothesis) testing the data for changes relative to some previous value under the Gaussian distribution. The cumulative sum (CUSUM) test considers a significant change in the mean of the data and the Page-Hinkley test is an extension of this idea. The input to this test could be the residual from the Kalman filter (Bifet, Gavald,

et al. 2018, p. 76). Another method of change detection under the assumption of the stationarity of the data and label distribution is the drift detection method (DDM) of Gama (2010, p. 78). Here the prediction error of the model is monitored and a change alarm is raised when prediction error increases.

ADWIN: The adaptive sliding window (ADWIN) algorithm is used in several streaming packages (Bifet, G. Holmes, et al. 2010; Montiel et al. 2018) and is used to keep track of the mean of real valued numbers. The ADWIN algorithm stores a variable length window of past data and recommends change to the length of the window if there has been a change detected over the average of the data within the window. That is, the window size is relative to the rate of change of the data based on some confidence parameter δ. An exponential histogram (Bifet, Gavald, et al. 2018, p. 57) is maintained for the window and a hypothesis test between two window lengths is carried out, in the default but not all cases, after every new data item arrives. A window is rejected (change is detected) if the average value of the window has not significantly changed.

ADWIN is not the only method for dealing with drift and change and the reader should consult Bifet, Gavald, et al. (2018, pp. 68–81) for a more thorough description. The next subsections will focus on tree methods in streaming statistics in the classical setting. A Bayesian approach to sequential tree modelling has been presented in Section 2.3.2.2 and the particular methodology, Sequential Monte Carlo, will be presented as an alternative to MCMC in Section 3.3.2.

2.5.2.2 Trees in streaming

The tree methods suggested by Bifet, Gavald, et al. (2018) are aimed mostly at classification models but some alternatives for regression are also offered. The general approach is the same as the of Breiman et al. (1984) and J. Quinlan (1993) but to estimate the splitting criteria the Hoeffding concentration inequality (Bifet, Gavald, et al. 2018, p. 38):

$$\mathsf{P}\left(|\bar{X} - \mu| > \epsilon\right) \leq 2\exp\left(-2\epsilon^2 n\right) \tag{2.61}$$
$$\bar{X} = \sum_{i=1}^{n} X_i \quad \mu = \mathsf{E}\left[X\right] \text{ and } n \text{ is the number of data pionts, } \epsilon \in (0,1)$$

is used to replace the Gini measure of Breiman et al. (1984) or the entropy measure of J. Quinlan (1993). However, to use these measures stationarity is required and it has also been argued that the Hoeffding bound is inappropriate (Bifet, Gavald, et al. 2018, p. 101). Further, in the basic formulation of the Hoeffding tree numeric attributes are not considered.

The Hoeffding tree of Hulten and Domingos (2003) uses the Hoeffding bound to generate an $\epsilon = \sqrt{\frac{R^2 \ln 1/\delta}{2n}}$. If the Gini criterion for the best covariate of a never-before-seen data point is greater than that for the second best covariate, use the first covariate as the splitting rule for a new leaf. Note that this tree only grows. It may now grow with every data point (i.e. if G(best) - G(second best) $< \epsilon$) but once a tree has been grown it does not seem to be modifiable. This approach has been modified to improve the rate of learning by creating the VFDT (Bifet, Gavald, et al. 2018, p. 104).

The CVFDT (Bifet, Gavald, et al. 2018, p. 105) is a concept adapting approach to the VFDT. The general idea is that a sliding window of data points for threshold and covariate choice is maintained as well as creating subtrees that can be used to replace existing parts of the original tree. The same criterion above is used to check attributes and the tree continues to grow if the test is passed. In addition to this process existing node instances are checked and if the attributes that would now be chosen are different to those that were chosen a subtree at that node is generated. This subtree is also monitored for some time period and if the results are better (via hypothesis testing) than the alternate branch the existing subtree is replaced with the new subtree, else it is discarded.

An alternative to the CVFDT is the Hoeffding adaptive tree (Bifet and Gavaldà 2009). Bifet, Gavald, et al. (2018) describe some default parameter values for the CVFDT as saying that only the last 50000 examples of the stream are relevant, change does not happen faster than 10000 examples and that 1000 examples are sufficient to quantify the change between subtrees for newly chosen attributes. Their criticism is that these default values are data and/or user dependent and that variation in the rate of change in the stream is not accounted for (Bifet, Gavald, et al. 2018, p. 108). Their Hoeffding adaptive tree is claimed to adapt to change based on the "scale of time change in the data, rather than relying on the a priori guesses made by the user." (Bifet, Gavald, et al. 2018, p. 109).

Bifet, Gavald, et al. (2018, p. 107) extend the Hoeffding tree with VFDTc and UFFT by using a Naïve Bayes classifier at each node, using the drift detection method of Gama (2010) to detect evolutionary changes and they provide their own approach to using numeric attributes based on storing a binary tree of all the data seen by the stream (Bifet, Gavald, et al. 2018, p. 110).

In general, Bifet, Gavald, et al. (2018, pp. 109–113) state that the handling of numeric attributes in a streaming setting is difficult. They suggest several methods of discretization of the covariate over the range of the attributes by, in general, creating empirical distributions of the data and sampling from these

distributions. An alternative method to this is using a Gaussian approximation to the empirical distribution of each covariate (for each class label) as only 3 numbers need be maintained to draw from this distribution, the minimum and maximum of the values observed and the mean.

2.5.2.3 Ensemble methods for trees in streaming

The advantage of ensemble methods in streaming is that, if the elements of the ensemble are independent of each other, the ensemble is easy to parallelise and can be decentralized This latter term means greater flexibility in modifying the members of the ensemble without having to worry about large effects in the outcomes. The combination of elements is, in general, more accurate than an individual element in its own right (Bifet, Gavald, et al. 2018, p. 129). This has been notably shown in Breiman (2001) in the non-streaming setting.

For ensemble approaches an important consideration is the "weight" or proportion of the contribution of each element to the ensemble. In its simplest form each member of the ensemble is "voted for" by choosing, in the classification case, the member whose class is the most popular among all classes (Bifet, Gavald, et al. 2018)[129-130]. Another method called the Weighted Majority Algorithm (Bifet, Gavald, et al. 2018, pp. 130–131), a version of Stacking uses experts (a binary predictor function) to make prediction $\hat{y}_t$ based on attribute vector x_t. When observed point y_t arrives, the expert compares y_t and $\hat{y}_t$ and emits 1 if correct (i.e. a match in the classification case), 0 otherwise for each member of the ensemble. A prediction from the ensemble is 1 if the sum of the weights and experts is greater than a half. A forgetting factor β is included in the model that weights each expert so that the more an expert is correct the higher the weighting of the ensemble. Normalisation of the weights ensures that weights sum up to 1 at each iteration.

Other methods for weighting learners or experts are bagging (Breiman 1996) and boosting (Freund and Robert E Schapire 1997). Bagging, in the non-streaming domain, uses bootstrapping of the training set of data to draw random samples that are applied to a set of experts, often decision trees. In the case of classification the ensemble model makes a prediction using a majority vote of the classifications produced by each of the experts. In the streaming setting, assuming sampling with replacement, the distribution of the n data points tends towards a Poisson distribution, Pois(1), so that each of the incoming sample points is weighted with this random draw and the expert is updated according to this weight (Bifet, Gavald, et al. 2018, p. 133). The basic bagging models are then updated by the ADWIN model to "replace the loser", the worst performing

expert, each time the ADWIN model detects a change in the incoming data.

Boosting is a sequential algorithm where, in the non-streaming setting, current experts are weighted according to past performance, with more weight being given to the misclassification by experts so that in the new iteration more attention is given to experts that performed poorly in the previous iteration. According to Bifet, Gavald, et al. (2018) sequential weighting via boosting is more difficult because of the sequential the chain of experts. A list of online boosting methods are provided in Bifet, Gavald, et al. (2018, p. 135).

Hoeffding option trees (HOT) models provide several options for splitting values at each node. Thus the ensemble is effectively a single tree with sets of subtrees that are created when the choice of splitting attribute at a node suggests that several attributes might be suitable. A prediction is made by a majority vote of the classifiers over all the option trees that a data point would visit in the traversal of a tree (Bifet, Gavald, et al. 2018, p. 136). A regression version of the HOT has been proposed by Ikonomovska, Gama, Ženko, et al. (2011).

According to Bifet, Gavald, et al. (2018), the success of Breiman's Random Forests has not been matched in the streaming setting (Bifet, Gavald, et al. 2018, p. 137) although it is suggested that the Adaptive Random Forest (Gomes et al. 2018) is a good contender. An approach that, on the surface, seems similar BART (Chipman et al. 2010) is the Perceptron stacking of Restricted Hoeffding trees (Bifet, Frank, et al. 2012). Here, weak learners are created by choosing small (size 1 or size 2) subsets of the attributes and using these to grow trees. The perceptron (a simple stochastic gradient descent weight (Bifet, Gavald, et al. 2018, p. 114)) is used to stack (connect) the trees so that a weighted majority of trees is used to form a prediction.

A model for regression using trees is the Fast Incremental Model Tree with Drift Detection (FIMT-DD) (Ikonomovska, Gama, and Džeroski 2011). This model is based on the Hoeffding tree with the following modifications (Bifet, Gavald, et al. 2018, p. 146):

- variance reduction of the predictor $\sum (\bar{y} - \hat{y}^2) / N$, in a similar vein to the regression approach of Breiman et al. (1984), is used to choose the covariates at internal nodes;
- numeric attributes are incorporated;
- some pruning rules are used to limit tree size and hence reduce the storage requirements of the tree;
- perceptrons (not unlike simple univariate Kalman filters) are used at the

leaf nodes;

- concept drift via the Page-Hinkley test is used at inner nodes to detect changes in the incoming stream;
- if a tree is performing badly a new tree is grown which, if the new tree outperforms the old tree in terms of minimising, for example, the mean square error, then the old tree can be replaced by the new tree.

This section has briefly summarised some of the many available methods of statistical streaming in the classical setting using trees. The next sections will consider two approaches to streaming architecture, both of which are commercially available. Note that the first approach has been selected because it is a widely used framework that other streaming service providers use and AWS has been selected as it is typical of service providers rather than special in some way.

2.5.3 Streaming architectures

Real-time distributed systems provide the foundations for streaming architectures. The systems that will be discussed in this section are a few of those that are commercially available and some that are largely academic. Only the essence of a selection of these systems will be described.

Note that none of these architectures have been used in the proposed approach. Bifet, G. Holmes, et al. (2010) is a stand-alone package for the algorithms described in Bifet, Gavald, et al. (2018). Similarly, Hulten and Domingos (2003) has its own implementation as do others. To be fully functional a streaming system must either be able to directly connect to, or select, source data or sit on top and architecture that provides this. Making this connection is beyond the scope of the proposed model because it involves greater depth in computer science than can be provided by this author.

A brief (and incomplete) list of streaming architecture approaches is:

1. Massive Online Analysis (MOA)(Bifet, G. Holmes, et al. 2010)
2. SciKit-learn(Pedregosa et al. 2011; Montiel et al. 2018)
3. TensorFlow(*TensorFlow - Probability* 2021)
4. Very Fast Machine Learning(VFML)(Hulten and Domingos 2003)
5. Noah's Ark Lab(*streamDM: Data Mining for Spark Streaming* 2021).
6. Apache (Kafka, Storm, Spark)(Sax et al. 2018; *Apache Kafka* 2021)

7. Amazon Web Services(*Amazon Kinesis - Streaming* 2021)

Note that some of these are streaming architectures and some are packages or APIs that are used by the streaming data architectures to perform calculations. These have been grouped together as together they form a streaming approach to online calculation and analysis. MOA ((Bifet, Gavald, et al. 2018)) and the work of Gama (2010) are closely related. SciKit-learn (Pedregosa et al. 2011) and SkiKit-mulitflow (Montiel et al. 2018) provide an API in python for the implementation of some of the methods provided by MOA. VFML forms the basis for the streaming approach to using trees for data analysis used in MOA via the Hoeffding Tree (Bifet, Gavald, et al. 2018, p. 102) and in its later revisions as the concept-adapting very fast decision tree (CVFDT)(Bifet, Gavald, et al. 2018, p. 105), very fast decision trees (VFDTc) and ultra fast forest of trees (UFFT)(Bifet, Gavald, et al. 2018, p. 107). Noah's Ark Lab is a subsidiary of Huiwei™ which has taken over the development of the implementations of the methods outlined in MOA called StreamDM (*streamDM: Data Mining for Spark Streaming* 2021). This is based on the architecture of Spark streaming, one of the suites provided by Kafka. TensorFlow is largely concerned with deep learning and does provide some tools for Bayesian calculation in its TensorFlow probability module (*TensorFlow - Probability* 2021). Their aim is to maximise the computational resources for large and difficult problems.

This section will provide more detail Kafka and AWS. As mentioned, this will be only be a broad outline and there are many methods for calculation and statistical streaming analysis that are not covered.

2.5.3.1 Kafka

Kafka (*Apache Kafka* 2021) defines event streaming as an "always on" system where often "the user of software is more software". The aim is to ensure that real-time and stored data are continuously delivered to the "right place, at the right time". The architecture is based on a consumer/producer model where servers and clients are parts of a distributed system that communicate over the TCP network protocol. A central focus of Kafka is fault-tolerance (if one server fails the data has been stored and is retrievable from other parts of the system architecture) and scalability for "mission-critical" use cases. The servers can be distributed across several data centres and across several cloud regions.

Event, record and message are synonymous terms and consist of

$$\langle Key, Value, Timestamp, Metadata \rangle$$

headers. Producers write events and consumers (subscribers) read and process events. These two client applications are decoupled from each other: producers never need to wait for consumers. Servers take the form of "Brokers" where some are responsible for storing data and others for continuously importing and exporting data via event streams.

Events are organised into topics: these can be considered as files within folders. Each topic has multiple producers and subscribers. Events can be accessed multiple times and Kafka's performance w.r.t to data size is "effectively constant". A topic is partitioned over a number of buckets located on different brokers. It is this distributed placement that allows for scalability and parallel access. A new event is appended to one of the topic's partitions that has that same event key. Every topic can be replicated for maintainability and fault-tolerance.

The Kafka environment is much broader than what is described here and consists of several APIs one of which is the stream API (*Apache Kafka - Streaming* 2021) which is used to perform the processing and analysing of data. There is also an admin API (*Apache Kafka - Documentation* 2021) for managing topics, brokers and other Kafka objects and as well as a set of monitoring tools (*Apache Kafka - Monitoring* 2021) that consist of classes of metrics for monitoring the nodes of a Kafka cluster.

Kafka streams This is a client library of applications (*Apache Kafka - Streaming* 2021) that can be used to process the data stored in the Kafka architecture of the preceding paragraph. It is within this streaming environment that concepts around time and timing are discussed within Kafka. It is also here that one is able to make real-time queries of the application state. Note that this is separate from monitoring the behaviour of the producers and consumers of the Kafka system (*Apache Kafka - Monitoring* 2021).

The central idea of Kafka streams is that, in practice, both streams and databases are necessary. This leads to the key concept of the duality of streams and tables (Sax et al. 2018). A stream is considered as the changelog of a table where a table is simply an array of $\langle Key, Value \rangle$ pairs. Each state change in the table is a record in the stream and if one attempts to collate a set of changes for a particular key, one reconstructs the table. Thus a table is a snapshot of the stream at a point in time. A stream is called a Kstream and a table is called

a Ktable. A Kstream is an append only function and a Ktable is an update only function. A GlobalKtable returns access to all partitions of a topic which facilitates multiple joins and message broadcasting to running instances of the application. However it has a higher storage requirement and network/server load (possibly with consequent time delays).

The topology of stream processing is an abstraction that represents an unbounded, continuously updating set of data. In Kafka, the stream is ordered, re-playable and fault-tolerant. A stream processing application then uses the graph topology of nodes (stream processors) and edges (streams) to defines its computational logic. A source is a node with no incoming streams and sink is a node with no outgoing streams. A node is a processor that receives a single input at a time and transforms this before passing it on to a subsidiary node.

The notion of time in Kafka has several different definitions that can be broadly broken down into two categories: time with respect to the outside world:

- Event time is a point in time assigned at source, often via a timestamp on the local device;
- Wall-clock time is time as one might measure it via timing device such as the time of day or that measured by a timer held outside of the computational system;

and time within the Kafka architecture:

- Ingestion time is the time point when a record is assigned to a topic partition via a broker.
- Processing time is the time when a record is being consumed by a processing application
- Stream time is a per record timestamp that marks the progress of data record through a Kafka stream. It is different from wall-clock time as it is used by time-dependent operations that are particular to the topology of the application and it refers to stream timing rather than real timing.
- New records are assigned timestamps according to certain rules that depend on the type of function at a node. For example, for aggregations of data records the timestamp assigned to the result is the maximum timestamp overall records per unique key.

The choice of using ingestion time versus event time is application specific. Thus the notion of real-time depends on the application. Using event time can relate stream, processing and wall-clock time while using ingestion time ignores the

source time and is only concerned with speed of the application and the rate of stored data retrieval.

There are two general categories of operations: stateless and stateful. The former means that the processing of an event or record is independent from the processing of all other records and some examples include:

- Filter and Inverse filter: boolean operations that choose according to a matching rule
- Branch: splits a stream into one or more streams according to a rule (predicate)
- Map: modifies a single record, value or key.
- Group: Groups records by key, a prerequisite to other stateful functions. Windowing is form of grouping but using timestamps.

There are other operations such as Peek, Merge, Groupby, CoGroup, FlatMap and a function that maps tables to streams and streams to tables. The stateful operations include:

- Aggregate: Calculates averages or sums of values of records independent of a time window.
- Join: A function that combines data records.

Stateful operations and their associated functions can be performed within a specified time window or over the duration of the streaming application. Each stateful operation requires a state store to be able store and query data required for processing.

Kafka provides several guarantees such as

- The ability to process events exactly once. For operations that cannot handle missing data or data duplicated a batch orientated framework is used in addition to stream processing.
- A consumer will always read a partition of events in the same order that they are written. Out-of-order handling exists, say, for occasions when the order of the event time is different to the order of the ingestion time. Another case is when there is no buffer allowed for waiting for multiple records from multiple topic partitions to be assembled. Out-of-order data for stateful functions can cause errors in processing logic. To deal with out-of-order data the user must trade-off latency, cost and correctness.

This is very brief and superficial view of many of the services that are offered by Kafka which also included interfaces with Apache Spark (*Apache Spark 3.1.2* 2021), a batch processing alternative to streaming bases on the MapReduce paradigm. Apache Storm (*Apache Storm* 2021) seems to be the forerunner to Kafka and exists as a lower level API for stream integration. Despite this cursory overview, the main issues of streaming are highlighted that being timing and trade-offs with respect to latency, cost and accuracy.

2.5.3.2 Amazon Web Services

Amazon Web Services also provide a suite of services not dissimilar to Kafka and in fact include Amazon managed streaming (MSK) (*Amazon MSK* 2021) which uses Kafka. It also includes Amazon Kinesis (AK) (*Amazon Kinesis - Streaming* 2021), AK Data Streams (AKDS) (*Amazon Kinesis - Developer Guide* 2017) and AK Firehose (AKF) (*Amazon Kinesis - Data Firehose* 2021). AK is the platform for the Kinesis related services and for MSK. AKDS captures and stores data and comes with guarantees such as being available within "70ms of being collected" and enduring the durability of the data, based on cost/user specification.

AKF loads streaming data into AWS based on the Extract load and transform (ELT) (Smallcombe 2021) paradigm. ELT and extract transform load (ETL) are two approaches to retrieving data from data warehouses such as AWS. The latter is the more established approach for retrieving data as it uses the fact that data is first stored in a warehouse in a particular format that allows or requires some formatting of the data before it is loaded into the user's application. ELT bypasses the warehouse storage and staging but the data are regarded as poor quality because they have not not been "cleansed" prior to the user's application. ELT is typically faster for this reason.

The structure of AKDS is similar to that of Kafka in the sense that a stream is partitioned into "shards" with optimal timing guarantees based on cost and a choice of parallelisms: Shared and Enhanced fan out where the former restricts the message propagation via bandwidth manipulation.

3 Methodology

3.1 Introduction

The previous chapter provided a review of the existing methods for solving Problem 1.1. This chapter will address Bayesian probability and modelling, the chosen methodology within which Section 1.3 will provide a solution to Problem 1.1. The Bayesian approach has an intuitive appeal in that, by definition, it updates the history of a process when new data arrives. However, to perform the calculations necessary to weigh the new data over all the evidence one must include all the historical data along with the new in these often intractable updates. A method that alleviates some of the difficulties is Markov chain Monte Carlo which is a method for calculating averages of samples from the posterior distribution so that, by the law of large numbers, the properties of posterior distribution can be understood.

This thesis will show that the Bayesian methodology is a good choice to approach regression in a data streaming setting. This chapter will support this argument by stating the philosophical position of the Bayesian paradigm and showing how this a coherent framework for the problem at hand. This brief and broad positional statement will quickly narrow to an operational means of providing quantities that capture the measurements of real-world phenomena. The rest of the first part of this chapter will focus on modelling and some of the methodological structures necessary to maintain sensible results.

The second part of this chapter will delve into stochastic processes and Markov chain Monte Carlo (MCMC). The methods described here are foundational in what follows in the rest of the document as many choices behind Section 1.3 have been informed by (and possibly inform) the MCMC method. A light touch on stochastic process theory is necessary to support the MCMC sampling approach to problem solving and this will be followed by a description of the some specific techniques used by MCMC practitioners.

The final part of this chapter will consider inference in a streaming setting. This is a can of worms that spills far out beyond the scope of this thesis because one can very easily be dragged into questions regarding temporal relevance and causality. These problems will be deftly side-stepped by accepting that there are some existing methods for inference in a streaming setting, which will be discussed, and hence it can be assumed that a new approach to modelling and inference will not unduly upset those who raise these difficult philosophical issues.

3.2 Bayesian Methodology

3.2.1 Probability is a Belief

De Finetti et al. (1974) states that "*Probability does not Exist*" and that the notion of subjective probability is the measure of the "*degree of belief* in the occurrence of an event attributed by a given person at a given instant and with a given set of information" (De Finetti et al. 1974, p. 4). Substitute the words judgement, expectation and probability for belief and one will have the essence of the Bayesian approach to probability: all assessments of probability are subjective and dependent on the observer, analyst or whoever is asked to make some bet (take a defined position) on the outcome of some phenomenon often in an experiment but commonly an everyday event.

The consequence of accepting that subjectivity is the basis for existence of probability is that its inherently biased nature becomes irrelevant: "It is purely a question of studying [the evaluation of the probability] and saying whether it is coherent or not."(De Finetti et al. 1974, p. 8). The reasons for this are that, for a subjectivist (De Finetti et al. 1974, p. 7):

- No two events are ever identical or else one would not be able to the difference between them. Conversely, if one cannot tell the difference between two events then one is indifferent to these events and hence, must have the same probability.
- A subjectivist is aware that it is not possible to completely account for all factors affecting events (occurrences) but if, as far as She is aware, these events do not affect each other then She may choose to consider these events independently.
- A subjectivist is also willing to consider her position in the role of analyst as well as the the opinions and biases of others when performing an analysis. She incorporates this prior knowledge, fully expecting the data to modify

these initial conditions.

Thus probability centres on coherent statements about uncertain events wherein one must have some stake. Initially there may be bias in the assessments of the likely outcome(s) but this bias will ultimately be irrelevant as more evidence is obtained to support the data.

A parallel concept is that of chance: a property of the world or of objects (N. Singpurwalla 2006, p. 50) or as the hypothesised statistical long-run behaviour (Lindley 1991, p. 17) of some events. Thus chance is a property that may be considered objectively if it were possible to control the experiment (clearly not everyday events) for its entirety, study these results and show that such a property exists.

Hacking and Romeijn (2016, p. 196) raises the point, in criticism of De Finetti, that it is difficult, when flipping a coin to consider that the frequency of heads is anything other than either the property of the coin or of the flipping device. But as David Lindley points out (Hacking and Romeijn (2016, p. 196)), there is a difference between a statement about a one off event and a statement about a phenomenon that has been known to occur several times already. That is, it is only after some coin, presumably the exact coin being flipped this time, has been flipped many times and the outcome of those flips faithfully shared that one can consider that frequency of heads is a property of the coin and not before one has flipped the coin.

The debate about the correct definition of probability extends beyond these two viewpoints and continues unabated. In this document the subjective Bayesian definition has been adopted largely because within this Bayesian framework one can begin with very little information and support inference by the accumulation of data and hence, should the driving mechanism of the phenomenon under study change, one can then proceed to a new position of inference in a natural and coherent way.

3.2.2 Coherence

A subjective probability is the "extra-logical, subjective and personal" (De Finetti et al. 1974, p. 72) attribute that all stakeholders involved in a particular pursuit attribute to a particular event that may or may not occur as a result of the activity being pursued.

If a stakeholder is certain of the outcome of an event then they must assign the value 1 to that outcome and if they are certain that the same outcome is

impossible then they must assign the value 0 to that event. Between these two poles of certainty and impossibility it is necessary to provide weights or gradations to the possible outcomes of the event or events about which you are uncertain (De Finetti et al. 1974, p. 25). It is assumed that there is at least one outcome that corresponds to exactly one event and the lack of that minimal event and hence its outcome means that not event has taken place.

If there are several events, each with distinct outcomes from distinct events and one is indifferent to the exchange of these events then the total outcome is the union of these events. That is, if a stakeholder is indifferent to permutations of the order of events and these events do not interact with each other then one can sum these events to achieve the outcome of interest.

The previous paragraph begs the question: what if the events do interact? This implies that for at least two events there is a third event, the event of the interaction between these two events called the intersection of the events (Lindley 1991). At the core of the Bayesian approach to probability is the assumption that all events interact to some degree with the history of the activity of interest and even prior to this activity being started. Call this event the History or Hypothesis and note that "everything is conditional upon '*hypothesis*' H" (De Finetti et al. 1974, p. 135).

These three paragraphs simply described the three laws of probability in an intuitive manner. More formally, the first paragraph is known as the convexity law:

$$0 \leq \mathsf{p}\,(\,E \mid H\,) \leq 1 \tag{3.1}$$

where H is now explicitly included but after this section will be implicitly assumed. The second paragraph introduces the addition law:

$$\mathsf{p}\,(\,E_1 \text{ or } E_2 \mid H\,) = \mathsf{p}\,(\,E_1 \mid H\,) + \mathsf{p}\,(\,E_2 \mid H\,) \tag{3.2}$$

were E_i denotes some event with an unambiguous outcome. Extending Equation (3.2) to a finite number of events is called finite additivity and seems reasonable. Extending Equation (3.2) to an infinitely countable number of events is not without controversy (N. Singpurwalla 2006, p. 12). The final law is call the multiplication law:

$$\mathsf{p}\,(\,E_1 \text{ and } E_2 \mid H\,) = \mathsf{p}\,(\,E_1 \mid H\,)\,\mathsf{p}\,(\,E_2 \mid E_1, H\,) \tag{3.3}$$

which gives the probability of the aforementioned third event, that both events

occur simultaneously. If there is no relationship between E_1 and E_2 (these events come from exclusive sets with no shared history) then the events are called independent and Equation (3.3) can be written as:

$$\mathsf{p}\,(\,E_1 \text{ and } E_2 \mid H\,) = \mathsf{p}\,(\,E_1\,)\,\mathsf{p}\,(\,E_2\,) \tag{3.4}$$

There are two useful theorems that result from these laws, the first is called the the Theorem of Total Probability or the Extension of the Conversation and the second is Bayes' theorem. H is omitted in what follows for clarity.

Suppose that E_1 are mutually exclusive and exhaustive (do not interact and fully describe all possible outcomes of events) then some event A can, by the Theorem of the Extension of the Conversation be calculated by considering:

$$\mathsf{p}\,(\,A\,) = \mathsf{p}\,(\,A \mid E_1\,)\,\mathsf{p}\,(\,E_1\,) + \mathsf{p}\,(\,A \mid E_2\,)\,\mathsf{p}\,(\,E_2\,)\,. \tag{3.5}$$

Thus, if A is some hard to describe or calculate problem it is possible to extend the problem to include additional events E_1 and E_2 that may be easier to describe and hence calculate.

Suppose that there are two events that interact, call them event E and event F. Then by Equation (3.3), because resulting probability does not depend on the order of E and F,

$$\mathsf{p}\,(\,E\,)\,\mathsf{p}\,(\,F \mid E\,) \text{ and } \mathsf{p}\,(\,F\,)\,\mathsf{p}\,(\,E \mid F\,)$$

so that

$$\mathsf{p}\,(\,E\,)\,\mathsf{p}\,(\,F \mid E\,) = \mathsf{p}\,(\,F\,)\,\mathsf{p}\,(\,E \mid F\,)$$

Thus, provided that the event E has some non-zero probability of occurring, if one wants to learn about the outcome of $F \mid E$ (and H) and have occurred it is enough to know

$$\mathsf{p}\,(\,F \mid E\,) = \frac{\mathsf{p}\,(\,F\,)\,\mathsf{p}\,(\,E \mid F\,)}{\mathsf{p}\,(\,E\,)}. \tag{3.6}$$

As Lindley (1991, pp. 43–44) points out, $\mathsf{p}\,(\,F \mid E, H\,)$ and $\mathsf{p}\,(\,E \mid F, H\,)$ are two different probabilities concerning the same two events where in the former E is part of H and in the latter E is the random variable and F is part of H. It is often that case that some event is easier to formulate than some other event so setting up an experiment in terms of the easier to formulate events is necessary

to learn about the (often) intractable event. However, this equally often requires making a guess, judgement or statement of belief about $\mathsf{p}(F)$ which is where the controversy around Bayesian statistics originates.

The next section will present the Bayesian approach to statistics and modelling based on Equation (3.6).

3.2.3 Modelling

Probability Model

The Bayesian approach to modelling requires the specification of a marginal joint probability distribution or measure

$$\mathsf{P}(Y_1, \ldots, Y_t) \tag{3.7}$$

over the events of interest $Y_1, \ldots, Y_t$ called the observations or outputs. As this paper is focused on regression it is assumed that random observations are real numbers or vectors of dimension n: $Y_t \in \mathbb{R}^n$. It is also assumed that all distributions are representable as either Lebesgue or Lebesgue-Stieltjes integrals and that the densities for these distributions exist via the Radon-Nikodym derivative so that

$$\mathsf{P}(Y_1, \ldots, Y_t) = \int \mathsf{p}(Y_1, \ldots, Y_t)\, dY^t$$

where $dY^t = dY_1, \ldots, dY_t$ denotes a differential with a joint measure that is random over all random variables from $1, \ldots, t$ (more accurately written $\int d\,\mathsf{P}(Y_1, \ldots, Y_t)$ but this notation is not used in the rest of the paper). t is an indicator set, in this paper $t \in \mathbb{Z}^+$, but in general t could be any ordered set up to $\mathbb{R}$. The notation Y^t means the whole set of random variables Y_t up to index t, which in this document is always increasing in the forward direction.

The motive for the subjective approach to modelling is "learning from experience" (Bernardo and A. Smith 2007, p. 166). This is encapsulated in Equation (3.3) of the previous section where it is implicitly implied that the joint specification of the probability model (Equation (3.7)) can be partitioned in any way necessary to support the modelling of some subset of the observations and that some future possible observations, $Y_{t+1}, \ldots, Y_s$ conditional on the existing observations can be specified as:

$$\mathsf{p}(Y_{t+1}, \ldots, Y_s \mid Y_1 = y_1, \ldots, Y_t = y_t) = \frac{\mathsf{p}(Y_{t+1}, \ldots, Y_s)}{\mathsf{p}(Y_1 = y_1, \ldots, Y_t = y_t)} \tag{3.8}$$

However it is extremely difficult, if not impossible, to fully specify the marginal and conditional judgements explicitly (Bernardo and A. Smith 2007, p. 167; Goldstein and Wooff 2007, p. 5) so it is necessary to resort to some specific forms of model that the stakeholders in the problem believe will accurately make specification of the model easier.

The first approach towards simplification is to choose which aspects of the model are dependent on other aspects. The Bayesian approach to this is to specify which of the observations are exchangeable with other observations hence removing a dependence on order or rank. A further step towards simplification is to attest to some kind of geometric or invariant relationship between observations to give a particular symmetry to the model. Yet another way to reduce the complexity of the model is to state that a summary of some of the data is sufficient to describe the dependency between the data, particularly between the past and the future. Each of these aspects of simplification are used in this paper and will be briefly summarised in the following subsections.

3.2.3.1 Exchangeability

The assumption of exchangeability is similar to assuming that observations of the random variables are independent which would mean that

$$\mathsf{p}\left(Y_{t+1}, \ldots, Y_s \mid Y_1 = y_1, \ldots, Y_t = y_t\right) = \mathsf{p}\left(Y_{t+1}, \ldots, Y_s\right) = \mathsf{p}\left(Y_{t+1}\right)\mathsf{p}\left(Y_s\right)$$

by the Equation (3.4). However this means that there is no "learning from experience" (Bernardo and A. Smith 2007, p. 168). In the Bayesian setting, conditionality on the history of the observed process which may include known explanatory variables, other random variables or missing data, also called parameters, is the main thrust of the methodology and exchangeability of observations, which includes this history, is essential.

To include the effect of past behaviour in the current model, it is required to specify a dependency structure between individuals, and subsets of individuals, of the observables. This is the intention of exchangeability. That is, exchangeability is the belief that all observations possess a symmetry with each other so that, from the modellers point of view, the order of the distribution of events does not matter (Bernardo and A. Smith 2007, p. 168). More formally:

Definition 3.1. The random quantities Y_1, Y_2, ... Y_t are considered finitely exchangeable with respect to the measure $\mathsf{P}\left(Y_1, Y_2, \ldots Y_t\right)$ if the joint distribution of these random quantities is the same as the joint distribution of any permutation

of the same random quantities:

$$\mathsf{P}\,(\,Y_1,\,Y_2,\,\ldots\,Y_t\,) = \mathsf{P}\left(\,Y_{\pi(1)},\,Y_{\pi(2)},\,\ldots\,Y_{\pi(t)}\,\right)$$

where $\pi(i)$ represents a permutation of the index labels.

Infinite exchangeability is the extension of finite exchangeability to sequences of finitely exchangeable random quantities. Partial exchangeability is an extension of exchangeability where the modeller considers that there are additional "labels"on the random quantities that designate groups of data that may exchangeable while allowing the modeller to specify a judgement about between group dependency (Bernardo and A. Smith 2007, p. 170). This will be dealt with in a separate subsection below because partial exchangeability is essential for expanding models to be able to use all available information to strengthen the learning process (Bernardo and A. Smith 2007, p. 211).

From the some belief about the exchangeability of a sequence of random variables De Finetti was able to prove his famous representation theorem. The idea is that if one makes a judgement that, in some way, the sequence of observables is exchangeable then the joint distribution of the sequence will present itself in the form of an expectation of the observables with respect the prior (or belief (Bernardo and A. Smith 2007, p. 179) distribution. More formally:

Proposition 3.1. ***General representation theorem.***
If $Y_1,\,Y_2,\,\ldots$ is an infinitely exchangeable sequence of $0-1$ random quantities with probability measure $\mathsf{p}\,(\,Y_1,\,Y_2,\,\ldots\,)$, there exists a probability measure $\mathsf{Q}(\cdot)$ over $\mathcal{F}$, $\mathsf{Q}(F)$, the space of all distribution functions on $\mathbb{R}$, such that the joint distribution of $Y_1,\,\ldots,\,Y_t$ has the form

$$\mathsf{P}\,(\,Y_1,\,\ldots,\,Y_t\,) = \int_{\mathcal{F}\in\mathcal{F}} \prod_{i=1}^{t} F(Y_i) d\mathsf{Q}(F)$$

where

$$\mathsf{Q}(F) = \lim_{t\to\infty} \mathsf{P}(F_t),$$

and F_t is the empirical distribution function defined by $Y_1,\,\ldots,\,Y_t$

Thus there is a representation over all possible, measurable distributions, judged exchangeable, where the empirical distribution function F_t acts as a random (unknown) measure of the actions involved in distributing the quantities of interest The representation theorems (finite and and general) provide the theoretical support and clarification needed that link the familiar ideas of statistical modelling

to the subjective viewpoint (Bernardo and A. Smith 2007, p. 179).

3.2.3.2 Invariance

A further assumption of invariance places some additional restrictions on the nature of the space of the observations. For example, suppose that the sequence of observables has been considered to be exchangeable. Then, suppose the geometry of the space within which all of these observables, and subsets of these observables, is also considered symmetric in all directions, that is, spherical. i.e. $\mathbf{Y} = A\mathbf{Y}_i$ for all orthogonal matrices $A^T A = I$. If the predictive model $\mathsf{P}(\cdot)$ preserves this distribution then Bernardo and A. Smith (2007, p. 182) show, using the representation theorem, that the normal model of the distribution of observations is a consequence of the assumption of rotational symmetry.

Other forms of invariance give rise to some other well known probability models. For example, the additional assumption (after exchangeability) that any two beliefs that occur in the positive quadrant are symmetrical about the 45° line gives rise the exponential model. Further, it can be shown (Bernardo and A. Smith 2007, p. 188) that if the line is invariant to the origin then the "memorylessness" property of the exponential model can be explained. Similarly for the only other model that displays the "memorylessness" property: the geometric model.

3.2.3.3 Sufficiency

Sufficiency is the judgement or belief that a summary of the observables is sufficient to model the situation at hand. This summary is a number that captures a feature of the sequence of observables upon which it is possible or perhaps necessary to rest the model. This number is called a statistic (Bernardo and A. Smith 2007, p. 190):

Definition 3.2. Given a sequence of random quantities or vectors $Y_1 \ldots Y_t$ with specified sets of possible values $y_1 \ldots y_t$ then a random vector $\mathbf{s}_t : y_1 \times \ldots \times y_t \to \mathbb{R}^{k(t)}$ where $k(t) \leq t$ is called a $k(t)$-dimensional statistic.

Familiar examples of this type of summary are:

- The sample mean: $\mathbf{s}_t = t^{-1}(y_1 + \ldots + y_t),\ k(t) = 1$
- The sample size, total and sum of squares: $\mathbf{s}_t = (t, (y_1 + \ldots + y_t), y_1^2 + \ldots + y_t^2)$ where $k(t) = 3$

Typically $k(t) < t$ so that the summary produces a reduction in in t. The reason for these summaries is to make it easier to achieve the evolution of beliefs. That is, based on the chosen summary one hopes to make predictive state-

ments about new data $Y_{t+1} + \ldots + Y_k$ described by the probability statement $\mathsf{p}\,(\,Y_{t+1} + \ldots\, Y_k \mid Y_1 + \ldots\, Y_t\,)$. To do this one requires that a statistic is has predictive sufficiency (Bernardo and A. Smith 2007, p. 191):

Definition 3.3. Given a sequence of random quantities or vectors $Y_1 \ldots Y_t$ with probability measure $\mathsf{P}\,(\,\cdot\,)$ and density $\mathsf{p}\,(\,\cdot\,)$ the sequence of statistics $\mathbf{s}_1$, $\mathbf{s}_2$, ..., defined over the same set as the random variables $Y_1 \ldots Y_t$, is said to be predictively sufficient for the sequence $Y_1\, Y_2, \ldots$ if for all $s \geq 1$, $r \geq 1$ and $\{i_1, \ldots, i_r\} \cap \{1, \ldots, t\} = \emptyset$:

$$\mathsf{p}\,(\,Y_{i_1}, \ldots, Y_{i_r} \mid Y_1, \ldots, Y_t\,) = \mathsf{p}\,(\,Y_{i_1}, \ldots, Y_{i_r} \mid \mathbf{s}_t\,) \tag{3.9}$$

An example of this kind of summary used extensively in this document is known as the Markov assumption. In that case, the statistic is $Y_{t-1} = y_{t-1}$ which means that all relevant information about the stochastic process is summarised in the data up to the previous value of the current observation. Another way to interpret this kind of summary is that past observations are conditionally independent of future observations given the current observation. Thus the assumption of summary statistics makes is easier to make predictions about future behaviour based on past behaviour.

Now suppose that knowing the current random value $Y_t = y_t$ means that it possible to assume that the past data is also exchangeable and they can be summarised by some parameter θ. Then it is possible to show via the representation theorem (Bernardo and A. Smith 2007, p. 192) that learning about the unknown parameter θ is sufficient to transmit the new information to the posterior distribution. Thus an alternative definition of a statistic is a parameter that is sufficient to perform this task.

Bernardo and A. Smith (2007, p. 193) proceeds to show that predictive and parametric sufficiency are equivalent and that a sequence of statistics is (parametrically) sufficient if and only if it is conditionally independent of some other statistic, say θ. Further, it is clear from Definition 3.2 that $s_t(Y_t, \ldots, Y_t) = (t, Y_1, \ldots, Y_t)$ is always a sufficient statistic. Therefore to achieve summaries of data a minimally sufficient statistic is defined as a sequence of statistics $s_1, s_2, \ldots$ such that for any other sequence of statistics, $u_1, u_2, \ldots$ these can be represented by some function(s) such that $s_i = g_i(u_i), i = 1, 2, \ldots$.

Partial Exchangeability

The idea of partial exchangeability is to be able to extend Bayesian probability modelling to several random variables, possibly with different indices that may themselves be either random variables or statistics. Bernardo and A. Smith (2007, pp. 211–215) show several possible forms of partial exchangeability including the exchangeability between sequences of exchangeable random variables, exchangeability of sequences of random variables conditional on sufficient statistics, structured exchangeability (for example, K replicates, in I groups with J treatments), exchangeability due to the knowledge imparted by some known (or random) covariates and exchangeability due to a structured approach of the prior specification of random quantities, Q that occurs in hierarchical modelling.

Of particular importance in this paper is modelling due to some missing or latent data, that is parameters which are sufficient statistics by Section 3.2.3.3, modelling with covariates of which regression is an example and hierarchical modelling because this allows one to learn and adjust parameters or hyperparameters.

In the first case partial exchangeability means that there is unrestricted exchangeability for sequences of predictive sufficient statistics so that one can write a representation of the probability model as

$$\mathsf{p}\left(\, Y_1(k_1), \ldots, Y_t(k_t) \,\right) = \int_{\theta \in \Theta^*} \prod_{i=1}^{k} \prod_{j=1}^{t_k} \mathsf{p}\left(\, Y_{i,j} \mid \theta_i \,\right) d\mathsf{Q}(\theta_1, \ldots, \theta_k) \tag{3.10}$$

where $\Theta^* = \prod_{i=1} \Theta_i$. Suppose then that the parameters of interest were $Z_b, b \in \{1, \ldots, K_T\}; K_t \in \mathbb{N}$ and that K_T was also a random variable. Unrestricted exchangeability means that this relationship between Z_b and K_T is representable in a Bayesian probability model.

Suppose there are some known (specified and observed) variables $x(t) = (x_1, \ldots, x_p)$. Bernardo and A. Smith (2007) show that the model $\mathsf{p}\left(\, Y_1(p_1), \ldots, Y_t(p_t) \mid x_1, \ldots, x_p \,\right)$ has representation

$$\mathsf{p}\left(\, Y_1(p_1), \ldots, Y_t(p_t) \,\right) = \int_{\theta \in \Theta^*} \prod_{i=1}^{p} \prod_{j=1}^{t_p} \mathsf{N}(Y_{ij} \mid \mu_i(x_i), \sigma_i(x_i)) \mathsf{Q}(\theta(x))$$

where

$$\mu_i(x_i) = \lim_{t \to \infty} Y_t(i), \qquad \sigma_i^{-1}(x_i) = \lim_{t \to \infty} s_t^2(i), \; s^2 \text{ is the standard error of } Y^t \text{ and}$$
$$\theta(x) = (\mu_1(x_1), \ldots, \mu_t(x_p), \, \sigma_1(x_1), \ldots, \sigma_t(x_p)). \tag{3.11}$$

Thus modelling with covariates is simply a case of assuming that the random

quantities or vectors are conditionally independent (that is, the same as Equation (3.8)) given the known covariates and assigning a prior distribution to the parameters θ.

Hierarchical modelling involves additional beliefs about the prior distribution $\mathsf{Q}(\theta_1, \ldots, \theta_k)$ when it is believed that additional structure to this prior may lead to interesting and hence informative representations between the sequences of data that were represented in Equation (3.10). The general idea is that the prior specification takes the form

$$g(\theta_1, \ldots, \theta_k) = \int_{\phi \in \Phi} g(\theta_1, \ldots, \theta_k \mid \phi) d\Pi(\phi) = \int_{\phi \in \Phi} \prod_{i=1}^{k} g(\theta_i \mid \phi) d\Pi(\phi). \tag{3.12}$$

where $g(\cdot) \in G$ is a function that is distributed according the parametric distribution G.

Assuming that sufficient statistics $s_i(t_i), i = 1, \ldots, k$ exist then Equation (3.12) defines the hierarchical structure

$$\mathsf{p}\left(s_i(t_i), \ldots, s_k(t_k) \mid \theta_1, \ldots, \theta_k \right) = \prod_{i=1}^{k} \mathsf{p}\left(s_i(t_i) \mid \theta_i \right) \tag{3.13}$$

$$g(\theta_1, \ldots, \theta_k \mid \phi) = \prod_{i=1}^{k} g(\theta_i \mid \phi) \tag{3.14}$$

$$\Pi(\phi) \tag{3.15}$$

where Equation (3.13) relates data to parameters, Equation (3.14) models the parameters θ as if they were a random sample from some parametric distribution G indexed by the hyperparameters ϕ and the third stage, Equation (3.15) specifies beliefs about the hyperparameters ϕ.

The next section briefly discusses inference in the Bayesian setting.

Bayesian Inference

The development of the methodology began with explaining the philosophical and theoretical basis behind Bayesian probability. Using this theoretical basis, Section 3.2.3 showed how one can represent a set of beliefs about quantities so that a measure of these observed phenomena, in a structured form called a probability model, can be provided. This section will show how one can make inference about these observed, but uncertain quantities for the purposes of making decisions and taking actions.

Assuming the model prescribed has a parametric representation, Bayesian infer-

ence concerns the manner in which one updates one's beliefs about parameters or missing data in the light of evidence (and new evidence) in the form of data. The theorem for doing this is Bayes' theorem or rule, Equation (3.6).

Bayes' rule, as means for learning about parameters, is a step in the predictive process (Bernardo and A. Smith 2007, p. 243) in passing from

$$\mathsf{p}\,(\,y_1,\ldots,y_n\,)=\int \mathsf{p}\,(\,y_1,\ldots,y_n\mid\theta\,)\,\mathsf{p}\,(\,\theta\,)\,d\theta$$

to

$$\mathsf{p}\,(\,y_{n+1},\ldots,y_m\mid y_1,\ldots,y_n\,)=\int \mathsf{p}\,(\,y_{n+1},\ldots,y_m\,)\,\mathsf{p}\,(\,\theta\mid y_1,\ldots,y_n\,)\,d\theta$$

through

$$\mathsf{p}\,(\,\theta\mid y_1,\ldots,y_n\,)=\frac{\mathsf{p}\,(\,y_1,\ldots,y_n\mid\theta\,)\,\mathsf{p}\,(\,\theta\,)}{\int \mathsf{p}\,(\,y_1,\ldots,y_n\mid\theta\,)\,\mathsf{p}\,(\,\theta\,)\,d\theta}. \tag{3.16}$$

The equation in 3.16 is made of:

$\mathsf{p}\,(\,\theta\,)$ the prior density for one's beliefs about θ; (3.17)

$\mathsf{p}\,(\,y_1,\ldots,y_n\mid\theta\,)$ the likelihood function; (3.18)

$\int \mathsf{p}\,(\,y_1,\ldots,y_n\mid\theta\,)\,\mathsf{p}\,(\,\theta\,)\,d\theta$ the marginal of the data and (3.19)

$\mathsf{p}\,(\,\theta\mid y_1,\ldots,y_n\,)$, the posterior distribution. (3.20)

is called the evidence, constant of proportionality or the

Using the Representation Theorems of Subsection 3.2.3.1 it was shown that beliefs about parameters corresponded to long properties of observables and what these might be in a future sample. As Bernardo and Smith put it "Inference about parameters in thus seen to be a limiting form of predictive inference about observables" (Bernardo and A. Smith 2007, p. 244). Parametric inference is only part of the process and, as Andrew Gelman and Shalizi (2013) put it, should not be seen as an end itself. However, inference about parameters is still important, especially in the dynamic model of Subsection 2.2.2 where it is vital to transfer information from the past into the future.

It is possible to marginalise out nuisance parameters λ from parameter vector $\theta = (\phi,\, \lambda)$ to focus interest on parameters of interest ϕ (Bernardo and A. Smith

2007, p. 245). This provides the posterior

$$\mathsf{p}(\phi \mid y_1, \ldots, y_n) = \int \mathsf{p}(\theta \mid y_1, \ldots, y_n)\, d\lambda = \int \mathsf{p}(\phi, \lambda \mid y_1, \ldots, y_n)\, d\lambda \text{ where}$$
$$\mathsf{p}(y_1, \ldots, y_n) = \int \mathsf{p}(y_1, \ldots, y_n \mid \theta)\, d\theta = \int \mathsf{p}(y_1, \ldots, y_n \mid \phi, \lambda)\, d\phi d\lambda \qquad (3.21)$$

A not uncommon approach in inference is to introduce latent parameters as a form of nuisance parameter so that one can obtain marginals over observations of complicated objects by then integrating out the nuisance or latent parameter. An example of this is using a continuous variable to perform regression on a discrete data model, as explained in A. Gelman et al. (2013, p. 408), by using a latent variable to express the the distribution of the change between 0 and 1. This is equivalent to a Probit model.

An important consequence of parametric sufficiency discussed in Subsection 3.2.3.3 is that $\mathbf{t}(x)$ is therefore sufficient in inference to transfer in formation between the existing and new state based on new or incoming data. Sufficient statistics can themselves be partitioned in to two exclusive subsets where, given the an ancillary statistic $\mathbf{a}(x)$, the statistic $\mathbf{s}(x) \mid \mathbf{a}(x)$, θ is sufficient for use as the record of information in the likelihood (Bernardo and A. Smith 2007, p. 248).

Bernardo and A. Smith (2007, p. 249) point out that the Likelihood Principle (Birnbaum 1962) (that proportional likelihood functions should produce identical inference), is a natural consequence of Bayes' theorem rather than being an imposed principle.

3.3 Bayesian Methods

3.3.1 Stochastic Processes

A stochastic process $\{Y(t)\}_{i=0}^{t}$ is a collection of random variables ordered by an index set t where both $Y(t)$ and t can be either discrete or continuous. t is usually one of the sets $\mathbb{Z}$, $\mathbb{Z}^+$, $\mathbb{R}$ or $\mathbb{R}^+$. The state space S of the process are values $s \in S$ that the random variables $Y(t)$ map to from the sample space Ω, indexed by the set t so that $Y(t; \omega) = s$. It is through observations of the sample paths, trajectories or realisations that one attempts to infer the properties that underlie the generation of the process.

The definition of a stochastic process used in this paper is:

Definition 3.4. *A **stochastic process** is a collection of random variables* $\{Y(t;\, \omega) : t \in \{0, \ldots, t\};\, \omega \in \Omega\}$ *where* Ω *is the sample space and* t *is the index set that*

describes the sample ordering. The observed processes are the sequences (path, realizations, trajectories) of the realisations $s \in S$ that the random variables $Y(t)$ have taken up to time t.

The difference between a discrete-time process and continuous-time process is that in the former case the observations occur a regular, deterministic intervals and the in the latter case the timing of the occurrence of an observation is at random index t so that the intervals or waiting times between observations are of random duration (length).

Definition 3.5. *A **discrete-time process** is a collection of the random variables $\{Y_t : t = -\infty, \ldots, \infty\}$ where the index set t is a countable set. The state space S is then mapped to a set of realizations or observations, typically denoted by i and j but these may be elements of $\mathbb{R}^n$ or $\mathbb{Z}^n$. A (fully) discrete process is where both the index and state spaces are discrete where the latter is often also assumed to be finite.*

Discrete-time processes are ubiquitous in the study of random processes. They are found wherever the approximation of a waiting time can be considered to be regular. Especially useful is the study of Markov chains in the discrete space because the transitions from one state to another can be formulated in matrix notation and this lends itself well to linear analysis. For the most part this paper will consider discrete-time processes over state spaces $\mathbb{R}^n$ and $\mathbb{Z}$ however is worth mentioning that the Kalman filter is a discretisation invariant (Särkkä and Solin 2019, p. 197) modelling procedure meaning that it can, like other finite difference methods, be used to solve continuous time processes.

3.3.1.1 Markov Chains

Suppose there is a general stochastic process $\{Y(t) : t \geq 0\}$ on some state space S. Then one way to describe the evolution of this process, and hence its path, is via the conditional probability that the random variable $Y(t_n)$[1] will take some state j, given the all the states $Y(t_1), \ldots, Y(t_{n-1})$ that the path has visited in the past. If this conditional probability is based only on the previous state, or said another way, if the previous state has all the relevant information required to assess the probability of the next state then the process is said to have Markov Property. More succinctly:

Definition 3.6. *The **Markov property** of some process $\{Y(t)\}$ is said to exist*

[1]Here the notation uses t_i rather than i to avoid confusion with the general description of the state indices i, j

if:

$$\begin{aligned}&\mathsf{P}\left(\, Y(t_n) = j \mid Y(t_1) = i_1,\, Y(t_2) = i_2,\, \ldots,\, Y(t_{n-1}) = i_{n-1} \,\right) \\ &\qquad = \mathsf{P}\left(\, Y(t_n) = j \mid Y(t_{n-1}) = i_{n-1} \,\right) \end{aligned} \tag{3.22}$$

for all $i_1, \ldots, i_{n-1}, j \in S$ *and any sequence of times* $t_1 \leq \ldots \leq t_n$.

If the process is a discrete-time process then:

Definition 3.7. *A* ***discrete-time Markov chain*** *is any discrete-time process* $\{Y_t\}$ *that satisfies the Markov property. The transition probability from state* i *to state* j *is* $p_{i,j} = \mathsf{P}\left(\, Y_t = j \mid Y_{t-1} = i \,\right)$.

When the transition matrix is known (not changing as a function of index set t) and fixed over all t it is common to assume that the Markov chain is homogeneous. That is:

Definition 3.8. *A discrete* ***homogeneous*** *Markov chain is a stochastic process with the Markov property such that*

$$\mathsf{P}\left(\, Y_t = j \mid Y_{t-1} = i \,\right) = \mathsf{P}\left(\, Y_1 = j \mid Y_0 = i \,\right). \tag{3.23}$$

is the transition probability $p_{i,j}$ *for every* (t_n, t_{n-1}) *in* $t_1 \leq \ldots \leq t_n$ *so that the probability of changing states is independent of the time* t_n *of that transition.*

If the stochastic process $\{Y(t)\}$ is a homogenous, discrete-time process over a finite state space the the chain is known as a finite Markov chain and it permits description by:

Definition 3.9. *A* ***transition matrix*** *or* ***transition kernel****,* $\mathbf{P} = (p_{i,j})$*, that is the* $|S| \times |S|$ *matrix of transition probabilities from state* i *to state* j *as described by the Markov property of the process 3.6. This matrix is a* ***stochastic matrix*** *which means that all entries* $p_{i,j} > 0$ *and row sums* $\Sigma_i p_{i,j} = 1$*. If both the row and columns sums equal* 1 *then the matrix is called a doubly stochastic matrix.*

Of particular interest are the k-fold iterations of the stochastic matrix, $\mathbf{P}^0 \cdot \mathbf{P}^1 \cdots \mathbf{P}^k$. If $\mathbf{P}(t, t+k)$ is the k-step ahead transition matrix and $p_{ij}(t, t+k)$ are the k-step head transition probabilities then Grimmett et al. (2001, p. 215) use the Chapman-Kolmogorov equations, combined with the property of homogeneity above, to show that after repeated iterations $\mathbf{P}(t, t+k) = \mathbf{P}^k$. That is, after k matrix multiplications of the transition matrix starting at some arbitrary index t, the distribution of the chain is $\mathbf{P}^k$ where $\mathbf{P}^k$ contains the probability of going to state j after k steps starting from state i at the t^{th} index.

If $\boldsymbol{\mu}_0$ is defined as the row vector of initial probabilities for random variable Y_0, i.e. $\mathsf{P}\,(\,Y_0 = i\,)\;\forall i \in S$, and $\boldsymbol{\mu}_t$ is defined as the vector of transition probabilities for all states of Y_t at some index t then:

$$\mathsf{P}\,(\,Y_t = i; \forall i \in S\,) = \boldsymbol{\mu}_t = \boldsymbol{\mu}_0 \boldsymbol{P}^t \tag{3.24}$$

That is, the distribution of the state i of random variable Y_t after t iterations is determined by the initial distribution over the states and the t-fold products of the transition matrix. Thus each $\boldsymbol{\mu}_i$ in $\boldsymbol{P}^t$ is the marginal probability distribution of the state i over all the states j or, in other words, each row i contains the probability of going from every state j (every column in $\boldsymbol{P}$) to state i after t iterations (matrix multiplications) of $\boldsymbol{P}$. If a chain has distribution $\boldsymbol{\mu}_t$ at index t then $\boldsymbol{\mu}_t \boldsymbol{P}^t$ is the distribution of the chain at index $t+1$.

From W. Gilks et al. (1995, p. 46) some additional properties of Markov chains necessary for their use in MCMC sampling are now stated. Let $\tau_{ii} = min\{t > 0 : Y_t = j \mid Y_0 = i\}$ be the index of the first return to state i starting from state i. Then:

Definition 3.10.

- ***Irreducibility*** *If, for every state i, j in some finite Markov chain there exists $t > 0$ such that, $p_{i,j}(t) > 0$ then the chain is irreducible. That is, starting from any state i, very state j as some positive chance of being visited.*
- *If a finite chain is irreducible it is called* ***recurrent*** *if $\mathsf{P}\,(\,\tau_{ii} < \infty\,) = 1$ otherwise it is called transient. Another way of seeing this is that the sum over all transitions from state i to state j up to some large enough index t is infinite so that the chain can return to its starting state, or the chain can recur, infinitely often.*
- *An irreducible, recurrent Markov chain is called* ***positive recurrent*** *if $\mathsf{E}\,[\,\tau_{ii}\,] < \infty$ for some state i otherwise it is called null-recurrent. Thus a Markov chain that has a finite average number of revisits to each of its starting states i is positive recurrent.*
- *The of a state i is defined as $d(i) = gcd\{t : p_{i,i}(t) > 0\}$. If $d(i) = 1$ then the state i is* ***aperiodic*** *otherwise it is periodic. An non periodic chain is aperiodic for every state i in S.*

Another necessary property of a Markov process for MCMC sampling is stationarity. Stationarity of a process means that for all t, the properties of the stochastic

process are invariant. There are two versions of stationarity, strong and weak or second-order stationarity. Strong stationarity means that for any two sequences of the process $\{Y(t)\}$, $\{Y(t_1), \ldots, Y(t_k)\}$ and $\{Y(t_1+h), \ldots, Y(t_k+h)\}$ for some arbitrary discrete h, the two distributions have the same properties (Grimmett et al. 2001, p. 361). Weak stationarity is a relaxation on strong stationarity by insisting on equivalence of some properties of two adjacent random variables rather than insisting on a full joint distribution over the finite dimensional distributions mention previously in this paragraph. That is,

Definition 3.11. *A **weakly** or **second-order stationary process** is a process* $\{Y(t) : t \geq 0\}$ *where, for all* t_1 *and* t_2 *and* $h > 0$*:*

$$\mathsf{E}[\,Y(t_1)\,] = \mathsf{E}[\,Y(t_2)\,] \text{ and} \tag{3.25}$$

$$\mathsf{Cov}[\,Y(t_1),\, Y(t_2)\,] = \mathsf{Cov}[\,Y(t_1+h),\, Y(t_2+h)\,] \tag{3.26}$$

Suppose that π is the (weakly) stationary distribution of some Markov chain, called the stationary or invariant distribution of the chain, then $\pi = \pi\boldsymbol{P}$ and $\pi(j) = \sum_i^S \pi(i)p_{i,j}$, $\forall j \in S$ and assuming that $\boldsymbol{P} = \boldsymbol{P}^t$ for "enough" t.

One final property of Markov chains that is necessary for MCMC is that of ergodicity. The idea of ergodicity is that some transformation over the index set t, call it ϕ, on some measure, say ψ, preserves the measure. Let $A(t)$ be some subset of $Y(t)$. Then $\psi(A(t)) = \psi(\phi^{-1}(A(t)))$ for all $A(t) \in Y(t)$ if $\phi^{-1}(A(t))$ is a measure preserving map. Thus for any any event $A(t)$ in $Y(t)$, no matter how much the map ϕ distorts $A(t)$, if the measure ψ is invariant to begin with its will remain invariant after the transformation. Thus, for example, if one were to attempt to measure the area of a plane (or any bounded space) by taking random samples, as long as the measure is invariant, no matter how one chooses coordinates, the measure remains invariant and averaging over that measure to get the area (volume) will make sense by the law of large numbers.

The point of MCMC, which is the topic of the next section, is to sample from a bounded space in such a way that the ergodic average(s) of the measures of that space can be used to define the properties of the stationary distribution that produced the samples. In particular, a theorem taken from W. Gilks et al. (1995, p. 47) states that,

Theorem 3.1. *If a stochastic process* $\{Y(t)\}$ *is a Markov chain that is positive recurrent, hence irreducible, and aperiodic then its stationary distribution* π *is the unique probability distribution that is positive recurrent. Thus*

1. $p_{i,j}(t) \to \pi(j)$ *as* $t \to \infty$ *for all states* i, j*;*

2. ***Ergodic Theorem*** *if* $\mathsf{E}\,[\,|f(Y(t))|\,] < \infty$ *then* $\mathsf{P}\left(\bar{f}_N \to \mathsf{E}\,[\,f(Y(t))\,]\right) = 1$. *That is, if the average of some function* $f(i), i \in S$ *of the states of the Markov chain is absolutely convergent then the probability that the average of the samples* $\bar{f}_N$ *of this chain converges to the theoretical average,* $\mathsf{E}\,[\,f(Y(t))\,] = \sum_{i=1}^{S} f(i)\pi(i)$ *is* 1.

3.3.2 Markov Chain Monte Carlo

Markov chain Monte Carlo (MCMC) is a method for calculating averages via sampling. Monte Carlo integration is used to calculate difficult integrals by averaging over the outputs of some function $f(Y = y)$, assuming that $Y = y$ is sampled from some known distribution π over the domain of f. Clearly if π is a difficult distribution then

$$\mathsf{E}\,[\,f(Y)\,] \approx \frac{1}{N}\sum_{i=1}^{N} f(Y(i)), \tag{3.27}$$

the Monte Carlo average, will be hard to achieve. Using a Markov chain with the properties described above makes sampling more efficient because, if the stochastic process that is the Markov chain fulfils the properties in Theorem 3.1, then one is guaranteed that each sample drawn for the ergodic average in Equation (3.27) is from π, the invariant distribution of the chain and so, on average, one is not wasting time drawing less important parts of the domain of f.

Assume that π is the posterior of some Bayesian probability calculation. There is no guarantee that π is a stationary distribution of a Markov chain. However, it may well be possible to find some $\boldsymbol{P}$ for π so that π is a stationary distribution. Thus MCMC can be seen as the inverse of finding or creating π (Levin and Peres 2017). That is, assume that π, the posterior of some Bayesian (or some other) distribution, is a stationary distribution and the use some method to construct $\boldsymbol{P}$ so that after a "long enough" number of samples or enough repeats of the sampling process, the averages produced are the averages that describe the properties of π.

The first subsection that follows provides a general introduction to Metropolis chains and this followed by a more detailed look at Glauber dynamics (Levin and Peres 2017), a more general description of Gibbs sampling. The ideas in both of these approaches are used to design and modify the MCMC method used in this document.

3.3.2.1 Monte Carlo Integral Solving

To begin with consider a transition matrix $\boldsymbol{\psi}$ that is symmetric. Thus $\boldsymbol{\psi}$ is reversible (satisfies the detailed balance requirements $\pi(i)p_{i,j} = \pi(j)p_{j,i}$) with respect to the uniform distribution on $\{Y(t)\}$ (Levin and Peres 2017). Suppose that i is the current state of the chain with kernel $\boldsymbol{\psi}$ and choose some new state j using $p_{i,j}$ from $\boldsymbol{\psi}$. Reject this new state j with probability $1-\alpha(i,j)$ otherwise accept the new state of the chain j. This process of accepting and rejecting proposed new states generates a new Markov chain with transition matrix $\boldsymbol{P}$ defined below as:

$$\boldsymbol{P}(i,j) = \begin{cases} \boldsymbol{\psi}(i,j)\alpha(i,j) \text{ if } i \neq j \\ 1 - \sum\limits_{k \in S, k \neq j} \boldsymbol{\psi}(i,k)\alpha(i,k) \text{ if } i = j \end{cases}$$

which is the kernel of a stationary distribution π by Proposition 1.20 in Levin and Peres (2017, p. 13) i.e. if $\pi(i)\boldsymbol{\psi}(i,j)\alpha(i,j) = \pi(j)\boldsymbol{\psi}(j,i)\alpha(j,i)$. Rejecting moves to new states is wasteful so the art of developing a MCMC method is in choosing some $\pi(i)\alpha(i,j)$ as large as possible so fewer moves are rejected. The Metropolis chain achieved this by insisting that $\alpha(i,j) = \min(\pi(j)/\pi(i), 1)$ so that the kernel of the Metropolis chain is:

$$\boldsymbol{P}(i,j) = \begin{cases} \boldsymbol{\psi}(i,j)(\frac{\pi(j)}{\pi(i)} \wedge 1) \text{ if } i \neq j \\ 1 - \sum\limits_{k \in S, k \neq j} \boldsymbol{\psi}(i,k) \cdot (\frac{\pi(k)}{\pi(i)} \wedge 1) \text{ if } i = j. \end{cases} \tag{3.28}$$

The assumption of a symmetric chain is not necessary and the Metropolis-Hastings approach to MCMC provides a more general chain that requires additional terms that "balance" the Metropolis approach of Equation (3.28). Thus, rather than comparing the distributions of the states $\pi(i)$ and $\pi(j)$ directly (because if the kernel is not symmetric this may introduce a bias towards one state over the other) introduce terms $\boldsymbol{Q}(i,j)$ and $\boldsymbol{Q}(j,i)$ so that

$$\boldsymbol{P}(i,j) = \begin{cases} \boldsymbol{Q}(i,j)(\frac{\pi(j)\boldsymbol{Q}(j,i)}{\pi(i)\boldsymbol{Q}(i,j)} \wedge 1) \text{ if } i \neq j \\ 1 - \sum\limits_{k \in S, k \neq j} \boldsymbol{Q}(i,k) \cdot (\frac{\pi(k)\boldsymbol{Q}(k,i)}{\pi(i)\boldsymbol{Q}(i,k)} \wedge 1) \text{ if } i = j. \end{cases} \tag{3.29}$$

The terms $\boldsymbol{Q}(i,j)$ and $\boldsymbol{Q}(j,i)$ are proposal kernels which, from Equation (3.29), show that in the general MCMC case, the choice between accepting and rejecting

a new state is

$$\boldsymbol{P}(i,j) = \begin{cases} \frac{1}{\pi(i)}\{\pi(j)\boldsymbol{Q}(j,i) \wedge \pi(i)\boldsymbol{Q}(i,j)\} & \text{if } i \neq j \\ 1 - \sum\limits_{k \in S, k \neq j} \frac{1}{\pi(i)}\{\pi(k)\boldsymbol{Q}(k,i) \wedge \pi(i)\boldsymbol{Q}(i,k)\} & \text{if } i = j \end{cases} \tag{3.30}$$

the minimum of the going to the new state and staying in the current state, proportional to the current state.

Generalising Equations (3.29) and (3.30): if one makes $\pi(i) = \pi(\theta_i \mid y_t)$ and $\pi(j) = \pi(\theta_j \mid y_t)$ samples from the posterior distribution of $\theta \mid y_t$ then $\pi(\theta_i \mid y_t) \propto \mathsf{p}(\theta_i)\,\mathsf{p}(y_t \mid \theta_i)$ and $\pi(\theta_j \mid y_t) \propto \mathsf{p}(\theta_j)\,\mathsf{p}(y_t \mid \theta_j)$ where θ is the parameter of interest, $\mathsf{p}(\theta)$ is the prior belief in this parameter and $\mathsf{p}(y_t \mid \theta)$ is the likelihood of this parameter over the data sequence y_t. This makes the appeal of MCMC clear because at no point is it necessary to calculate the marginal of the data distribution. Rewriting the first part of Equation (3.30),

$$\frac{1}{\mathsf{p}(\theta_i)\,\mathsf{p}(y_t \mid \theta_i)}\{\mathsf{p}(\theta_i)\,\mathsf{p}(y_t \mid \theta_i)\,\boldsymbol{Q}(j,i) \wedge \mathsf{p}(\theta_i)\,\mathsf{p}(y_t \mid \theta_i)\,\boldsymbol{Q}(i,j)\}, \tag{3.31}$$

it is clear that to find the target posterior it is only necessary to know the prior and the likelihood of the parameter of interest, thus avoiding further complex, often intractable, calculations.

3.3.2.2 Glauber Dynamics

Glauber dynamics (Levin and Peres 2017) is a generalised term for the familiar Gibbs sampler, a very common approach to MCMC. The reason for using the generalisation is that it is intended to show that the Gibbs sampler is one example of sampling over graph configurations.

From Levin and Peres (2017), let V be a finite set of vertices of some graph G and S some finite state space. Then S^V is the set of configurations of the graph G with vertex labels S and $\mathcal{X}$ is some subset of S^V. Let π be the probability measure with support $\mathcal{X}$. For $x \in \mathcal{X}, v \in V$, let $\mathcal{X}(x,v) = \{y \in \mathcal{X} \ : \ y(w) = x(w) \text{ for all } w \neq v\}$ be the set of states agreeing with x every where but at v and define

$$\pi^{x,v}(y) = \pi(y \mid \mathcal{X}(x,v)) = \begin{cases} \frac{\pi(y)}{\pi(\mathcal{X}(x,v))} & \text{if } y \in \mathcal{X}(x,v) \\ 0 & \text{if } y \notin \mathcal{X}(x,v) \end{cases} \tag{3.32}$$

to be the distribution π conditioned on the set $\mathcal{X}(x,v)$. The rule for updating a configuration x is: pick a vertex v uniformly at random and choose a new config-

uration according to $\pi^{x,v}$. The set $\mathcal{X}(x,v)$ is the set of allowable configurations and $\pi^{x,v}(y)$ describe the transition probabilities of going from configuration x to configuration y. The very useful point of this description is that one can describe almost any type of allowable set of configurations and then perform MCMC sampling over this.

An example given by Levin and Peres (2017) is over proper q-colourings (configurations x of G so that neighbours of v do not have the same colouring) but one can imagine that many collections of configurations of nodes in G can be used.

In the Gibbs sampler suppose that $S = \Theta$ is the finite state space of the Gibbs sampler and that V is the dimension of the state space so that $\Theta = (\Theta_1, \ldots, \Theta_V)$ are the components of parameter space Θ. Then Θ^V is the set of configurations of some graph G with vertex set V. The Gibbs sampler aims to find the stationary Markov chain $\Theta^1, \ldots, \Theta^M$ that has transition matrix $\boldsymbol{P}(\Theta_i, \Theta_j) = \mathsf{p}\left(\Theta_j \mid \Theta_i\right), \forall i, j \in 1, \ldots V \times V$. This is done by iteratively updating one component of Θ, Θ_j say, conditional on the components of Θ that have already been updated, call this vector $\Theta_{j'}$. Thus the set of allowable configurations at each iteration is the set $\{\Theta_k \in \Theta_{j'}\}$ that form the conditional distributions of the transition matrix $\mathsf{p}\left(\Theta_j \mid \Theta_i\right)$. That is, $\Theta_1, \ldots \Theta_i$ are updated sequentially so that at each $m \in M$, Θ_j is conditional only on Θ_i, hence Markovian. The elements of $\boldsymbol{P}$ are the conditional distributions of the parameter vector Θ and $\boldsymbol{P}$ is a tri-diagonal matrix where each entry of the matrix is sequentially updated and full transition from state S_i to state S_j is completed when all conditional distributions have been updated.

Markov chain Monte Carlo has very many implementations with many advantages but none of which has been used in the streaming setting. These cannot all be shown here but one of the motivations behind choosing MCMC rather than other methods is that once it has been shown that MCMC can get very close to the target distribution in the streaming setting then other methods of MCMC can be employed to extend the proposed approach.

The section that follows this will give a brief description of another method of solving integrals by sampling called Sequential Monte Carlo. This method is not used in this document but it has been used in the streaming setting and is used here as a benchmark that is known to converge to correct target distribution.

3.3.3 Sequential Monte Carlo

Not as much space will be dedicated to this section an only a flavour of the method is required for the basis of comparison. A brief description of the method and the

way it is used will be provided. Sequential Monte Carlo (SMC) has been used by Taddy et al. (2011) for providing the posterior distribution of tree models and this has been adapted by Anagnostopoulos and Gramacy (2013) for the streaming setting. Perhaps appropriately, SMC is based on the particle filtering method described in Gordon et al. (1993) which has Kalman Filtering (via R.S. Bucy of Kalman-Bucy filtering (Kalman and Bucy 1961)) as one of its progenitors.

In a nutshell, SMC is a modification of sequential importance sampling that prevents all of the importance weight falling on a single particle of the empirical distribution used to approximate target posterior distribution. Algorithm 3.1 below, taken from Doucet et al. (2001, p. 11), shows how the particle filter is used:

Algorithm 3.1: SMC Bootstrap filter.

Result: The empirical posterior distribution $\mathsf{p}(z^t \mid y^t)$

At $t = 0$
for $i = 1, \ldots, N$ **do**
    Sample $z_0 \sim \mathsf{p}(z_0)$
At $t = 1$
Importance sampling step
for $i = 1, \ldots, N$ **do**
    Sample $\tilde{z}_t^i \sim \mathsf{p}\left(z_t \mid z_{t-1}^i\right)$
    Set $\tilde{z}^{t,i} = (z^{t-1,i}, \tilde{z}_t^i)$
    Evaluate importance weights $\tilde{w}_t^i = \mathsf{p}(y_t \mid \tilde{z}_t^i)$
Normalise the importance weights: $\tilde{w}_t^i = \frac{w(z^{t,i})}{\sum_{i=1}^N w(z^{t,i})}$
Particle selection step
for $i = 1, \ldots, N$ **do**
    Sample particles $z^{t,i} \sim \tilde{w}_t^i$ from the set $\tilde{z}^t$
Set $t = 2$ and repeat from the importance sampling step.

The attraction of this algorithm is its simplicity requiring, in its simplest form, only modifications to the importance distributions. It can easily be implemented in parallel and it is a black-box routine needing only importance weights and indices as inputs.

As mentioned in Section 2.3.2.2, Taddy et al. (2011) developed trees as particles. They use a resample $\rightarrow$ propagate routine for each iteration where the importance weight of the tree is generated by the tree marginal at each t and the propagation step uses the moves of Chipman et al. (1998) to generate a new tree to act as the prior for the tree model and then, using the new covariate to choose a terminal node, update the tree marginal and the leaf model statistics using only the data at the chosen terminal node.

In Doucet et al. (2001), Chapter 2, D. Crisan shows that the particle filter is guaranteed to converge to the target posterior distribution. However SMC still has a degeneracy problem (a single particle tends to be preferred over others) and methods to generate new particles have been developed (W. R. Gilks and Berzuini 2001; Andrieu et al. 2010) that combine MCMC and SMC. The former generates particles as one does SMC but uses MCMC proposal moves (resample and move algorithm) to keep the particle distribution from degenerating. In the latter case, the authors use SMC to generate proposal distributions for MCMC sampling where the target distribution is high dimensional distributions and the where the parameters of the posterior might also be correlated.

4 A Regression Model for Streaming Data

4.1 Introduction

The concept behind this thesis is to learn about the state of nature of an observed process by sampling a set of possible filters. The set of filters are indicated, or pointed to, by tree models based on explanatory variables and the rules embedded within the tree structure. However, not only is the set of possible tree models very large but, in the streaming setting, there is a demand that the state of nature should be available as soon as is practically possible. Thus an ensemble of tree models, each adapting to the changing state of nature by assessing a completed likelihood, is proposed. This ensemble of models both covers some portion of the space of tree models and aids in the search of models in the nonstationary streaming environment.

Consider the ensemble model from Proposition 1.1 where:

- $\mathcal{F}$ is an ensemble function over a set of tree models $T(\cdot,\, x^t,\, \xi_T(t))$.
- $T(\cdot,\, x^t,\, \xi_T(t))$ provide a set of random bases for filtering models $g(Z_{t,b}, \psi_b, t)$, $b \in 1, \ldots, K_T$ where K_T is the number of parts (leaves) in the partition created by T and
- $g(Z_{t,b}, \psi_b, t)$ is a linear Kalman filter model for the latent process Z^t that, for each discrete t, has some probability of being updated where the probability of an update is based on the random tree function $T(\cdot,\, x_t,\, \xi_T(t))$ at each t.

Then Equation (1.2) can be written as:

$$\begin{aligned} Y^t &\approx \mathcal{F}(x^t,\, \Theta,\, \varepsilon^t, t) \\ &= \sum_{\forall T \in \mathcal{F}} T(g(Z^t, \psi_T, t),\, x^t,\, \varepsilon^t, \xi_T)\, \mathsf{P}\left(T \mid Y^t, g(Z^t, \psi_T, t),\, x^t,\, \varepsilon^t, \xi_T\right). \end{aligned} \tag{4.1}$$

which is a mixture of tree models $T(\cdot)$ over an approximation to the distribution of these models weighted by $\mathsf{p}\,(\,T \mid \cdot\,)$. As such the Equation (4.1) is a composition of model functions which will be called model constituents to distinguish them from the usual terminology of mixture models which describe the individual models as components.

This composition of approaches to modelling has been adapted to the streaming setting. This setting imparts conditions and restrictions that are different to the familiar model assumptions, formulations and approaches that were presented for each these constituents in Chapter 2.

The immediate sequel, Section 4.2, provides a detailed description of each of the constituents of the model. More specifically;

- an adaptation of the intermittent filter $g(Z_{t,b}, \psi_b, t)$ in Equation (4.2) for inference on the state;
- a revision of the approach to random generation, selection and modification of the tree models, $T(\cdot, x^t, \xi_T)$, in Section 5.2 to using MCMC in the streaming data setting and finally;
- in Section 5.4 an ensemble $\mathcal{F}(\cdot)$ approach to MCMC and regression which generates a posterior weighted mixture of trees for prediction and model search is presented.

The next part of this chapter will delve more deeply into the tree filter model. This tree filter is a stepping stone for the next chapter but is of fundamental importance because the tree filter not only improves upon the Kalman filter (Kalman 1960) if it is suspected that there may be more than one process worth tracking but provides a means for specifying the conditions under which the Kalman filter can adapt to these suspected different processes. The result is an adaptation of the Intermittent Kalman filter (IKF) (Sinopoli et al. 2004) to the case where there is more than one filter, the update of the filter is conditionally structured on the tree model and at every iteration at least 1 filter is updated if an observation is available.

The next part of the chapter will be a detailed calibration study for the leaf parameters of the tree filter. Unlike the IKF where interest lies predominantly in maintaining a continuous signal or in control of the devices that rely on the signal, the aim of this tree filter is to be able to reduce a complex signal to a simpler collection of signals that must operate for an undefined length of time without the opportunity for control so leaf parameter choice is important. This calibration study has further implications for the next chapter where the number of leaves is

randomly changing and ensuring a probability of an update is impossible.

The final part of this chapter are simulation studies that explore the effect of the calibration study on various aspects of the simplest form of tree model: the 2-leaf model.

4.2 Model Description

The purpose of this section is to provide a detailed description of the proposed model as described in Equation (4.1). Chapter 4 contains only part of the fully proposed model so this section is meant to act as an overview of contents of this chapter and Chapter 5, where the reminder of the model is presented and demonstrated.

This chapter focuses on leaf modelling and inference assuming a fixed tree model. The next chapter considers the case where the tree models are changing and model selection using MCMC acts a method for adapting models to the changing stream environment.

When the tree structure is fixed and known the tree acts to provide a set of conditional bases for a collection of Kalman filters. The contribution of this model is to adapt the IKF into a tree filter where, conditional on the tree, the support of the observation process can be specified more accurately.

The tree filter model has two constituents of the model presented in Equation (4.1). The first of these constituents is an intermittent filter at the each of the leaves:

$$g(Z_{t,b}, \psi_b, t) = \begin{cases} z_{t,b} = F_{t,b} z_{t-1,b} + \mathsf{N}(0, W_{t,b}) & \text{if } \eta_{x_t,T} \neq b \\ \begin{cases} z_{t,b} = F_{t,b} z_{t-1,b} + \mathsf{N}(0, W_{t,b}) \\ y_{t,b} = H_{t,b} z_{t,b} + \mathsf{N}(0, V_{t,b}) \end{cases} & \text{if } \eta_{x_t,T} = b \end{cases} \tag{4.2}$$

where $\eta_{x_t,T} \in \{2, \ldots, K_T\}$ is a function that indicates the leaf number chosen by x_t in tree T.

The prediction and estimate of the level at each of the leaves is

$$\hat{\mu}_{t|t-1,b} = F_{t,b} \hat{\mu}_{t-1|t-1,b} \qquad \text{if } \eta_{x_t,T} \neq b \tag{4.3}$$

$$\hat{\mu}_{t|t,b} = \hat{\mu}_{t|t-1,b} + K_t S^{-1} \varepsilon_t \qquad \text{if } \eta_{x_t,T} = b \tag{4.4}$$

where

$$\varepsilon_t = y_t - H_{t,b}\hat{\mu}_{t|t-1,b}$$
$$S_{t,b} = V_{t,b} + H_{t,b}\hat{\Sigma}_{t|t-1,b}H_{t,b}^T$$
$$K_{t,b} = \hat{\Sigma}_{t|t-1,b}H_{t,b}^T S_{t,b}^{-1}$$

The estimate of the covariance is

$$\hat{\Sigma}_{t|t-1} = F_{t,b}\hat{\Sigma}_{t-1|t-1,b}F_{t,b}^T + W_{t,b} \qquad \text{if } \eta_{x_t,T} \neq b \tag{4.5}$$
$$\hat{\Sigma}_{t|t} = \hat{\Sigma}_{t|t-1,b} - K_{t,b}H_{t,b}\hat{\Sigma}_{t|t-1,b} \qquad \text{if } \eta_{x_t,T} = b \tag{4.6}$$

Notice that these are the IKF updates from Section 2.2.2.1 except that each filter is subscripted with a b to indicate which of the parts $b \in \{2, \ldots, K_T\}$ the filter belongs to.

The second constituent is the tree model

$$T(g(Z^t, \psi_T(t), t),\ x^t,\ \varepsilon^t, \xi_T(t)) \tag{4.7}$$

where

$g(Z^t, \psi_T(t), t)$ is from Equation (4.2);

$$\varepsilon_t = y_t - H_{t,\eta_{x_t,T}}\hat{\mu}_{t|t-1,\eta_{x_t,T}} \tag{4.8}$$
$$\xi_T(t) = (\alpha, \beta, p, B) \tag{4.9}$$

x^t is the set of sequential (assumed) deterministic covariates. The error ε_t is a function of the tree because it changes depending on which leaf is chosen at each t. That is, it is dependent on a particular $g(\cdot)$ from Equation (4.2). In Equation (4.9) α and β have the same rôle as they do in the Bayesian tree model of Chipman et al. (1998). p is the number of covariates which could be altered by an analyst (G. Box and Luceño 1997) but also vary depending on their nature. B are the bounds of the covariates. More detail on this in Chapter 5 because in this section the tree model is fixed.

A fixed tree model means a simplified version of the tree model in Section 2.3.2.2. All tree models in this document start as a weak-learner: a tree with a single root node, a single covariate x_j at the root node and threshold value $c_j \in B = (0, 1)$. The x_j and c_j are deterministically chosen in the fixed tree model and do not change throughout the analysis. In this section only a 2-leaf tree is considered because this section and the 2-leaf tree are building blocks for the models in

Chapter 5 that follows. However, as Section 4.3 shows, by forming a completed marginal likelihood for the tree model over the leaf filter that is randomly updated at each t and each of those filters at the leaves that are not updated, the tree filter can outperform the Kalman filter if, for example, one wanted to separate signals or track particular objects based on some conditional rules. This makes the tree filter a useful contribution itself although fully exploring this is not the aim of this section nor this document.

Constituent two is taken up more fully in the next chapter and a more detailed description would be best left to there. In short, in that chapter the fixed the tree model is allowed to change form based on a randomly sampled structure and rules that were designed by Chipman et al. (1998). Their approach has been modified for the streaming setting but the idea for the random sampling process for tree modelling is theirs. The difference between their approach and this one is that there the search is over a finite data set for a collection of appropriate models that would then be reviewed by an expert. Here, the trees are sampled for as long as the analysis runs so that estimation via the Kalman filter, learning and choosing models and adapting to data is all performed on-the-fly. This approach breaks with the train - test - validate - predict approach that is usually, in other tree but also more general models, done in separate stages and often cycled over.

The contribution of that chapter is a Gibbs-within-Metropolis approach to MCMC using trees and model bounds to select an appropriate set of coordinates for model search and then using a Metropolis-Hastings acceptance ratio to choose between trees. The marginal likelihood for the fixed tree model of the previous subsection is incorporated into the evolving tree model in a simple way.

To do the train, estimate, predict and learn routine on the fly in a sensible way in a probabilistic setting means that there must be some kind of well-behaved distribution from which one can sample trees. This leads on to the final constituent of the model in Equation (4.1): the mixture over the ensemble.

$$\begin{aligned} Y^t &\approx \mathcal{F}(x^t,\, \Theta(t),\, \varepsilon^t, t) \\ &= \sum_{\forall T \in \mathcal{F}} T(x^t,\, \Theta(t),\, \varepsilon^t, t)\, \mathsf{P}\left(T \mid Y^t, x^t,\, \Theta(t),\, \varepsilon^t, t\right) \end{aligned} \tag{4.10}$$

is the complete model where $T(\cdot)$ is from Equation (4.7) and $\Theta(t)$ represents all parameters in the model including those of Equation (4.2). Each of the trees $T(\cdot)$ represents a sample from the space of possible tree models but also each tree represents a Markov chain. This is taking the tree sampling stochastic process of Chipman et al. (1998) to heart because this potentially very long sampling process forms a stochastic trajectory over the space of trees. If the sample space

were finite then Chipman et al. (1998) show that this trajectory converges to the target posterior for the tree using Markov chain Monte Carlo methods. Taddy et al. (2011) and Anagnostopoulos and Gramacy (2013), using SMC (Section 3.3.3), show that by generating an ensemble of tree particles one can maintain the target posterior for the tree model when new data is sequentially added. The third constituent of the proposed model differs from both of these approaches because it is an MCMC method: the tree models are compared and learned by accepting/rejecting new models based on new data. However it is also an ensemble method because there are $\mathcal{F}$ Markov chains that are used to cover more of the tree model space sooner than a single chain might achieve. MCMC depends on a finite sample space for the target distribution to be attained and for the ergodic averages to make sense so it is not expected that at $t+1$ the target distribution from t for T will achieved by any one of the trees individually. However, the mixture model of Equation (4.10) sums over all posterior weighted outputs of the tree model thereby combining estimates and predictions so that at each t, it is not the output of a single chain that provides information about the parameters of the target distribution. Thus, as will be demonstrated in Chapter 5 and more convincingly in Chapter 6, the difference between the "true" target distribution and the approximation to the target distribution produced by the ensemble is negligible.

4.3 Tree filter

The model at each leaf of the tree is an intermittent Kalman filter (Sinopoli et al. 2004) (IKF) where the intermittency arises as only one leaf is updated at each t. The difference between the IKF and the model proposed here is that in this case both the updated and nonupdated filters must be considered at each t and each leaf b. Thus, assuming that a tree has a fixed structure (number of leaves, covariates and rules), there are a fixed number of filters that can be chosen from at each t to provide an estimate of the level of the latent process Z^t and hence the observed process Y^t.

Algorithm 4.1 provides a simplified view of the intermittent tree filter. In this section both the x_j and c_j are fixed and known so the conditional statement around c_j can be ignored. It is included here for some generality in the fixed tree approach. Also note that while it is possible to include waiting times between predictions and updates (and between known covariates and random observations) the approach in this document is that each $d_t = (x_t, y_t)$ arrives as a single data point and so 1-step ahead predictions and updated estimates are not output

separately.

Algorithm 4.1: Streaming algorithm for a fixed tree filter.

Result: At each $t \in 1, \ldots, :$ $\mu_{t|t,b}$, the estimate of the latent process Z^t at index t, $\Sigma_{t|t,b}$, the variance-covariance of the estimate of Z^t and $\log \mathsf{p}\left(Y_t \mid x_t, T, \psi_T \right)$, the log of the integrated likelihood at t

```
Initiate the fixed tree structure from tree adjacency matrix:
foreach η ∈ I_T do
    Get x_j assigned to η
    if c_j exists then
        assign c_j
    end
    else
        Get B(x_j) = (LB_j, UB_j)
        Choose c_j ~ U(LB_j, UB_j)
        Update I(η_j)
    end
end
foreach b ∈ L_T do
    Get ψ_b = (H_b, V_b, F_b, W_b, μ_{0,b}, Σ_{0,b})
    Initialise the Kalman Filter at each leaf
    Initialise log p( Y_t | x_t, T, ψ_T )
end
Start the streaming analysis:
while !STOP do
    foreach b ∈ L_T do
        Predict μ_{t|t-1,b}
    end
    Wait for x_t
    Using x_t, choose the filter at leaf b from T
    Wait for y_t
    Using y_t, update the Kalman Filter at leaf b
    Output μ_{t|t,b}, Σ_{t|t,b}, log p( Y_t | x_t, T, ψ_T )
end
```

Bearing in mind that a tree model is strongly conditional on the choice of rule and coordinate at the root (Denison, C. C. Holmes, et al. 2002), the tree acts as a prior for a set of possible models. The root covariate and threshold value can be considered the main effect on the observation. The covariates and thresholds at the nodes in the subtree act as interaction effects on this main effect. Thus one way to interpret the leaves of the tree is that they present a set of possible interaction models conditional on the main effect and hence the leaves that are not updated provide a set of predictions of the latent state process Z^t that complete the support of the likelihood of the nonstationary process Y^t. The leaf that is chosen at index t provides a conditional likelihood for observation at $Y_t = y_t$ and

so marginalising over the tree at each t provides a completed marginal likelihood for the observation $Y_t = y_t$.

An issue with the tree filter model is that there are 6 parameters,

$$\psi_b = (H_b, V_b, F_b, W_b, \mu_{0,b}, \Sigma_{0,b}),$$

that must be specified for each of the filters at each of the leaves. Resolving this issue can done by replicating the approach of Chipman et al. (1998) to the sampling of the covariate values by assuming that these parameters come from a set of known parameters that are selected either uniformly or deterministically at each t. The lack of identifiability of Kalman filter parameters precludes inferring all parameters and in the streaming setting this would place too much of a time burden on the process. Sampling all parameters is also problematic because stipulating and adapting the distributions from which samples are taken is difficult in the streaming setting. The calibration study of Section 4.4 presents a deterministic method for setting a large number of parameters. It is included in this chapter because examining the leaf parameters is the focus here but the the main consequence of the calibration study is the ease with which one can initialise a large ensemble of trees in a sensible way and have some idea of how the leaf filters will perform and affect the tree filter as the tree model changes.

The first part of the subsection that follows will give a simple demonstration of the proposed model in the form of a fixed tree model of only two leaves. This simple model, called the base or 2-leaf model, is suitable to introduce the main ideas proposed in this document and for the exploration of the ideas that are necessary to perform inference on the latent state process without being distracted by model selection. This subsection will formalise the probability model previously described with a particular focus on the model likelihood.

The third part of this subsection will provide, firstly, a simulation study will to investigate the effects of random updates on the filter estimates and tree marginal for the 2-leaf tree with different update probabilities and secondly when there are more than two leaves in the tree. In both of these cases the parameters of for the filters, ψ_T will be fixed. Thus the third simulation study will focus on the effect of random updates on the tree estimates and marginal likelihood but across different sets of parameters.

4.3.1 2-leaf Model

This section will begin with a description and demonstration of a specific 2 leaf tree model with linear Kalman filters at each of the leaves. This will be followed by the specification of the probability model for the proposed modelling approach. The derivation of the tree marginal likelihood which is supported by the derivations in Appendix A1 in particular, Appendix A1.1.1 is the main focus of this section with the development of the prior and posterior deferred to Section 5.2.

The data for this demonstration was simulated from a binary tree with a threshold at the root node of 0.5. The single covariate x_{t1} has its values uniformly chosen from $(0, 1)$ hence each of the leaves has a uniformly random chance of being selected at each t. The 1-dimensional observation at a chosen leaf is generated from a 1-dimensional latent process:

$$z_{t,b} = f^s_{t,b} z_{t-1,b} + N(0, w^s_{t,b})$$
$$y_{t,b} = h^s_{t,b} z_{t,b} + N(0, v^s_{t,b})$$

where each of the leaf parameters $\psi^s_b = (h_b, v_b, f_b, w_b)$ are specified in Table 4.3.1a and the superscript s shows that the these parameters are those chosen for the simulation.

The 2-leaf model is a binary tree model with a single covariate at the root node. The threshold of this covariate is 0.5 so that each leaf $b = 2, 3$ has a 50% prior chance of being updated because all covariate values are normalised to be in the range $(0, 1)$. Conditional on the tree, the 1-dimensional leaf models for each $b = 2, 3$ are:

$$z_{t,b} = f_{t,b} z_{t-1,b} + N(0, w_{t,b})$$
$$y_{t,b} = h_{t,b} z_{t,b} + N(0, v_{t,b})$$

and the 1-dimensional predictions for the level parameters $\hat{\mu}_{t,2}, \hat{\mu}_{t,3}, \hat{\sigma}_{t,3}$ and $\hat{\sigma}_{t,3}$ are:

$$\hat{\mu}_{t|t-1,b} = f_{t,b,b} \hat{\mu}_{t-1|t-1,b}$$
$$\hat{\sigma}_{t|t-1,b} = f^2_{t,b,b} \hat{\sigma}_{t-1|t-1,b} \tag{4.11}$$

at both leaves at every index t. If a data input $x_{t1} < 0.5$ then the left leaf model is chosen to be updated and the right leaf model level parameter prediction is propagated to the next iteration. If the known input $x_{t1} \geq 0.5$ then opposite happens with the right filter being updated and the left prediction being propagated.

The update equations for the estimates of the level parameters are:

$$
\begin{aligned}
\mathrm{I}_{\mathrm{t,b}}\,\tilde{y}_{t,b} &= \mathrm{I}_{\mathrm{t,b}}(y_t - h_{t,b}\hat{\mu}_{t|t-1,b}) \\
\mathrm{I}_{\mathrm{t,b}}\,s_{t,b} &= \mathrm{I}_{\mathrm{t,b}}(h_{t,b}^2\hat{\sigma}_{t|t-1,b} + v_{t,b}) \\
\mathrm{I}_{\mathrm{t,b}}\,k_{t,b} &= \mathrm{I}_{\mathrm{t,b}}\left(\frac{h_{t,b}\hat{\sigma}_{t|t-1,b}}{s_{t,b}}\right) \\
\hat{\mu}_{t|t,b} &= \hat{\mu}_{t|t-1,b} + \mathrm{I}_{\mathrm{t,b}}(k_{t,b}\tilde{y}_{t,b}) \\
\hat{\sigma}_{t|t,b} &= \hat{\sigma}_{t|t-1,b} - \mathrm{I}_{\mathrm{t,b}}(k_{t,b}h_{t,b}\hat{\sigma}_{t|t-1,b})
\end{aligned}
\tag{4.12}
$$

where $\mathrm{I}_{\mathrm{t,b}} = 1$ if the leaf is chosen and 0 otherwise.

Thus at every t at least one model must be updated and all other leaf models provide a prediction for the their trajectory. Note that the value x_{t1} is only used to choose the appropriate model. This value, described here as an input, does not form part of the prediction or update step other than to indicate which model subspace of $\mathcal{Z}_t = b$ best describes the latent trajectory of Y_t at index t. The predictions and updates are only functions of their prior trajectories and, by the Markov assumption, of their position in the state subspace at index $t-1$.

The estimates of the levels are not direct functions of the inputs so that the leaf models are independent of $x^t \in \mathcal{X}$. The benefit of this is that, from the streaming point of view, there is no need to store any of the input data: the observation and explanatory variables are only needed at the particular index at which they arrive; the tree stores the information about the covariate subspaces and the leaf models store the necessary latent information about the trajectories to both predict and estimate the level at the current index and some future indices of the sample spaces $t+k, Z^{t+k}$ and Y^{t+k}.

The parameters for the 2-leaf model filters are specified in Table 4.3.1b. For this demonstration these parameters have been chosen to help explain some of the properties of the model. Choosing parameters more generally will be explained by the calibration study.

Leaf	Simulation Parameter Values					
	v	w	f	h	μ_0	w_0
2	1	1	0.75	1	0	1
3	0.5	3	0.5	1	1	1

(a) Leaf Simulation Parameters

Leaf	Model Parameter Values					
	v	w	f	h	μ_0	w_0
2	1	1	0.95	1	0	1
3	0.5	1	0.75	1.5	1	1

(b) Leaf Model Parameters

Table 4.3.1

Figure 4.3.1 presents the state estimates for the 2 leaf tree model. Each of the leaf

mean estimates for the latent process are denoted $\hat{\mu}_{t|t,2}$ and $\hat{\mu}_{t|t,3}$; the observation process by y_t; the latent process by the dashed line z_t and the estimates chosen by the tree model as $\hat{\mu}_{t|t,tree}$. Also shown on this graph is the trajectory of the linear Kalman filter $\hat{\mu}_{t|t,kal}$ which is the black dashed line. The parameters used for this filter are the same as those used for leaf 2 As Figure 4.3.1 shows, there is not much to distinguish between the linear Kalman filter and the tree filter, where in some cases the Kalman filter estimate of the level at index t is closer to the simulated state estimate and in some cases the tree filter estimate is. This was to be expected because the Kalman filter is an optimal linear filter and both leaf simulations are from a linear filter. However, the advantage of the tree filter can be seen in Figure 4.3.2 where the graph shows the 95% estimate bounds for both the tree and Kalman filters.

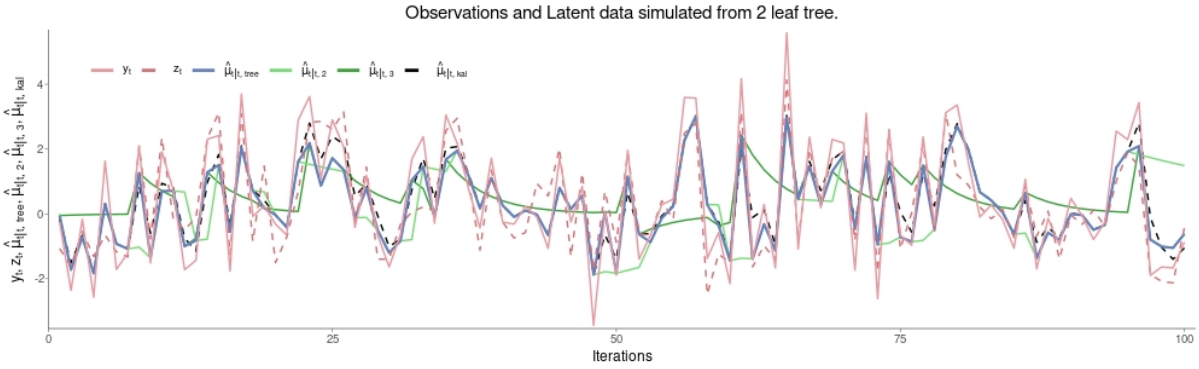


Figure 4.3.1: Tree and leaf filter output for the 2-leaf model.

In Figure 4.3.2 the dashed lines show the pointwise errors of the trajectories of the tree and Kalman filters which are the solid lines in the graph. The third solid line is the observation trajectory. Here it is clear that by using the tree filter the bounds of the pointwise estimates are tighter than by just using the Kalman filter estimates alone. Note that a direct comparison between the tree and filter

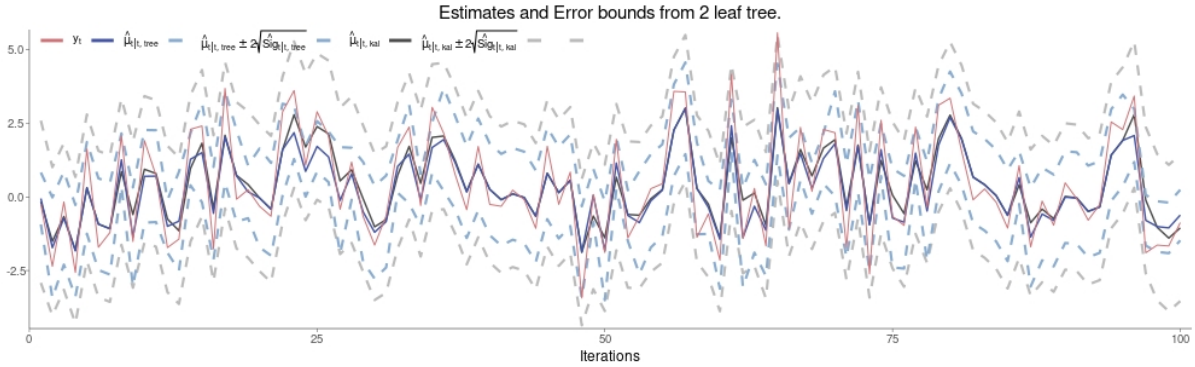


Figure 4.3.2: Tree and Kalman filter output for the 2-leaf model.

trajectories would be the incorrect approach. To make a valid comparison one

would need to include the model(s) that were not selected for the tree filter via the marginal likelihood of the tree (as opposed to the conditional marginal likelihoods for each of the leaf filters). Including the nonupdated model broadens the support for the distribution of the observed process so that as values tend towards the tails (away from the mean of the process) of the distributions the density of these values is greater than would be by using the tighter pointwise conditional marginal distribution that is depicted in Figure 4.3.2.

Having something more concrete to discuss the next section will present the tree filter probability model. The main focus in this section is the tree likelihood and its benefits to filtering. The calibration study that follows the probability model forms part of the tree prior concept but the rules and modifications made to this prior will be further detailed in Section 5.2. The posterior distribution of the tree is important for model prediction but again these formulations and descriptions will be more appropriate in later sections.

4.3.2 The Probability Model

The proposed probability model for the tree is a joint distribution of a tree model T, the observation process $Y^t = \{Y_i\}_{i=1}^t$ and a latent data process $Z^t = \{Z_i\}_{i=0}^t$ that completes the observation process. This model can be factorised into the prior model for the tree, the initial distribution for the latent process, and the joint distribution of observations that are assumed conditionally independent and exchangeable given the explanatory variables x^t and the latent process Z^t:

$$\mathsf{p}\left(T, Y^t, Z^t \mid x^t, \theta_T\right) = \mathsf{p}\left(T \mid \xi_T\right) \mathsf{p}\left(Z_0 \mid T, \psi_T\right) \cdot \prod_{i=1}^{t} \mathsf{p}\left(Y_i \mid T, Z_i, \psi_T, x^t\right) \mathsf{p}\left(Z_i \mid Z_{i-1}, T, \psi_T, x^t\right) \tag{4.13}$$

Using Bayes' rule the joint posterior probability for the tree and the latent process is:

$$\mathsf{p}\left(T, Z^t \mid x^t, Y^t, \theta\right) \propto \mathsf{p}\left(T \mid x^t, \xi_T\right) \mathsf{p}\left(Y^t \mid x^t, Z^t, T, \psi_T\right) \cdot \mathsf{p}\left(Z^t \mid x^t, T, \psi_T\right) \tag{4.14}$$

where x^t are the independent variables and $\theta_T \subset \Theta$ are a set that contains the parameters of the model, including ψ_T for the Kalman Filter models and ξ_T for the tree model, but excluding latent parameter process Z^t which has been made explicit.

The tree provides a prior domain and a set of parameters for the filter model. The leaves of the tree are independent *apriori*, hence, conditional on the tree, the filters are independent *aposteriori*. Thus it is possible to have a tree filter that is a product of independent leaf filters. In this approach the term tree filter does not imply that the trees themselves are filtered (hence this is different from Taddy et al. (2011)). At no point is the joint distribution of the trees over t calculated. Rather the tree acts a means of sampling from the space of tree models at each t and the independent trajectories of each of the filters evolves from $t-1$ to t conditional on the sample model of the latent space provided by the tree.

The prior $\mathsf{p}\left(T \mid x^t, \xi_T\right)$ is broadly based on the prior modelling of Chipman et al. (1998). Thus the tree prior is a tree sampling process as described in Section 2.3.2. However, a major difference between the proposed approach and that of every other Bayesian (or other) tree modelling approach is that at no point is there a specification on the number of data points that must be acquired or assigned to a partition before a rule or structural change is deemed allowable (Item BCARTMS rules 1, Section 2.3.2.2). Rather, the allowable rules for selecting covariates in the tree are specified independently of the data. For example, the allowable rule could be that every covariate in the tree is unique or that every internal node η has no matching covariate in $\mathrm{P_A}(\eta)$. Another rule to determine whether a threshold, and by extension a covariate, is allowable is to determine whether any choice of threshold would render one of the leaves of the tree unreachable. It is this rule that has implemented in this paper because was used to ensure that the support of the marginal for the tree was complete. Exactly what this means and how it is implemented will be discussed in Section 5.2 but it suffices now to state that without this rule or other similar rules the tree filter would not make sense because unreachable leaves means that impossible alternate models are being included as a part of the measure of the tree marginal likelihood.

The posterior, $\mathsf{p}\left(T, Z^t \mid x^t, Y^t, \theta_T\right)$, provides a joint measure over the tree and the latent variables. Factorising this leads to

$$\begin{aligned}\mathsf{p}\left(T, Z^t \mid Y^t, \theta_T, x^t\right) &= \mathsf{p}\left(T \mid Y^t, \theta_T, x^t\right) p(Z^t \mid T, Y^t, \psi_T, x^t\big) \\ &= \underbrace{\mathsf{p}\left(T \mid Y^t, \theta_T, x^t\right)}_{\text{Ⓐ Posterior for tree model}} \times \underbrace{\prod_{b=1}^{K_T} \mathsf{p}\left(Z_b^t \mid T, Y^t, \psi_T, x^t\right)}_{\text{Ⓑ Product of Gaussians from the IKF}}\end{aligned} \tag{4.15}$$

where the product of Gaussian densities is provided by the product of the Kalman filter densities at each leaf. If a leaf is updated then the density is a Gaussian with

mean $\hat{\mu}_{t|t,b}$ and variance-covariance $\hat{\Sigma}_{t|t,b}$ and if the leaf is not updated then the density is also Gaussian but with mean $\hat{\mu}_{t-1|t,b}$ and variance-covariance $\hat{\Sigma}_{t-1|t,b}$. This product of independent densities provides a measure over all the trajectories conditional on the proposed tree model and is updated via a Bayesian filtering process as described by Meinhold and N. D. Singpurwalla (1983) and others. Thus the observed data model $\mathsf{p}\left(\, Y_i \mid T, Z_i, \psi_T \,\right)$ of Equation (4.13) is incorporated into the posterior.

The posterior for the tree model, Ⓐ, provides a measure of the tree model itself upon which the product of Gaussians in Ⓑ is dependent. This measure is found by using Bayes' rule to factor out the marginal likelihood of the tree over all the latent processes proposed by the tree prior:

$$\begin{aligned}\mathsf{p}\left(\, T \mid Y^t, \theta_T, x^t \,\right) &\propto \mathsf{p}\left(\, T \mid \Theta_T \,\right) \mathsf{p}\left(\, Y^t \mid T, \theta_T, x^t \,\right) \\ &= \mathsf{p}\left(\, T \mid \Theta_T \,\right) \underbrace{\int \mathsf{p}\left(\, Y^t \mid T, Z^t, \theta_T, x^t \,\right) \mathsf{p}\left(\, Z^t \mid T, \theta_T, x^t \,\right) \, dZ^t}_{\text{Ⓒ Marginal for } T \text{ over latent process } Z^t}\end{aligned} \tag{4.16}$$

It is Ⓒ, the tree marginal that will form central part of the rest of this section. The posterior of the tree model will be revisited in Sections 5.2 and 5.4 as it is used to weight the mixture of trees.

4.3.3 The Marginal Likelihood of the Tree

In Section 2.2.2 the recursive approach to the latent process estimation provided a coherent probabilistic structure for a single filter. Namely, starting with initial states for the parameters $\mu_{0,b}$ and $\Sigma_{0,b}$ that have initial Gaussian distribution $N(\mu_{0,b}, \Sigma_{0,b})$ which forms the prior for the state process, it is possible, at each t, to recursively estimate the parameter $\hat{\mu}_{t|t,b}$ of the latent process and to determine $\Sigma_{t|t,b}$ (assuming constant ψ_b) by using the likelihood $\mathsf{p}\left(\, e_t \mid Y_{t-1}, Z_t \,\right)$ to update the posterior $\mathsf{p}\left(\, Z_t \mid Y_t, Y_{t-1} \,\right)$. This posterior is the prior predictive distribution for the following update step.

The IKF (Sinopoli et al. 2004) had a similar recursive procedure for estimating both $\hat{\mu}_{t|t,b}$ and $\hat{\Sigma}_{t|t,b}$ however there was an additional random variable in the likelihood, γ_t, such that $\mathsf{p}\left(\, e_t \mid Y_{t-1}, Z_t, \gamma_t \,\right)$ and so that $K_{t,b}$, the Kalman gain, and hence $\Sigma_{t|t,b}$ are random variables. For each leaf in the proposed model this is still the case: a single leaf is an intermittent filter with the same recursive procedure as described in Section 2.2.2 and the leaf likelihood is the same as the leaf likelihood for the IKF.

However, in the case of the tree filter both the updated and nonupdated filters need to be taken into account to form the tree likelihood at each t. This forms a central part of the proposed approach to streaming modelling: It is assumed that the observed process Y^t is nonstationary and hence it is not possible to know exactly what the support of the observation process is because the properties of a general nonstationary process can not be defined (Section 3.3.1). For example, a nonstationary process might jump, at some random index t, from one distribution to another. A set of models could better describe the support of Y^t if that set of models could be proposed in a way that covers a range of possible distributions for the observation process. Using a tree model allows one to specify just such a set of plausible models (in this case filters) by providing a set of coordinates conditional on the coordinate at the root node.

However, because the model is imperfect and there is noise in the observation, when a particular model is chosen the error in this choice of model (filter) must be taken into account. In other words, one must also include, by marginalising over the random processes at the leaves, the other (predicted) possible explanations for the observation. It is in this sense that it has been stated that, for each t, the tree model forms a completed marginal likelihood for an observation.

This not unusual. In the tree models presented in Section 2.3.2 the tree model is marginalised over as a method of using the sample data to learn about the a set of models that best describe a static population. What is different in this approach is that here the interest lies in providing a conditional set of auxiliary data for a single observation to account for the nonstationarity of the observation process. That is, the observation at each index t is assumed to be a single sample from a population of processes and a selection of these processes are modelled by the leaves of the tree.

Assume that T is known so that the number of leaf filters are fixed. Then the tree filter likelihood is calculated by marginalizing over the state variable of all the leaves of the tree and over all discrete time steps from 0 to t. Starting from Ⓒ in Equation (4.16):

$$\begin{aligned}
\mathsf{p}\left(Y^t \mid x^t, T, \cdot\right) &= \int \mathsf{p}\left(Y^t \mid x^t, Z_t, T, \cdot\right) \mathsf{p}\left(Z_{t-1} \mid T, x^t, \cdot\right) dZ^t \\
&= \prod_{b \in K_T} \int \mathsf{p}\left(Z_{0,b} \mid T, \cdot\right) \prod_{i=1}^{t} PdY_i \mid Z_{i,b}, T, \cdot \, \mathsf{p}\left(Z_{i-1,b} \mid T, \cdot\right) dZ^t \\
&= \prod_{b \in K_T} \int \mathsf{p}\left(z_{0,b} \mid T, \cdot\right) \prod_{i=1}^{t} \mathsf{p}\left(y_{i,b} \mid z_{i,b}, T, \cdot\right) \mathsf{p}\left(z_{i-1,b} \mid T, \cdot\right) dz^t
\end{aligned} \tag{4.17}$$

where the change to lower case was for visual clarity in the full derivation and this is continued here. The integral assumes that the other filter model parameters are fixed and known for all t. As each of the components of the integral are Gaussian one can use completing the square to get a recursive solution to the integral in Equation (4.17):

$$\begin{aligned}\mathsf{p}\left(Y^t \mid x^t, T, \cdot\right) &= \prod_{b=1}^{K_T} \int \mathsf{p}\left(z_{0,b}\right) \prod_{i=1}^{t} \mathsf{p}\left(y_{i,b} \mid z_{i,b}, T, \cdot\right)^{\mathrm{I_{i,b}}} \mathsf{p}\left(z_{i,b} \mid z_{i-1,b}, T, \cdot\right) dz^t \\ &= \mathsf{p}\left(T \mid \cdot\right) \prod_{b=1}^{K_T} (2\pi^m |W_{0,b}|)^{-\frac{1}{2}} \left(\prod_{i=1}^{t} (2\pi^m |W_{t,b}| 2\pi^m |A_{t,b}|)^{-\frac{1}{2}} (2\pi^n |V_{b,t}|)^{\frac{-I_{i,b}}{2}} \right) \cdot \\ &\exp\left[-\frac{1}{2} \left(\mu_{0,b}^T W_{0,b}^{-1} \mu_{0,b} - d_{0,b}^T A_{0,b} d_{0,b} + \sum_{i=1}^{t} I_{i,b} y_{i,b}^T V_{t,b}^{-1} y_{i,b} - d_{i,b}^T A_{i,b} d_{i,b} \right) \right]\end{aligned} \tag{4.18}$$

where

$$A_{0,b} = W_{0,b}^{-1}$$

$$d_{0,b} = W_{0,b}^{-1} \mu_{0,b}$$

$$A_{t-1,b}^{*} = \left(A_{t-1,b} + F_{1,b}^T W_{1,b}^{-1} F_{1,b} \right) \tag{4.19}$$

$$A_{t,b} = \left(W_{t,b}^{-1} + \mathrm{I_{t,b}}\, H_{t,b}^T V_{t,b}^{-1} H_{t,b} - v_{t-1}^T A_{t-1,b}^{-1} v_{t-1} \right) \tag{4.20}$$

$$d_{t,b}^T = \left(\mathrm{I_{t,b}}\, V_{t,b}^{-1} H_{t,b} y_{t,b} - d_{t-1,b}^T A_{t-1,b}^{-1} W_{t,b}^{-1} F_{t,b} \right) \tag{4.21}$$

The full derivation from Equation (4.17) to Equation (4.21) can be found as part of the derivation of marginal posterior for the tree in Appendix A1.1. Note that in the IKF, γ_t represented the chance that a filter was updated at any t. In the above derivation the symbol $\mathrm{I_{t,b}}$ has been used to indicate when the observation does or does not form part of the calculation for a particular leaf marginal but at every discrete t at least one leaf filter is always updated.

The presentation of the proposed model now returns to the the 2-leaf demonstration. Having described the tree marginal likelihood above in Appendix A1.1.1 a 1-dimensional version of the recursive equations are provided. Figure 4.3.3 shows several components of the log tree marginal likelihood: the lines denoted ℓm_2 and ℓm_3 are independent leaf marginals for each of the two leaves from $t = 0$ to t. The equations for these leaf marginals can be found in Appendix A1.1.1. The line denoted $\log \mathsf{p}\left(T \mid y\right)$ is the sum of the log leaf marginals and this includes the log of the prior $\mathsf{p}\left(T\right)$ while the line indicated by $\sum \ell m_b$ is purely the sum of the leaf marginals. The line shown by $\log \mathsf{p}\left(T, z \mid y\right)$ is the joint log posterior of the tree which includes the sum of the marginals, the prior and the sum of the log densities for each of the leaves. Finally, the line denoted ℓm_{kal} is the

log likelihood of the Kalman filter estimates which are calculated recursively as follows:

$$\ell m_k = \ell m_{k-1} - \frac{1}{2}\left(\frac{\tilde{y}^2}{s_{kal}} + \log(s_{kal}) + \log(2\pi)\right) \tag{4.22}$$

$$\text{where } s_{kal} = h^2\sigma^2_{t|t-1} + v^2 \text{ is the innovation variance.} \tag{4.23}$$

Returning to Figure 4.3.2 the advantage of using the tree filter, shown there by narrower bounds on the estimates, can be seen here by the fact that the log tree likelihood (and log tree posterior) of the model are (for the most part) more probable conditional on the tree model than just using the Kalman filter alone. As already mentioned, the reason for this is that the tree model increases the support of the filtering process, in the case of the 2-leaf model, to two possible distributions at each index t while the Kalman filter has the support of only a single distribution. Thus a more extreme value for the Kalman filter will have a lower log density than a tree model density because the estimated value will occur closer to the tails of the Kalman filter likelihood than it will in the case of tree model, assuming that the tree model has correctly chosen the leaf that best estimates the level parameter.

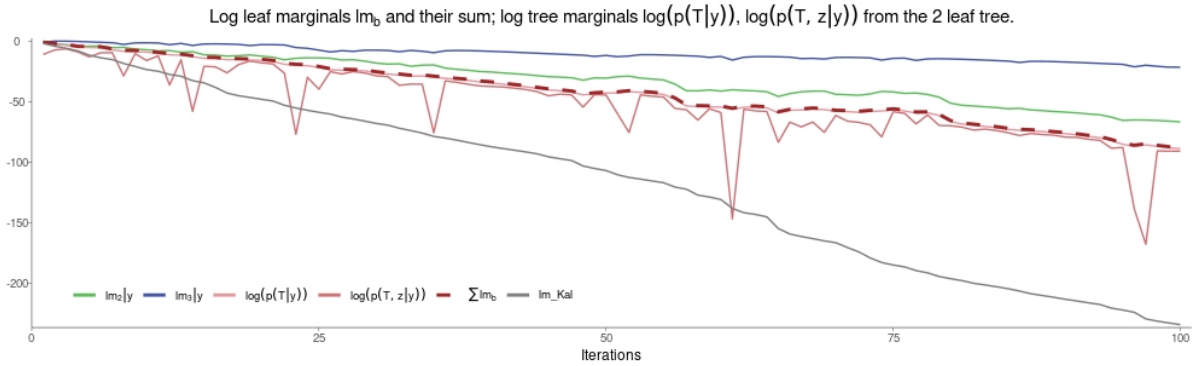


Figure 4.3.3: Tree and leaf filter output for the 2-leaf model.

In this simple demonstration the simulated tree and tree model were exactly the same up to the filter parameters. Thus no error in model design is accounted for in Figure 4.3.3. This is not the case in general and in Section 5.2 a method for choosing between two alternate models will be presented. This choice between two models will still not be adequate to explore the tree model space so that in Section 5.4 an ensemble of posterior weighted tree models is presented to attempt to cover the tree model space, at least up to the index t.

The next part of this section will present a more detailed look at the components of the tree and leaf likelihoods. This will be a precursor to the next section which is a calibration study of the parameters for leaf models. The supporting

calculations for this discussion can be found in Appendix A1.1.1, in particular the 1-dimensional log marginal parameters:

$$\begin{aligned}
\boldsymbol{a}_{0,b} &= \frac{1}{w_{0,b}^2} \\
\boldsymbol{d}_{0,b} &= \frac{\mu_{0,b}}{w_{0,b}^2} \\
\boldsymbol{a}_{t-1,b}^* &= \boldsymbol{a}_{t-1,b} + \frac{f_{t,b}^2}{w_{t,b}^2} && (4.24) \\
\boldsymbol{a}_{t,b} &= \frac{1}{w_{t,b}^2} + \mathrm{I}_{\mathrm{t,b}} \frac{h_{t,b}^2}{v_{t,b}^2} - \frac{f_{t,b}^2}{\boldsymbol{a}_{t-1,b}^* w_{t,b}^4} && (4.25) \\
\boldsymbol{d}_{t,b} &= \mathrm{I}_{\mathrm{t,b}} \frac{h_{t,b}}{v_{t,b}^2} y_{t,b} - \frac{f_{t,b}}{\boldsymbol{a}_{t-1,b}^* w_{t,b}^2} \boldsymbol{d}_{t-1,b} && (4.26)
\end{aligned}$$

and the recursion equations of these parameters,

$$\begin{aligned}
\boldsymbol{a}_{t,b}^0 &= \frac{\boldsymbol{a}_{0,b}}{w_{t,b}(\sum_{i=0}^{t-1} f^{2i})\boldsymbol{a}_{0,b} + f^{2t}} && (4.27) \\
\boldsymbol{a}_{t,b}^1 &= \frac{(h^2 w + v)\boldsymbol{a}_{t-1,b} + h^2/v f_{t,b}^2}{w_{t,b}\boldsymbol{a}_{t-1,b} + f_{t,b}^2} && (4.28)
\end{aligned}$$

and

$$\begin{aligned}
\boldsymbol{d}_{t,b}^0 &= \boldsymbol{d}_{0,b} \prod_{i=0}^{t} (-1)^i c_{i,b}^0 && (4.29) \\
\boldsymbol{d}_{t,b}^1 &= \frac{h_{t,b}}{v_{t,b}} (\sum_{i=1}^{t} y_i + \sum_{i=1}^{t-1} (-1)^i y_i \prod_{j=1}^{i} c_{j,b}^1) + \boldsymbol{d}_{0,b} \prod_{i=0}^{t} (-1)^i c_{i,b}^1 && (4.30)
\end{aligned}$$

where

$$c_{t-1,b}^{\mathrm{I_{t,b}}} = \frac{f_{t,b}}{w_{t,b}\boldsymbol{a}_{t-1,b} + f_{t,b}^2} \tag{4.31}$$

In what follows the marginal random component $\boldsymbol{a}_{t,b}$ will known as the marginal scale parameter and the random component $\boldsymbol{d}_{t,b}$ will be called the marginal residual parameter. The recursions in Equations (4.27) and (4.28) to (4.30) have superscripts $0, 1$ which indicate that these recursions are for the cases when a leaf is not or is, respectively, updated at each t. The importance of these recursions, especially those of Equations (4.27) and (4.28) is that they provide upper and lower bounds for the random variable $\boldsymbol{a}_{t,b}$. By looking at Equation (4.25), were it not for the indicator variable $\mathrm{I_{t,b}}$, this parameter would be completely determined by $\psi_b = (H_b, V_b, F_b, W_b)$. Similarly, considering Equation (4.26), if the indicator $\mathrm{I_{t,b}}$ were not present then this equation would be a random function of the data Y_t, the known parameter $\boldsymbol{a}_{t,b}$ and ψ_b. Thus, having some prior knowledge of

ψ_b and calibrating these provides some sensible values that one might structure alternatives for a tree model and the ensemble as a whole.

Figure Figure 4.3.4 shows the behaviour of the marginal scale parameter for the parameters shown in Table 4.3.1b. Each graph depicts a single leaf with the darker lines showing how $a_{t,b}$ responds to when a leaf is chosen and the lighter line showing how $1/a_{t,b}$ responds. The dashed lines are upper and lower bounds for $a_{t,b}$ and the dotted lines for $1/a_{t,b}$. The first thing to notice in both graphs is

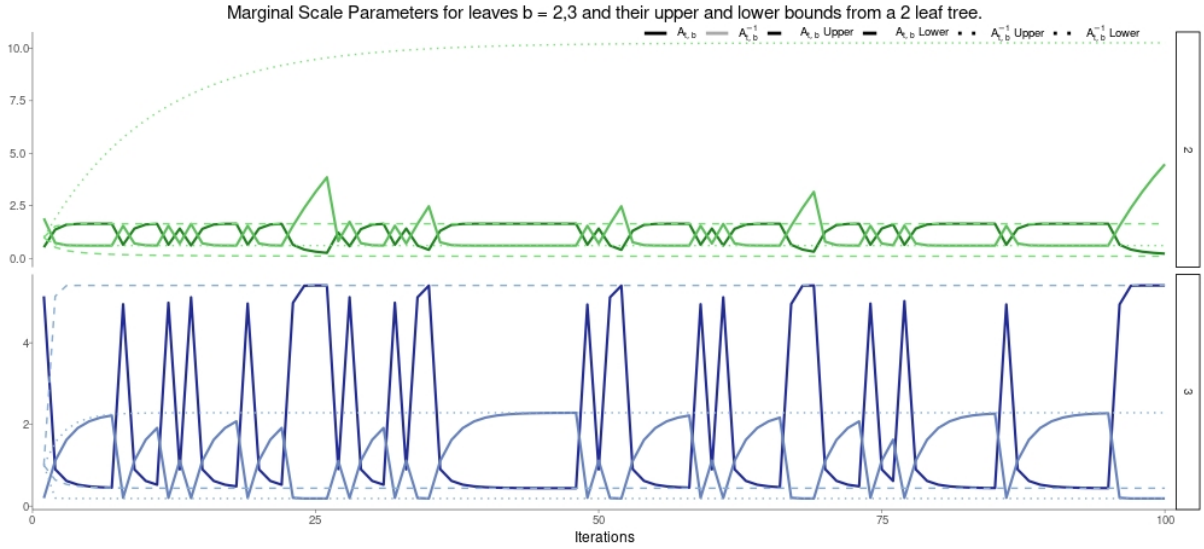


Figure 4.3.4: Individual leaf marginal scale parameter components $a_{t,b}$ and $1/a_{t,b}$

that the marginal scale parameter randomly oscillates according to leaf updates. However, for the most part these oscillations are bounded between either $a^0_{t,b}$ and $a^1_{t,b}$ or between $1/a^0_{t,b}$ and $1/a^1_{t,b}$. In the graph for leaf 2, the upper bound for $1/a_{t,b}$ is far above the general oscillation bounds. This difference is almost completely attributable to parameter F_b, the state matrix (in the 1d case this is just a value that is equal to its eigenvalue). As Table 4.3.1b shows, for leaf 2 this value is close to 1. At $F_b = 1$ this upper bound increases in direct proportion to the number of iterations so that it is possible that after many iterations the system will transition to instability. For leaf 3, the parameter F is well below 1 and the oscillations stick strictly to the bounds.

Keeping the marginal scale parameter between bounds is necessary because in the streaming setting the number of iterations is assumed unknown and large. Thus if $a_{t,b}$ were to converge to zero then $|\, a_{t,b} \,|$ would go to zero and $1/a_{t,b}$ would become undefined. Similarly if $a_{t,b} \to \infty$ then the scale of the tree marginal distribution would get very large, in effect tend towards infinity which would render any attempt at model comparison via MCMC untenable. Thus calibration of this parameter in particular is important for the functioning of the proposed

algorithm.

The marginal residual parameter, $\mathcal{d}_{t,b}$, is shown in Figure 4.3.5. These graphs show: a comparison between the innovation of each of the leaf filters, $\tilde{y}_b$, which is the dotted line; the data y_t, a dashed line shown in exactly the same way on both graphs; $\mathcal{d}_{t,b}$ the solid line on each graph and $\mathcal{d}^0_{t,b}$ and $\mathcal{d}^1_{t,b}$, the dot-dash lines on both graphs, are the conditional extremes of the residuals where the conditionality is based on the data vector y_t in this case but could be any Gaussian random sequence. Because $\mu_{0,b} = 0$ for both leaves, by looking at Equation (4.29), it can be seen that $\mathcal{d}^0_{t,b} = 0$ for all t.

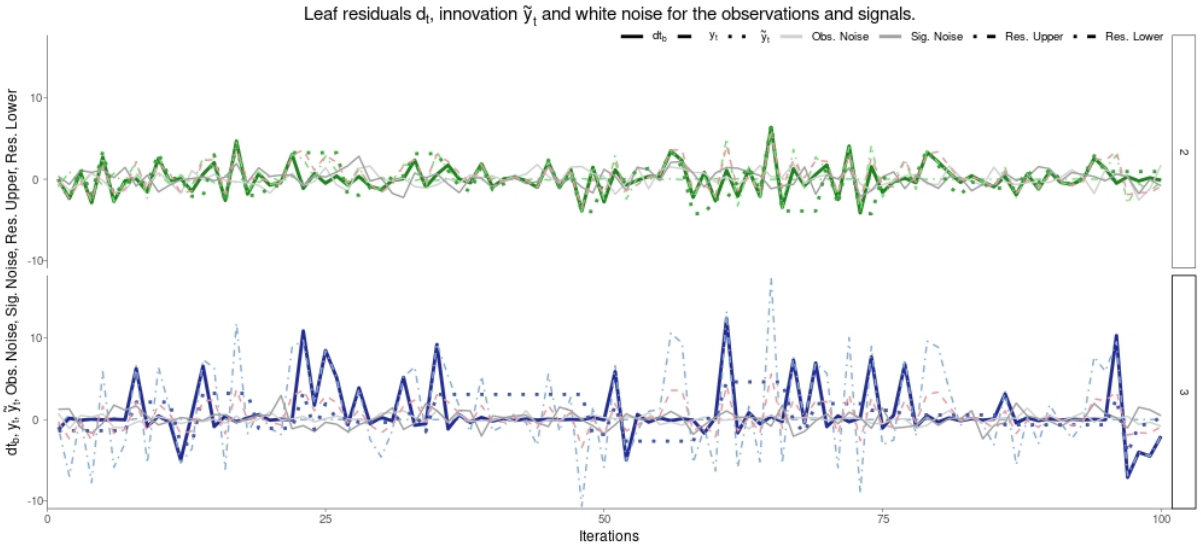


Figure 4.3.5: Individual leaf marginal residual parameter component $\mathcal{d}_{t,b}$

To understand how the parameters affect the marginal residual and the how these can be used to calibrate the leaf parameters revisit Table 4.3.1b and the graph for leaf 3 in Figure 4.3.5. The first thing to notice is the relationship between the data vector and the upper extreme marginal residual $\mathcal{d}^1_{t,3}$. For each y_t, $\mathcal{d}^1_{3,b}$ exaggerates the size of the data input while in the graph for leaf 2 both $\mathcal{d}^1_{t,2}$ and y_t are closer. The reason for this is that for $\mathcal{d}^1_{t,3}$, $h/v = 3$ while $\mathcal{d}^1_{t,2}$, $h/v = 1$.

The dotted lines are leaf innovations which, by design, are constant when a leaf is not updated and form the prefit-residual, $\tilde{y}$ from the Kalman filter when a leaf is selected. The two noise lines, drawn as solid grey lines in both graphs, are Gaussian white noise sequences of mean 0 and variance of either v or w. The reason for including these is that, in general, if the innovation or pre-fit residual, $\tilde{y}_t = y_t - h\mu_{t|t-1,b}$, follows a white noise process then it can be assumed that filter has explained all information contained within the signal. This is clearly not the case in either of the graphs shown but in the graph for leaf 2, the marginal residual is closer to a white noise process than in the graph for leaf 3.

To gain further insight into the tree and leaf marginal it is worth comparing the recursive form marginal likelihood of the Kalman filter for $t = 0$ to t,

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(\frac{\left(y_t - hf\hat{\mu}_{t-1|t-1}\right)^2}{s_t} + \log(s_t) + \log(2\pi)\right) \tag{4.32}$$

and the recursive form of the leaf marginal likelihood:

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(\frac{\mathrm{I}_{\mathrm{t,b}}\, y_{1,b}^2}{v_{t,b}} - \frac{d_{t,b}^2}{a_{t,b}} + \log\left(\frac{(2\pi)^2 w_{t,b}\, \mathrm{I}_{\mathrm{t,b}}\, v_{t,b}}{2\pi\, a_{t,b}}\right)\right) \tag{4.33}$$

which, as shown in Equations (A1.45) and (A1.46), can be written as:

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(\frac{(v a_{t-1,b}^* - h_{t,b}^2)(a_{t-1,b}^* w y_{t,b} - hf\frac{v d_{t-1,b}}{(v a_{t-1,b}^* - h^2)})^2 + f v d_{t-1,b}^2 (1 - \frac{h^2}{(v a_{t-1,b}^* - h_{t,b}^2)})}{a_{t,b} a_{t-1,b}^* \sqrt{v_{t,b}^3 w_{t,b}}} + \log\left(\frac{(2\pi)^2 w_{t,b} v_{t,b}}{2\pi\, a_{t,b}}\right)\right) \tag{4.34}$$

It has been assumed for the purposes of comparison that the leaf marginal in Equation (4.34) has been updated at every t from 0 to t. Thus this version of the leaf marginal is equivalent to the upper leaf marginals shown on both graphs of Figure 4.3.5. The important thing to notice about both innovation equations is that they are functions of the distribution of y_t and hence, in a sense, similar. The clear difference is that in Equation (4.32) all of the information about the previous state is embodied in $\mu_{t-\hat{1}|t-1}$ and s_t while in Equation (4.34) the information about the previous state is included via several terms, most notably $a_{t,b}^*$ and $c_{t,b}$, and additional information about the current state is included in $a_{t,b}$. Recalling that these parameters are random functions of the tree then it can be seen how the tree affects the innovation process and it can be seen that choosing a range of leaf model parameters ψ_b for the tree can influence the performance of the tree model.

The previous discussions about the tree and leaf marginals has paved the way for the next section which is a study for the calibration of the leaf parameters for the proposed tree model.

4.4 Calibration Study

4.4.1 Introduction

In the preceding sections it has been claimed that one reason for using the tree model in the streaming setting is to provide a set of possible models which can be used to estimate or predict events based on explanatory data x^t. The reasoning used so far is that if the data is nonstationary in the streaming setting then if there is some information that explains the observations, this information, in the form of x^t can be used to choose the most appropriate model for estimation. However there is another way to look at the choice of a tree model that has nothing to do with the nonstationarity of the data that is more in keeping with the original objectives of Chipman et al. 1998 and that is for model design and selection.

Experimental parameter values for KF						
Exp	V	W	F	H	μ_0	W_0
1	1	1	0.5	1	1	1
2	1	1	0.95	1	0	1
3	0.5	1	0.75	1.5	1	1
4	1	3	0.5	1.5	1	1
5	0.75	5	0.15	1	1	1

Table 4.4.1: Parameter values for several Kalman filters.

In Figure 4.3.2 the Kalman filter was compared to the tree filter using the parameters equivalent to those in Experiment 1 of Table 4.4.1 (and the same as those used leaf 2 of the 2 leaf model). That graph showed how the tree filter provided tighter bounds around the estimates because the covariate at the root was able to distinguish the correct distribution to use for each estimate at each t.

Figure 4.4.1 shows the set of filtering experiments from Table 4.4.1 overlaid over the tree filter. The observed data in this case is the dashed line and the simulated latent data is the thick solid line. This figure demonstrates the second reason for choosing a tree model: a single pass over some sequence of data is unlikely to produce the best or most correct model and this is especially true in the streaming setting where a single pass of the data is all that is "permitted" according the streaming data axioms (Section 2.5).

One way to learn about suitable parameters for the filter is to estimate them using one of several methods such as dual estimation or by using expectation maximization as illustrated in Haykin (2004). However these methods are unavailable in the streaming setting because they either involve several iterations of the algorithm to find a maximum, in expectation-maximisation for example, or

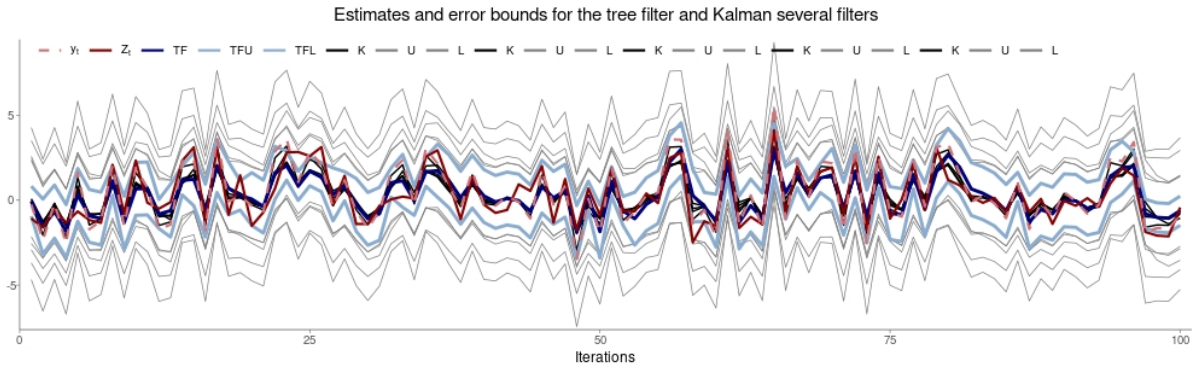


Figure 4.4.1: Several Kalman filters over the same data, each with a different parameter specification.

they require tuning (repeated runs of the same experiment) or they require sampling from some distribution which, in a hierarchical modelling scenario, requires the hyper-parameters be set. Further, the Kalman filter suffers from an issue of identifiability of parameters which limits which parameters might be inferred. Still further, $\psi_b = (V_b, H_b, W_b, F_b, \mu_{0,b}, \Sigma_{0,b})$ has 6 parameters for each leaf and for a tree with K_T leaves there are then $6 \times K_T$ parameters to specify not-to-mention that the number and position of the filters in tree is random in the case with random tree model selection.

The tree model can provide several alternatives for a model without the need for parameter estimation but this requires a rich set of parameters to choose from. Choosing parameters in this manner is akin to treating parameters as explanatory data rather than missing data. The calibration study presents a method for setting up a collection of parameters from which one can choose parameters appropriate to the problem at hand. The algorithms used in the calibration study are independent of those used for streaming an can be run before the streaming analysis has begun. The output of the analyses are bounds for sets of the parameters.

4.4.2 Method

The calibration study is performed for 1-dimensional parameters only. The algorithm uses the recursions of Equations (4.27) and (4.28), and their inverses to estimate the upper and lower bounds of $\boldsymbol{a}_{t,b}$ and $1/\boldsymbol{a}_{t,b}$. These were the upper and lower bounds shown in Figure 4.3.4. The parameter of primary interest is F_b because, in general, for the system to be stable the state matrix must have its leading eigenvalue[1] in the range $(-1, 1)$. However, setting this parameter alone may be not be enough to get the best performance out of the tree model. This

[1] which for a 1-d matrix is the value itself

can be seen by looking at both Figures 4.3.4 and 4.3.5 where the proportion between the leaf innovation estimates $d_{t,b}$ depends on several parameters and the constants h^2/v and $(h^2w + v)$ (Equation (4.26)). Equation (4.34) shows how the leaf marginal is also dependent on more than the choice of F_b although the complexity of this equation limits its direct use for analysis.

Apart from F_b there are two other relationships of interest: $r_b = W_b/V_b$ the signal to noise ratio for the leaf b and the relationship between H_b and V_b, most simply shown in Equations (4.25) and (4.26) as h_b^2/v_b and h_b/v_b respectively. Two parameters shown in Tables 4.3.1a, 4.3.1b and 4.4.1, μ_0 and W_0, have so far been ignored. These are parameters for the initial distribution of the filter and their effect is quickly forgotten for short runs of the Kalman filter. However, their importance will explained in Section 5.2 when the tree model is recursively adapted because at each adaptive move, each new filter has not had a length of run in which these parameters might be forgotten.

Table 4.4.2 provides a set of values for an initial factorial study of the parameters $\psi_b = (H_b, V_b, F_b, W_b)$ where the response is either the upper and lower bound of $a_{t,b}$, namely $a_{t,b}^1$ and $a_{t,b}^0$, or its inverse, the upper and lower bound of $1/a_{t,b}$, $1/a_{t,b}^1$ and $1/a_{t,b}^0$. These initial study values were chosen based on the requirements for stability that parameter F_b places on the leaf filter and then by considering reasonable values of H, V, W based on some idea of the simulation data. For example, Table 4.3.1a shows that the simulation parameters are $W_3^s = 3$, $W_2^s = 1$, $V_3^s = 0.5$ and $W_2^s = 1$ so choosing some values for the leaf models much greater or lower than these might provide a set of distributions too extreme for an appropriate signal to noise ratio or scale of residual (viz. Figure 4.3.5).

	Parameter Values							
	1	2	3	4	5	6	7	8
μ_0	1							
W_0	1							
H	1.5	1	0.5	0.25				
V	0.05	0.5	0.75	1	1.5	2	3	5
F	-0.75	-0.5	0.05	0.15	0.5	0.75	0.95	1
W	0.05	0.5	0.75	1	1.5	2	3	5

Table 4.4.2: Parameter values used for calibration.

Based on the parameters in Table 4.4.2 the calibration was run by considering all combinations of these values generated by the recursion equations Equations (4.27) and (4.28) over 500 iterations. The choice of 500 is arbitrary as in all cases $a_{t,b}$ converges to its conditional bound well before 500 iterations are reached but this high value highlights extreme values well as can be seen, for example, in Table 4.4.4 for the inverse scale parameter.

The values produced in the results were not generated from a tree model. This is useful because it means that for a streaming process one can formulate some prior expectations of leaf model performance without having access to the streaming data itself. However, the effect of these parameters to the tree model, in particular the tree marginal, is left to further studies and demonstrations that will appear in Chapter 5, Sections 5.3 and 5.4.2.

4.4.3 Results

Calibrating F_b and W_b

Table 4.4.3 shows the upper and lower bound values when $H_2 = 1$ and $V_2 = 1$ in the 2 leaf model. In that demonstration the value of F_b was 0.95 and $W_b = 1$. When looking at Table 4.4.4 the upper bound for these same values for $1/a_{t,b}$ is very high relative to the actual expected range of values for the signal (a Gaussian range with mean zero and variance W_2) at that leaf so a value of F_b closer to 0 might have been more appropriate.

F	a_t, W, V = 1, H = 1							
	0.05	0.5	0.75	1	1.5	2	3	5
-0.75	12.9540	3.7924	3.3388	3.0963	2.8389	2.7029	2.5606	2.4412
	1.0000	0.8750	0.5833	0.4375	0.2917	0.2188	0.1458	0.0875
-0.5	17.8792	4.0292	3.4664	3.1771	2.8802	2.7281	2.5729	2.4460
	1.0000	1.0000	1.0000	0.7500	0.5000	0.3750	0.2500	0.1500
0.05	22.2051	4.2476	3.5821	3.2492	2.9163	2.7498	2.5832	2.4500
	1.0000	1.0000	1.0000	0.9975	0.6650	0.4988	0.3325	0.1995
0.15	21.8463	4.2289	3.5722	3.2431	2.9133	2.7480	2.5824	2.4496
	1.0000	1.0000	1.0000	0.9775	0.6517	0.4888	0.3258	0.1955
0.5	17.8792	4.0292	3.4664	3.1771	2.8802	2.7281	2.5729	2.4460
	1.0000	1.0000	1.0000	0.7500	0.5000	0.3750	0.2500	0.1500
0.75	12.9540	3.7924	3.3388	3.0963	2.8389	2.7029	2.5606	2.4412
	1.0000	0.8750	0.5833	0.4375	0.2917	0.2188	0.1458	0.0875
0.95	8.8099	3.5796	3.2207	3.0199	2.7987	2.6779	2.5481	2.4362
	1.0000	0.1950	0.1300	0.0975	0.0650	0.0488	0.0325	0.0195
1	7.9269	3.5262	3.1903	3.0000	2.7880	2.6712	2.5447	2.4348
	0.0385	0.0040	0.0027	0.0020	0.0013	0.0010	0.0007	0.0004

Table 4.4.3: Bounds for the tree marginal scale parameter $a_{t,b}$ of F_b and W_b where $H_b = 1$ and $V_b = 1$.

Notice that in Table 4.4.4 when $F = 1$ and $W = 1$, $1/a_{t,b} = 500$, the exact number of iterations of the calibration study. Thus the scale of the leaf marginal likelihood would be unbounded in the streaming setting and, with a very large and unknown number iterations, the bounds of an nonupdated leaf estimate could be infinite in size which would render the MCMC algorithm, detailed in later sections, inappropriate because to compare models in that setting it is necessary to have finite expectation and variance.

Tables 4.4.5 and 4.4.6 show the upper and lower bound values of $a_{t,b}$ when $F_b =$

F	$1/a_t$, W, V = 1, H = 1							
	0.05	0.5	0.75	1	1.5	2	3	5
-0.75	1.0000	1.1429	1.7143	2.2857	3.4286	4.5714	6.8571	11.4286
	0.0772	0.2637	0.2995	0.3230	0.3523	0.3700	0.3905	0.4096
-0.5	1.0000	1.0000	1.0000	1.3333	2.0000	2.6667	4.0000	6.6667
	0.0559	0.2482	0.2885	0.3148	0.3472	0.3666	0.3887	0.4088
0.05	1.0000	1.0000	1.0000	1.0025	1.5038	2.0050	3.0075	5.0125
	0.0450	0.2354	0.2792	0.3078	0.3429	0.3637	0.3871	0.4082
0.15	1.0000	1.0000	1.0000	1.0230	1.5345	2.0460	3.0691	5.1151
	0.0458	0.2365	0.2799	0.3083	0.3433	0.3639	0.3872	0.4082
0.5	1.0000	1.0000	1.0000	1.3333	2.0000	2.6667	4.0000	6.6667
	0.0559	0.2482	0.2885	0.3148	0.3472	0.3666	0.3887	0.4088
0.75	1.0000	1.1429	1.7143	2.2857	3.4286	4.5714	6.8571	11.4286
	0.0772	0.2637	0.2995	0.3230	0.3523	0.3700	0.3905	0.4096
0.95	1.0000	5.1282	7.6923	10.2564	15.3846	20.5128	30.7692	51.2821
	0.1135	0.2794	0.3105	0.3311	0.3573	0.3734	0.3924	0.4105
1	25.9500	250.5000	375.2500	500.0000	749.5000	999.0000	1498.0000	2496.0000
	0.1262	0.2836	0.3134	0.3333	0.3587	0.3744	0.3930	0.4107

Table 4.4.4: Bounds for the inverse tree marginal scale parameter $1/a_{t,b}$ for values of F_b and W_b where $H_b = 1$ and $V_b = 1$.

0.95 and $H = 1$. Using the parameter values for leaf 2 in the 2 leaf model, the first thing to notice from Table 4.4.6 is that for every V_b (and every H_b) the upper bound of $1/a_{t,b}$ is the same. This is not unexpected because by inspection of Equation (4.27) it is clear that neither of these parameters enter the equation. The next thing to notice is that every line in Table 4.4.6 corresponds to the second last line in Table 4.4.4. Thus, for the purposes of setting the upper bound of the leaf marginal scale parameter, it is sufficient to consider only F_b and W_b. Given that for stability of the leaf marginal F_b must be in $(-1, 1)$, to choose a set of upper bounds for the leaf marginals requires only the consideration of W_b.

Looking again at Table 4.4.4 where $F_b = 0.05$ it can be seen that $1/a_{t,b}$ is extremely close to W_b and that as the values of F_b diverge towards -1 and 1, $1/a_{t,b}$ increases proportionately.

Figures 4.4.2a and 4.4.2b are based on the values in Tables 4.4.7 and 4.4.8. For values of W_b less than 1 the upper bound of $1/a_{t,b}$ is always 1 for every value of F_b between $(-0.5, 0.5)$ (although only positive values are shown because in Table 4.4.4 it was shown that changes in $1/a_{t,b}$ are symmetrical about 0). For values of $W_b > 1$, there is a constant proportional shift between values of $1/a_{t,b}$ as W_b increases. These proportional shifts are shown in Figure 4.4.2b.

Looking at Equation (4.27) one might have guessed that the relationships shown in Figures 4.4.2a and 4.4.2b were the case because $1/a_{0,b}$ has constants that include these parameters in the numerator and the denominator. But the purpose of this exercise is to calibrate some exact values that can be used to formulate some prior choices and then automate the running and adaptation of the proposed model.

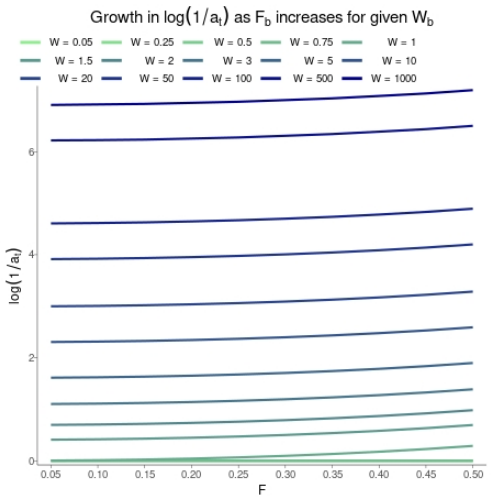
V	a_t, W, F = 0.95, H = 1							
	0.05	0.5	0.75	1	1.5	2	3	5
0.05	60.398240	46.925920	46.299558	45.980750	45.657996	45.495089	45.331136	45.199205
	1.000000	0.195000	0.130000	0.097500	0.065000	0.048750	0.032500	0.019500
0.5	12.797128	6.039824	5.597411	5.355791	5.096271	4.958291	4.813727	4.692592
	1.000000	0.195000	0.130000	0.097500	0.065000	0.048750	0.032500	0.019500
0.75	10.238738	4.420089	4.026549	3.808424	3.570527	3.442048	3.305528	3.189287
	1.000000	0.195000	0.130000	0.097500	0.065000	0.048750	0.032500	0.019500
1	8.809881	3.579565	3.220665	3.019912	2.798706	2.677896	2.548135	2.436204
	1.000000	0.195000	0.130000	0.097500	0.065000	0.048750	0.032500	0.019500
1.5	7.206845	2.698380	2.386377	2.210044	2.013275	1.904212	1.785264	1.680603
	1.000000	0.195000	0.130000	0.097500	0.065000	0.048750	0.032500	0.019500
2	6.298819	2.230422	1.949433	1.789783	1.610332	1.509956	1.399353	1.300627
	1.000000	0.195000	0.130000	0.097500	0.065000	0.048750	0.032500	0.019500
3	5.269184	1.728289	1.486948	1.349190	1.193188	1.105022	1.006637	1.000000
	1.000000	0.195000	0.130000	0.097500	0.065000	0.048750	0.032500	0.019500
5	4.292329	1.279713	1.080949	1.000000	1.000000	1.000000	1.000000	1.000000
	1.000000	0.195000	0.130000	0.097500	0.065000	0.048750	0.032500	0.019500

Table 4.4.5: Upper and lower bounds for the tree marginal scale parameter $a_{t,b}$ for values of V_b and W_b where $H_b = 1$ and $F_b = 0.95$.

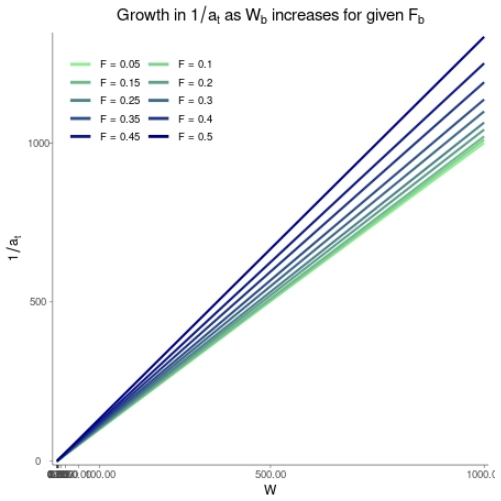
V	$1/a_t$, W, F = 0.95, H = 1							
	0.05	0.5	0.75	1	1.5	2	3	5
0.05	1.0000000	5.1282051	7.6923077	10.2564103	15.3846154	20.5128205	30.7692308	51.2820513
	0.0165568	0.0213102	0.0215985	0.0217482	0.0219020	0.0219804	0.0220599	0.0221243
0.5	1.0000000	5.1282051	7.6923077	10.2564103	15.3846154	20.5128205	30.7692308	51.2820513
	0.0781425	0.1655677	0.1786540	0.1867138	0.1962219	0.2016824	0.2077392	0.2131018
0.75	1.0000000	5.1282051	7.6923077	10.2564103	15.3846154	20.5128205	30.7692308	51.2820513
	0.0976683	0.2262398	0.2483516	0.2625758	0.2800707	0.2905247	0.3025236	0.3135497
1	1.0000000	5.1282051	7.6923077	10.2564103	15.3846154	20.5128205	30.7692308	51.2820513
	0.1135089	0.2793635	0.3104949	0.3311355	0.3573080	0.3734275	0.3924438	0.4104747
1.5	1.0000000	5.1282051	7.6923077	10.2564103	15.3846154	20.5128205	30.7692308	51.2820513
	0.1387570	0.3705927	0.4190453	0.4524796	0.4967032	0.5251516	0.5601413	0.5950246
2	1.0000000	5.1282051	7.6923077	10.2564103	15.3846154	20.5128205	30.7692308	51.2820513
	0.1587599	0.4483457	0.5129697	0.5587271	0.6209898	0.6622710	0.7146160	0.7688601
3	1.0000000	5.1282051	7.6923077	10.2564103	15.3846154	20.5128205	30.7692308	51.2820513
	0.1897827	0.5786069	0.6725185	0.7411855	0.8380906	0.9049592	0.9934064	1.0000000
5	1.0000000	5.1282051	7.6923077	10.2564103	15.3846154	20.5128205	30.7692308	51.2820513
	0.2329737	0.7814254	0.9251132	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000

Table 4.4.6: Upper and lower bounds for the inverse tree marginal scale parameter $1/a_{t,b}$ for values of V_b and W_b where $H_b = 1$ and $F_b = 0.95$.

Not much attention has been given to the lower bound of $1/a_{0,b}$ nor the upper nor lower bounds of $a_{0,b}$. This first omission will be addressed as a part of considering the values of V_b and H_b used in scaling the tree marginal residual. The value $a_{0,b}$ occurs in the denominator of the tree marginal as Equation (4.18) shows. Clearly this value must not reach zero for the entirety of the streaming analysis and, from the point of view of computational stability, should not get too small either. However, choosing F_b and W_b as suggested so far, with judicious choices of V_b and H_b to be suggested should prevent this.



(a) Behaviour of W_b as F_b increases.



(b) Behaviour of F_b as W_b increases.

Figure 4.4.2

F	$1/a_t$, W, V = 1, H = 1												
	0.05	0.5	0.75	1	1.5	2	3	5	10	50	100	500	1000
0.05	1	1	1	1.0025	1.5038	2.0050	3.0075	5.0125	10.0251	50.1253	100.2004	502.5126	1000.000
0.1	1	1	1	1.0101	1.5152	2.0202	3.0303	5.0505	10.1010	50.5051	101.0101	505.0505	1010.101
0.15	1	1	1	1.0230	1.5345	2.0460	3.0691	5.1151	10.2302	51.1509	102.2495	510.2041	1020.408
0.2	1	1	1	1.0417	1.5625	2.0833	3.1250	5.2083	10.4167	52.0833	104.1667	520.8333	1041.667
0.25	1	1	1	1.0667	1.6000	2.1333	3.2000	5.3333	10.6667	53.3333	106.6098	531.9149	1063.830
0.3	1	1	1	1.0989	1.6483	2.1978	3.2967	5.4945	10.9890	54.9451	109.8901	549.4505	1098.901
0.35	1	1	1	1.1396	1.7094	2.2792	3.4188	5.6980	11.3960	56.9801	113.8952	568.1818	1136.364
0.4	1	1	1	1.1905	1.7857	2.3810	3.5714	5.9524	11.9048	59.5238	119.0476	595.2381	1190.476
0.45	1	1	1	1.2539	1.8809	2.5078	3.7618	6.2696	12.5392	62.6959	125.3133	625.0000	1250.000
0.5	1	1	1	1.3333	2.0000	2.6667	4.0000	6.6667	13.3333	66.6667	133.3333	666.6667	1333.333

Table 4.4.7: Upper bound values for $1/a_{t,b}$ for extended values of F_b and W_b shown in Figure 4.4.2a.

F	W, $a_t[,i]/a_t[,i+1]$													
	0.05	0.25	0.5	0.75	1	1.5	2	3	5	10	20	50	100	500
0.05	1	1	1	0.9975	0.6666	0.75	0.6667	0.6	0.5	0.5001	0.4000	0.5003	0.1994	0.5025
0.1	1	1	1	0.9900	0.6666	0.75	0.6667	0.6	0.5	0.5000	0.4000	0.5000	0.2000	0.5000
0.15	1	1	1	0.9775	0.6667	0.75	0.6666	0.6	0.5	0.5001	0.4000	0.5003	0.2004	0.5000
0.2	1	1	1	0.9600	0.6667	0.75	0.6667	0.6	0.5	0.5000	0.4000	0.5000	0.2000	0.5000
0.25	1	1	1	0.9375	0.6667	0.75	0.6667	0.6	0.5	0.5001	0.4000	0.5003	0.2004	0.5000
0.3	1	1	1	0.9100	0.6667	0.75	0.6667	0.6	0.5	0.5000	0.4000	0.5000	0.2000	0.5000
0.35	1	1	1	0.8775	0.6667	0.75	0.6667	0.6	0.5	0.4999	0.4000	0.5003	0.2005	0.5000
0.4	1	1	1	0.8400	0.6667	0.75	0.6667	0.6	0.5	0.5000	0.4000	0.5000	0.2000	0.5000
0.45	1	1	1	0.7975	0.6666	0.75	0.6666	0.6	0.5	0.5001	0.3999	0.5003	0.2005	0.5000
0.5	1	1	1	0.7500	0.6666	0.75	0.6667	0.6	0.5	0.5000	0.4000	0.5000	0.2000	0.5000

Table 4.4.8: Values for $1/a_{t,b}$ where table values are the proportional increases in values found by dividing each row of Figure 4.4.2b by is succeeding row.

Calibrating V_b and H_b

Turning to the calibration of V_b and H_b, Figure 4.4.3 presents a (longer) run of the simulated data (the dashed black line) used for the 2 leaf model overlaid with a set of scaled residuals that were calculated using Equation (4.30). The top graph

shows all the various combinations of the tree likelihood residual scales assuming that $F_b = 0.95, W_b = 1, H_b = 1.5, 1, 0.5$ and for all values of V_b in Table 4.4.1. The bottom graph shows these scaled sequences but over a smaller domain and with the more extreme values removed.

Figure 4.4.3: Calibration of the tree marginal residual scale.

Table 4.4.9 shows, in the first column, the change in statistical summaries of the data vector, y_t, for different values of F_b, W_b and constant values of H_b, V_b. The values in the rest of the table are statistical summaries of the values of the leaf residual $d_{t,b}$ that change as a function of the aforementioned parameters. The table also shows the constants H_b/V_b and C_t from Equation (4.31) which is a function of constant values F_b and W_b and the previous value for $a_{t,b}$ which, in this case, is assumed to be $a^1_{t,b}$ from Equation (4.28) and that has reached its constant state.

The first thing to notice is that for all W_b, F_b, the value $H_b/V_b = 1$ and as W_b gets larger C_t gets smaller (in proportion to F_b). The next thing to notice is that, barring some small perturbations, all the statistical summaries of $d_{t,b}$ across the table are very close to each other. Thus, conditional on a fixed sequence of data, one could largely ignore the effect the values of F_b, W_b have on the scales of the residuals shown in Figure 4.4.3.

Tables 4.4.10, 4.4.11 and 4.4.12 show the changes in the statistical summaries of $d_{t,b}$ as H_b, V_b change for some selected values of F_b, W_b. As can be expected from the previous table, Table 4.4.9, comparing values of H_b, V_b across different tables produces roughly the same results, thus for example if $H_b = 1.5, V_b = 5$ for each F_b, W_b the change in the summaries of $d_{t,b}$ are minimal. The next thing to notice is that as V_b increases for each H_b, unsurprisingly, so does H_b/V_b but

F	Y		W, H = 1, V = 1							
			0.05	0.5	0.75	1	1.5	2	3	5
0.5	Mean	-0.05634	-0.0401	-0.0443	-0.0457	-0.0468	-0.0484	-0.0495	-0.051	-0.0526
	Std. Dev.	2.0381	2.2027	2.071	2.0491	2.0372	2.026	2.0218	2.02	2.0219
	Max	7.77438	8.0705	7.7858	7.7482	7.7309	7.7192	7.7181	7.7232	7.7347
	Min	-6.98183	-6.2122	-6.3515	-6.4596	-6.5367	-6.6389	-6.7034	-6.78	-6.8522
	H/V		1	1	1	1	1	1	1	1
	C_t		0.4693	0.3139	0.2679	0.2344	0.1883	0.1577	0.1193	0.0805
-0.75	Mean	-0.05634	-0.1509	-0.0924	-0.0843	-0.0794	-0.0736	-0.0702	-0.0663	-0.0628
	Std. Dev.	2.0381	3.5044	2.4981	2.3736	2.3011	2.2201	2.176	2.1297	2.0916
	Max	7.77438	8.6614	8.1593	8.0896	8.044	7.9851	7.9476	7.9019	7.8574
	Min	-6.98183	-8.7646	-6.9886	-7.0896	-7.1339	-7.1625	-7.1638	-7.1478	-7.113
	H/V		1	1	1	1	1	1	1	1
	C_t		-0.6805	-0.4313	-0.3683	-0.3233	-0.2617	-0.2208	-0.1691	-0.1158
0.05	Mean	-0.05634	-0.0501	-0.0518	-0.0524	-0.0529	-0.0535	-0.054	-0.0546	-0.0551
	Std. Dev.	2.0381	2.0206	2.0205	2.0215	2.0225	2.0245	2.0261	2.0285	2.0313
	Max	7.77438	7.7193	7.7283	7.733	7.7369	7.7431	7.7476	7.7536	7.7601
	Min	-6.98183	-6.7339	-6.8176	-6.8437	-6.8627	-6.8884	-6.9051	-6.9253	-6.9448
	H/V		1	1	1	1	1	1	1	1
	C_t		0.1427	0.0995	0.0852	0.0746	0.0597	0.0498	0.0373	0.0249

Table 4.4.9: Change in statistical values of y_t as a function of F_b, W_b for $H_b = 1, V_b = 1$

also that the scale change in the standard deviation of $\acute{d}_{t,b}$ is proportional to twice this value. As V_b increases, for each H_b, the scale of the mean of $\acute{d}_{t,b}$ decreases and as H_b decreases there is a corresponding decrease in the scale of $\acute{d}_{t,b}$ again. Finally notice that C_t is a small number that slightly increases as V_b increases and that as H_b decreases, it reaches a constant value faster for any V_b. For example, if $V_b = 0.5, H_b = 1.5$, then $C_t = 0.0211$ but for the same V_b and $H_b = 0.25$, $C_t = 0.0952$ and this is maintained for every V_b thereafter.

H	Y		W = 1, F = 0.95, V							
			0.05	0.5	0.75	1	1.5	2	3	5
1.5	Mean	-0.05634	-1.6595	-0.1487	-0.0955	-0.0696	-0.0445	-0.0325	-0.021	-0.0123
	Std. Dev.	2.0381	60.9738	6.0638	4.0574	3.0595	2.0664	1.5713	1.0758	0.6721
	Max	7.77438	232.8796	23.1554	15.4426	11.6013	7.7773	5.8742	3.9767	2.4483
	Min	-6.98183	-208.5577	-20.1463	-13.2195	-9.7785	-6.372	-4.6939	-3.1006	-1.8947
	H/V		30	3	2	1.5	1	0.75	0.5	0.3
	C_t		0.0203	0.1518	0.2017	0.2422	0.3052	0.3529	0.4219	0.4993
1	Mean	-0.05634	-1.0838	-0.0915	-0.0585	-0.0426	-0.0274	-0.0202	-0.0136	-0.0084
	Std. Dev.	2.0381	40.546	4.092	2.7695	2.1095	1.4479	1.1144	0.7468	0.4482
	Max	7.77438	155.0115	15.4866	10.396	7.8618	5.3329	4.0666	2.7203	1.6322
	Min	-6.98183	-138.3077	-12.9568	-8.4349	-6.2223	-4.1344	-3.1409	-2.1052	-1.2631
	H/V		20	2	1.3333	1	0.6667	0.5	0.3333	0.2
	C_t		0.0435	0.2599	0.3245	0.3728	0.442	0.4906	0.4993	0.4993
0.5	Mean	-0.05634	-0.4996	-0.0396	-0.0265	-0.0201	-0.0136	-0.0104	-0.0071	-0.0045
	Std. Dev.	2.0381	20.205	2.2292	1.4937	1.1202	0.7468	0.5602	0.3737	0.2248
	Max	7.77438	77.196	8.1331	5.4406	4.0805	2.7203	2.0402	1.3602	0.8161
	Min	-6.98183	-67.3864	-6.2817	-4.2104	-3.1578	-2.1052	-1.5789	-1.0526	-0.6316
	H/V		10	1	0.6667	0.5	0.3333	0.25	0.1667	0.1
	C_t		0.1403	0.4906	0.4993	0.4993	0.4993	0.4993	0.4993	0.4993
0.25	Mean	-0.05634	-0.2163	-0.0201	-0.0136	-0.0104	-0.0071	-0.0055	-0.0039	-0.0026
	Std. Dev.	2.0381	10.4184	1.1202	0.7468	0.5602	0.3737	0.2806	0.1878	0.1142
	Max	7.77438	39.0471	4.0805	2.7203	2.0402	1.3602	1.0201	0.6801	0.408
	Min	-6.98183	-31.5034	-3.1578	-2.1052	-1.5789	-1.0526	-0.7895	-0.5263	-0.5117
	H/V		5	0.5	0.3333	0.25	0.1667	0.125	0.0833	0.05
	C_t		0.3352	0.4993	0.4993	0.4993	0.4993	0.4993	0.4993	0.4993

Table 4.4.10: Change in statistical values of y_t as a function of H_b, V_b $F_b = 0.95, W_b = 1$ for

H	Y		W = 5, F = 0.5, V							
			0.05	0.5	0.75	1	1.5	2	3	5
1.5	Mean	-0.05634	-1.6869	-0.1659	-0.1097	-0.0816	-0.0536	-0.0397	-0.0261	-0.0157
	Std. Dev.	2.0381	61.1231	6.0968	4.0599	3.042	2.0249	1.517	1.0104	0.6062
	Max	7.77438	233.1915	23.2866	15.514	11.6284	7.7441	5.8031	3.8648	2.3189
	Min	-6.98183	-209.3598	-20.8521	-13.8712	-10.3813	-6.8924	-5.1489	-3.4127	-2.0476
	H/V		30	3	2	1.5	1	0.75	0.5	0.3
	C_t		0.0022	0.0211	0.0308	0.0401	0.0573	0.0731	0.0952	0.0952
1	Mean	-0.05634	-1.1218	-0.1084	-0.0711	-0.0526	-0.0347	-0.0261	-0.0174	-0.0105
	Std. Dev.	2.0381	40.7326	4.0542	2.6986	2.0219	1.3471	1.0104	0.6736	0.4042
	Max	7.77438	155.4285	15.5001	10.3219	7.7347	5.1531	3.8648	2.5765	1.5459
	Min	-6.98183	-139.4944	-13.8273	-9.1761	-6.8522	-4.5503	-3.4127	-2.2752	-1.3651
	H/V		20	2	1.3333	1	0.6667	0.5	0.3333	0.2
	C_t		0.0049	0.0445	0.0634	0.0805	0.0952	0.0952	0.0952	0.0952
0.5	Mean	-0.05634	-0.5538	-0.052	-0.0347	-0.0261	-0.0174	-0.0131	-0.0088	-0.0054
	Std. Dev.	2.0381	20.3277	2.0207	1.3471	1.0104	0.6736	0.5052	0.3368	0.2021
	Max	7.77438	77.6332	7.7296	5.1531	3.8648	2.5765	1.9324	1.2883	0.773
	Min	-6.98183	-69.5376	-6.8255	-4.5503	-3.4127	-2.2752	-1.7064	-1.1376	-0.6825
	H/V		10	1	0.6667	0.5	0.3333	0.25	0.1667	0.1
	C_t		0.0191	0.0952	0.0952	0.0952	0.0952	0.0952	0.0952	0.0952
0.25	Mean	-0.05634	-0.2656	-0.0261	-0.0174	-0.0131	-0.0088	-0.0067	-0.0045	-0.0028
	Std. Dev.	2.0381	10.1173	1.0104	0.6736	0.5052	0.3368	0.2526	0.1684	0.1011
	Max	7.77438	38.6998	3.8648	2.5765	1.9324	1.2883	0.9662	0.6441	0.3865
	Min	-6.98183	-34.3797	-3.4127	-2.2752	-1.7064	-1.1376	-0.8532	-0.5688	-0.3413
	H/V		5	0.5	0.3333	0.25	0.1667	0.125	0.0833	0.05
	C_t		0.067	0.0952	0.0952	0.0952	0.0952	0.0952	0.0952	0.0952

Table 4.4.11: Change in statistical values of y_t as a function of H_b, V_b $F_b = 0.5, W_b = 5$ for

To understand the relationships shown in these tables consider Equations (4.29)

H	Y		W = 0.05, F = 0.05, V							
			0.05	0.5	0.75	1	1.5	2	3	5
1.5	Mean	-0.05634	-1.6668	-0.163	-0.1085	-0.0813	-0.0541	-0.0406	-0.0271	-0.0163
	Std. Dev.	2.0381	61.0118	6.0835	4.0546	3.0405	2.0267	1.5199	1.0132	0.6079
	Max	7.77438	232.9615	23.2556	15.5011	11.6248	7.7491	5.8115	3.8741	2.3244
	Min	-6.98183	-208.7793	-20.7591	-13.8307	-10.3695	-6.9104	-5.1818	-3.4538	-2.0719
	H/V		30	3	2	1.5	1	0.75	0.5	0.3
	C_t		0.0154	0.0408	0.0435	0.0449	0.0465	0.0473	0.0482	0.0489
1	Mean	-0.05634	-1.1017	-0.1083	-0.0721	-0.0541	-0.0361	-0.0271	-0.0181	-0.0109
	Std. Dev.	2.0381	40.626	4.0538	2.7022	2.0265	1.3509	1.0132	0.6754	0.4052
	Max	7.77438	155.2014	15.4992	10.3319	7.7486	5.1655	3.874	2.5826	1.5495
	Min	-6.98183	-138.8936	-13.8243	-9.2131	-6.9086	-4.6049	-3.4533	-2.302	-1.3811
	H/V		20	2	1.3333	1	0.6667	0.5	0.3333	0.2
	C_t		0.025	0.0454	0.0469	0.0476	0.0484	0.0488	0.0492	0.0495
0.5	Mean	-0.05634	-0.5436	-0.054	-0.036	-0.027	-0.0181	-0.0136	-0.0091	-0.0055
	Std. Dev.	2.0381	20.2799	2.0263	1.3508	1.0131	0.6754	0.5065	0.3377	0.2026
	Max	7.77438	77.5228	7.748	5.1652	3.8739	2.5825	1.9369	1.2913	0.7748
	Min	-6.98183	-69.2101	-6.9067	-4.604	-3.4528	-2.3018	-1.7263	-1.1508	-0.6905
	H/V		10	1	0.6667	0.5	0.3333	0.25	0.1667	0.1
	C_t		0.04	0.0488	0.0492	0.0494	0.0496	0.0497	0.0498	0.0499
0.25	Mean	-0.05634	-0.2702	-0.027	-0.0181	-0.0136	-0.0091	-0.0068	-0.0046	-0.0028
	Std. Dev.	2.0381	10.1331	1.0131	0.6754	0.5065	0.3377	0.2533	0.1689	0.1013
	Max	7.77438	38.7442	3.8738	2.5825	1.9369	1.2912	0.9684	0.6456	0.3874
	Min	-6.98183	-34.5475	-3.4526	-2.3017	-1.7262	-1.1508	-0.8631	-0.5754	-0.3452
	H/V		5	0.5	0.3333	0.25	0.1667	0.125	0.0833	0.05
	C_t		0.0471	0.0497	0.0498	0.0498	0.0499	0.0499	0.0499	0.05

Table 4.4.12: Change in statistical values of y_t as a function of H_b, V_b $F_b = 0.05, W_b = 0.05$ for

and (4.30) and Equations (A1.38) and (A1.39) in Appendix A1. These tables have shown that the strongest influence on the scale of the leaf marginal residual

$d_{t,b}^1$, and hence any $d_{t,b}$ updated fewer times than this (because $d_{t,b}^0$ is negligible for appropriate F_b, W_b) is the value H_b/V_b. The value C_t achieves a constant value that has little influence on large values of H_b, V_b but, as shown in Table 4.4.12 when $H_b = 1.5, V_b = 5$, the mean of the data has been shrunk to such an extent that C_t is larger than the mean value. Thus choosing values of H_b, V_b can reduce the effect of the data on the leaf marginal to a large extent which might prove troublesome if the data is being used to choose tree models via the leaf marginals.

4.4.4 Conclusion

The purpose of this calibration study was to assist in the design and selection of tree models for nonstationary data. Alternatively the tree model can be used in place of parameter estimation for providing a selection of models that can be chosen based on some covariate data. This study has shown that for bounded estimates of the leaf marginal equations presented in Equation (4.33) and hence the tree marginal from Equation (4.18) irrespective of the probability of an update, one must choose F_b to have its leading eigenvalue in $(-1, 1)$. From Figures 4.4.2a and 4.4.2b it can be seen that $1/a_{0,b}^1$, the inverse of the leaf marginal likelihood scale parameter, scales in direct proportion to W_b for any given F_b. Tables 4.4.10, 4.4.11 and 4.4.12 have shown that judicious choice of H_b/V_b can help set the scale of the leaf marginal innovation that could be used ensure that the leaf filters can accommodate learning by the appropriate scaling of the distributions at the leaf nodes.

It would tedious to have to perform a calibration study of this kind for every streaming event. Thus it is necessary to minimise the inputs required for designing a tree model and, by extension, a modelling ensemble. Choosing parameters alone though is not enough. In this section and in the simulation studies to follow the trees are fixed and so the number of leaf parameters are known. However in the ensemble model of Chapter 5 the trees have a randomised structure so for an individual tree it is impossible to know the number of leaves with certainty. Thus some additional tree modelling assumptions are required. These will be detailed in Section 5.2 but in general it is assumed that each parameter choice is some deterministic function of the tree. For example, the value F_b could be positive to the right of the tree and negative to the left of the tree and could decrease in size as the either the depth or number of leaves increases. Another modelling assumption, that could be altered by the user, is that as the tree gets deeper, the parts of $\mathcal{Z}$, an hence $\mathcal{Y}$, become more homogenous so that V decreases with depth, W increases with depth or $r = W/V$ changes to reflect this idea. One aim of the simulation studies in Section 5.2 will be to examine some alternatives for

deterministically modifying parameters for model search and performance but in this section the focus on the design of single tree and its requirements assuming known but arbitrary number of leaves.

For a single tree model it is important that each leaf is different in some way. There is little reason for having the same model across different leaves because there would no differing of filter models and hence the support for the observation process would be same as for a single filter.

For a single tree model all that is required is a single value of F because its bounds are limited by $(-1, 1)$ and the value is symmetric about 0 as far as $a_{t,b}$ is concerned. There may be reasons for entertaining negative values of F_b because of the effect of z_{i-1} on z_t in the filter model. Suppose the value 0.5 was input for F_b. Then, if there are, for example three leaves, $2, 6, 7$, one option would be to choose values of F_b for each of the leaves at $0.5, 0.25, 0.25$ or, if negative values of F_b are required, $0.5, -0.25, 0.25$. These could be in any order but are allocated here according to tree depth.

Rather than fixing W_b it seems sensible to fix V_b for the tree model because even an empirical glance at nonstationary data and some prior knowledge would give the analyst some idea of the measurement noise. Then, it can be assumed, again from prior knowledge, that an appropriate scaling factor will set a broad enough range of distributions for the tree marginal residual to support the tree filtering process but without reducing the effect of the data. That is, if the mean of an empirical analysis of Y^t were around 3 then providing a range of scales between 1.25 and 0.75 for the the three leaves mentioned above might be appropriate. V, W and hence r would scale with the tree, by the assumptions mentioned above.

Stipulating one or several desired signal to noise ratios would possibly be easier than stating W_b for each leaf but one could further assume that $r = HW/V \implies Vr = HW$ would provide a set of values for W_b given, a range of inputs, V, H and r. Setting $r = 1$ implies $V = HW$ which from the above calibration study should have only a minimal recursive effect on the the expected scaling of the leaf residual and will fix, for each different value of H_b, a different but reasonable range of $1/a_{t,b}$.

To continue the previous example, choose F for a 3 leaf tree with positive state matrix values only, $F_{2,6,7} = 0.5, 0.25, 0.25$, $V = 3$ so $V_{2,6,7} = 2.5, 2, 2$, say, $H = (0.75, 1, 25)$ so that $H_{2,6,7} = 1, 0.95, 0.95$ for example then if $r = 1$, $W_{2,6,7} = 2.5, 1.9, 1.9$ which from Table 4.4.8 places $1/a_{t,b}$ in the range of values very close to $(2.1333, 2.6667)$ for each of the leaf marginal likelihoods. Thus the user has to input F, V and possibly H and r, and by using some spacing of values that

is tree size (number of leaves) and/or tree height dependent the tree parameters can be automatically allocated.

In the simulation studies that follow this will be the process that is followed. In this section there are two simulation studies, the first considers the effect of incorrectly specifying the splitting threshold value given a fixed set of leaf parameters on the tree marginal likelihood and state estimates when compared to the Kalman filter and the true model presented above. The second study examines how specifying the leaf model parameters affects the tree marginal and estimates for a few different threshold values.

4.5 Leaf model simulation studies

The calibration study above has provided one possible way of approaching a sensible method for choosing a set or sets of leaf model parameters that allows one to avoid parameter estimation in a streaming setting; to allow for nonstationarity of the data and to guard against extreme values of parameters that may cause computational irregularities and impossibilities such as zero determinants and divide-by-zero errors. One task of the simulation studies to follow is to evaluate this method.

There are two main parts to the simulation study. The first part, Section 4.5.1, examines the accuracy of the estimates and the effects on the tree filter marginal over a range of threshold values at the root of fixed 2-leaf trees. One of the experimental tree models will exactly match the model that generates the data. In this study, ψ_T the leaf parameter sets for each experimental tree, will be exactly the same for all trees. The results of this study will show that the accuracy of the estimates depends on both the updated and predicted models at every iteration. The full study in Appendix A1.2 will explain this in detail but the results of that study will be elaborated upon in Section 4.5.1.

This 2-leaf model is the building block of every other tree so a clear understanding of this model component is warranted. More so because in the tree prior sampling process it is this model that is added and removed from a tree at some of the model evolution steps so an analysis of this model now will go some way to setting up the approach for more complex models later on.

A difficulty of presenting simulation studies in the streaming setting is deciding when to present the outcomes and results. The general inductive hypothesis used in this document is that presenting the results that have held from $t = 0$ to t means that these results will hold from t to N where N is some random but un-

known endpoint. However, in a recursive environment this inductive assumption should be treated with some caution.

In the experiments that follow the number of data points has been limited to 1000. This is mostly due to the computational, data storage and time constraints that occur from having to repeat these experiments many times. If the data are assumed to be arriving at a constant rate of one data point every $100ms$ then 1000 represents only $10s$ worth of data. While this is somewhat arbitrary (if the constant rate of arrival were assumed to be $1s$ then 1000 represents $16'40s$ of data) it is hoped that it is enough to give support to the results presented below, at least up to the 1000^{th} data point.

4.5.1 2-leaf Simulation Study 1

This simulation study compares the effect of known (as opposed to randomly chosen) wrong covariates on the filter estimates and tree marginal against both the true model and the Kalman filter. The study averages over 100 data sets simulated from a 2-leaf tree model with known leaf parameters. The results will show that, on average, the 2-leaf filter accuracy of estimating the latent state will depend on the specification of the parameters at both the leaves that are and are not updated. The tree model is not shown to be more accurate than the Kalman filter in every case but the error bounds around the tree filter estimates are narrower thus confirming that the tree filter concentrates the likelihood of the latent state by more accurately describing the support of the sequence of data than just using a Kalman filter.

Data for the experiment are generated from 100 repeated simulations from a 2-leaf tree model, T_{sim}, where there is a single covariate, x_1 with bounds $(0, 1)$ and threshold value 0.5, at the root node. Figure 4.5.1 shows 10 samples from this data generator.

The experimental trees are initialised as 2-leaf models with covariates at the root fixed at $T_1 = 0.1$, $T_2 = 0.2$, $T_3 = 0.4$, $T_5 = 0.5$, $T_6 = 0.6$, $T_7 = 0.7$, $T_8 = 0.8$, $T_9 = 0.9$. The tree that exactly matches the data generator is T_5. The data generator, T_{sim}, provides, on average, an equal number of updates and predictions at each of the leaves. The above setup means that, on average, the proportion of updates to each of the leaves varies in a consistent manner. For example, for T_1 leaf 2 gets 10% of the updates and makes predictions 90% of the time and for T_9, the model at leaf 3 gets 10% of the observed data for updating the filter and makes predictions 90% of the time. This means that not only can one compare the performance of the results to the exact model, T_5 but one can make a pairwise

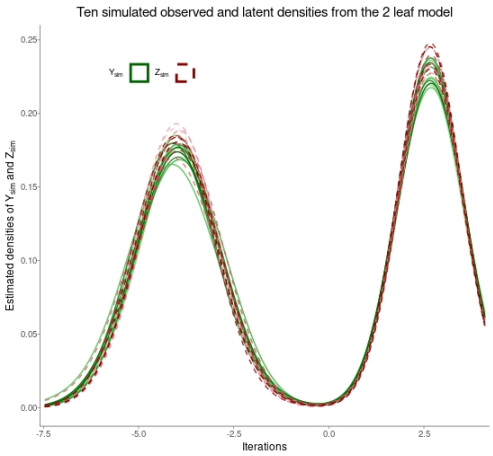


Figure 4.5.1: Ten example densities of the observed and latent distributions simulated from the 2-leaf tree model.

comparison between the models at the leaves between T_1, T_9, T_2, T_8, T_3, T_7 and T_4, T_6.

Also included in the study is the average of the Kalman filter estimates over the same set sample data. The Kalman filter parameters were fairly arbitrarily chosen based on those of the tree models used but there is only one set of parameters for the Kalman filter.

The results are presented in two parts. The first considers the accuracy of the filter estimates and the rôle of the leaf parameters in model performance. The second part looks at the tree marginal and leaf marginals and the effects of the scale parameters, their inverses and the data residual on these.

4.5.1.1 Estimate accuracy

There are two parts to this section. The first looks at the performance of the tree estimates and the second at the performance of the leaf models which includes both estimates and predictions at every t. To be clear, the tree estimates are the estimates $\hat{\mu}_{t|t,b}$ provided by a 2-leaf tree where at each t a single estimate from one of the $b = 2, 3$ leaves is output. The choice of the estimate that is output is determined by x_t at each t by choosing one of the two leaves $b = 2, 3$ of each T_k, $k = 1, \ldots, 9$. The variance of the state estimate at each t, $\hat{\Sigma}_{t|t,b}$, is a pointwise estimate from the same leaf that was chosen by x_t. The model at each leaf has different parameters so the size of the variance estimate of the tree filter is also a random variable not only dependent on when the filter was last updated but also on the values used for the parameters at each leaf.

Figure 4.5.2 shows that T_5, the model that exactly matches the data generator, is the worst performing model when comparing the cumulative mean of the averaged

tree filter estimates. The averaged Kalman filter cumulative CMSE shows that it is not the best performing filter but the tree filters that are better than it are pairs T_1, T_9, T_2, T_8. Pair T_3, T_7 fall either side of the Kalman filter estimate and Pair T_4, T_6 performs worse than the Kalman filter but better than T_5. The Kalman filter estimate has been used as a benchmark because it is the optimal filter conditional on its parameters and the mean of the estimates from this filter over 100 random data sets should, by the central limit theorem, approach the mean of the estimates generated data (this is shown in Appendix A1.2).

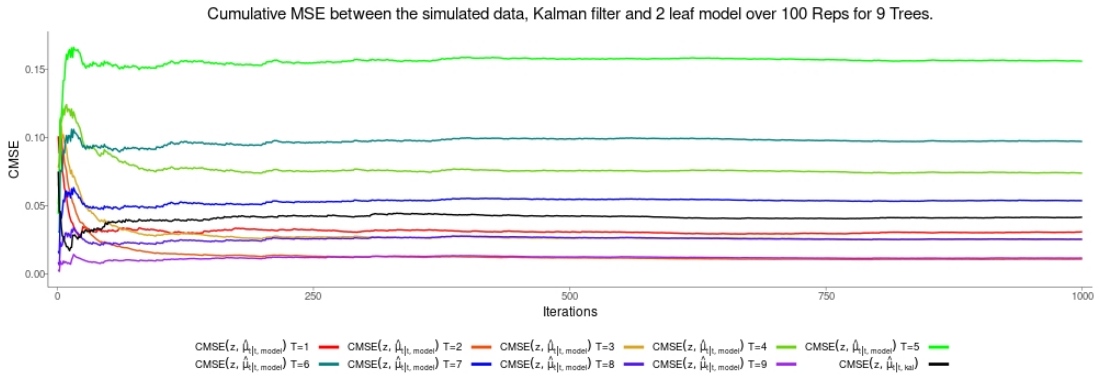


Figure 4.5.2: The cumulative MSE for 9 tree models.

The next figure, Figure 4.5.3, presents the result of comparing leaf estimates to the data generated at the corresponding leaves. The data used to generate this figure is the average over 100 data sets of both the estimates and predictions made at each leaf at each t. To be clear, at every t, every leaf produces a prediction for the filter level, $\hat{\mu}_{t|t-1}$ and if a leaf is chosen that prediction for the filter level is updated to be an estimate of the level $\hat{\mu}_{t|t}$ for the next iteration. Thus, for example, the values at $T_{5,2}$, leaf 2 of tree T_5 that exactly matches the data generator, T_{sim}, are made up, on average, of predictions 50% of the time and estimates the other 50% of the time and this is also true for $T_{5,3}$. Similarly, the proportion of predicted values to estimates depends on the value of the threshold c_k for T_k, $k = 1, \ldots, 9$ so that, for example, at $T_{9,2}$ there are, on average over 1000 iterations and 100 simulated data sets, estimates 90% of the time and predictions 10% of the time and vice versa for $T_{9,3}$.

Bearing the above in mind, Figure 4.5.3 shows the cumulative mean square error (CMSE) between $T_{sim,b}$, the latent data generated at each of the leaves of T_{sim}, and $T_{k,b}$ the combination of estimates and predictions at each of the leaves of the experimental trees. The dashed lines in Figure 4.5.3 represent the CMSE between leaves $T_{sim,3}$ and $T_{k,3}$ and the solid lines represent the CMSE between $T_{sim,2}$ and $T_{k,2}$. The result is in direct contrast to that shown in Figure 4.5.2 where now the best performing model is T_5, the worst performing model, on average, is the

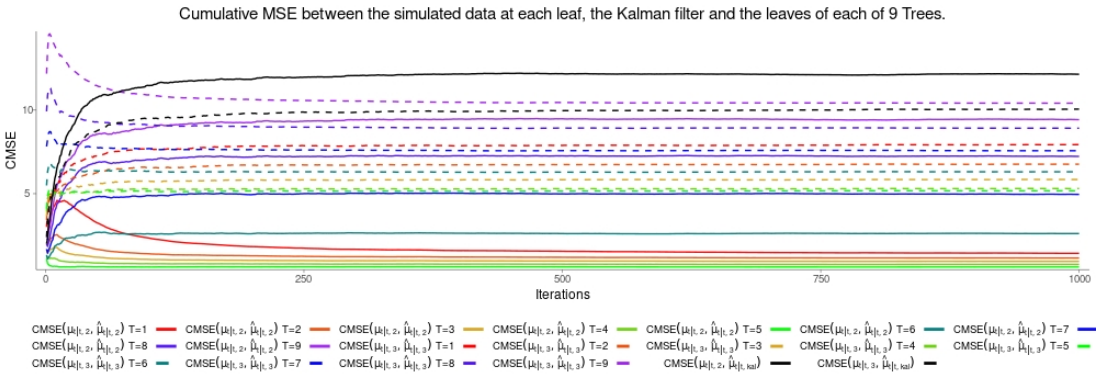


Figure 4.5.3: The cumulative MSE for each leaf of the 9 tree models and the Kalman filter when compared to the data generated at each leaf of T_{sim}.

Kalman filter (because it cannot separate models). The pairwise degradation of the models is not as clear as it was in Figure 4.5.2 but for leaf 2 it can be seen that model performance degrades from T_4 to T_1 then from T_6 to T_9. In other words, at leaf 2 for T_4 to T_1, as the proportion of predictions increases and the proportion of updates decreases the leaf model performance degrades but this is only gradual. However, for trees T_6 to T_9, as the proportion of predictions at leaf 2 decreases and the proportion of updates increases the degradation of the models is more pronounced. This counter intuitive behaviour, more data suggesting a worse performing model, requires further examination which is shown in Figure 4.5.5.

Before turning to Figure 4.5.5, Figure 4.5.4 shows some interesting properties of the leaf filters and relates the intermittent filter, calibration study and tree filter model to each other. The graph consists of 4 panels and only the top two labelled, L2:MSE and L3:MSE, will be considered now with the latter two being considered in Section 4.5.1.2.

Figure 4.5.4 presents the average of the estimates of the filter variance $\hat{\Sigma}_{t|t,b}$, $b = 2, 3$. This is the average loss or MSE of each of the trees at each of the leaves over 1000 iterations and averaged over 100 sample data sets. Also included is the average loss of the Kalman filter, presented as the black line in all of these panels as it was in the preceding graphs. The other colours correspond to the same colours in the graphs already presented so that $T_1 =$ red, $T_9 =$ purple etc.

The first thing to notice is that in every case the MSE reaches a steady state. The authors of the intermittent Kalman filter (IKF) pointed out that the variance of the state is a random variable that depends on both the number of updates and the choice of parameters of the filter. There, a critical rate of update was established for the signal to be considered unbroken or in, other words, the number of updates needed for the Riccati equation to converge to its steady state. Here, based on the calibration study, some parameters were chosen that were guaranteed

to reach steady state even though they may only be updated randomly at an unknown rate. Figure 4.5.4 shows that the MSE at each leaf for every tree does reach this steady state albeit at a rate that is slower than that of the Kalman filter. Thus in the tree filter model, in a similar fashion to the IKF, the variance of the leaf filters (hence of the tree filter) is a stochastic process rather than an asymptotic value.

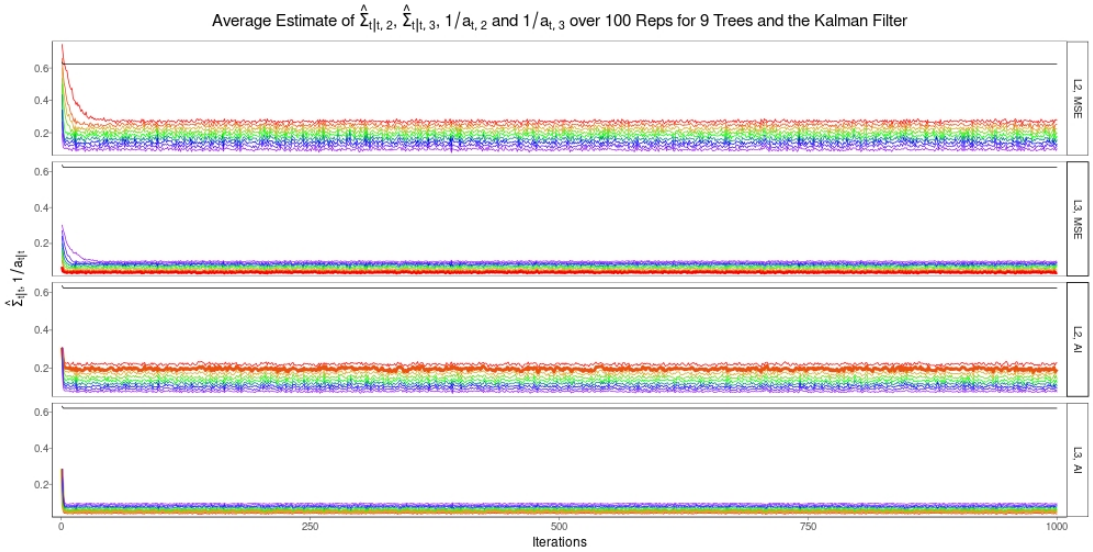


Figure 4.5.4: The variance and inverse of the scale parameter for each leaf of the 9 tree models and the Kalman filter.

The next thing to notice is that the MSE of the leaf filters is lower than the MSE of Kalman filter. This is due to the fact that the tree filter can have different parameters at the leaves while the Kalman filter has only one set of parameters. Next, notice that that for leaf 2 the spread of the processes is larger than for leaf 3 and finally also notice that in leaf 2 the order of the mean values of the stochastic variances proceeds from T_1 to T_9 while this is reversed in leaf 2. At this point it is necessary to look at Figure 4.5.5 which shows the proportions of the visits of the state variance to a discrete collection of continuous intervals.

Figure 4.5.4 is misleading because the values attained by the MSEs are drawn as lines. This was done to more clearly show the sequential relationship between the MSE values but the actual values attained by the variance of the state fall into specific intervals (perhaps even at specific continuous values) even when averaged or smoothed over 100 repeats of the sample data. This is most clearly shown in panel L2:MSE of Figure 4.5.5. The values of the MSE from leaf 2 of each model were placed in to 1000 bins and the bars represent the proportion of instances in each bin. Each bin represents an interval but it can be seen that there are spaces between the bins, smaller spaces between larger collections of spaces. The collections of bins follow a colour ordering from T_9 = purple that has

the greatest number of updates and least number of predictions at leaf 2 to its pairwise opposite $T_1 =$ red with the least number of updates and highest number of predictions at leaf 2. There are some very small bins that count values that occur as the tree attains its steady state and there is some overlap between bins which can be seen most clearly in the central portion of the graph where the MSE for T_5 is spread across 6 collections of bins. The range of values with more than 10 instances in each bin for leaf 2 is $(0.083, 0.283)$ and here the lowest values are associated with T_9 and the highest values with T_1.

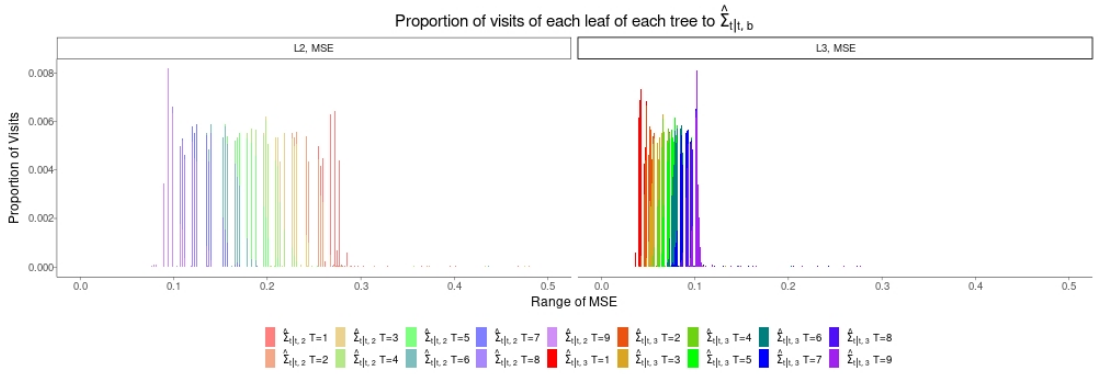


Figure 4.5.5: Proportion of visits of MSE

A similar analysis applies to panel L3:MSE but there the order of the bins is reversed because the proportion of updates and predictions is reversed. In leaf 3 is the range of values is much smaller and the number of bins and collections of bins are fewer although both sets of values were given 1000 bins to be allocated to. The range of values with more than 10 instances in each bin for leaf 3 is $(0.037, 0.106)$ but for leaf 3 the lowest values are associated with T_1 and the highest values with T_9.

Looking back at Figure 4.5.3 and at the solid lines that represent leaf 2, the effect of the lower MSE values for T_9 corresponds with a poor performance of the model. Thus when $T_{9,2}$ gets many updates it performs badly but when $T_{1,2}$ gets few updates it performs quite well. At leaf 3 $T_{1,2}$ gets many updates and T_9 has fewer but the performance of the leaf models are not reversed as one might expect. i.e. $T_{1,3}$ has lower CMSE than $T_{9,3}$ even though the amount of data sent to each leaf is reversed. Then compare these two trees in Figure 4.5.2 and both models seem perform better than all the other models.

This paints a confusing picture because what one would expect is that the more data that is received by a model, the better that model should perform. Except, this depends on the models that are being compared which in this case are the filters at the leaves and their parameters. This suggests a conclusion that the

specification of the leaf parameters, both those that are updated and those that are not updated, may outweigh the probability of an update in determining the performance of the model.

This study now turns to the performance of the tree marginal for the same parameter and tree settings that were used in this part of the study.

4.5.1.2 Tree marginal performance

The calibration study examined the best and worst case scenarios (updates 100% or 0% of the time) for the behaviour of scale parameter $a_{t,b,b}$, its inverse $1/a_{t,b,b}$ and the leaf residual parameter, $d_{t,b,b}$, of the tree marginal. The first part of this part of the study will show the tree and leaf marginals and will then progress to analyse the parameters of the tree marginal. The end result is that choosing the leaf parameters affects the tree marginal which in turn affects the performance of the model. This justifies the calibration study which examined parameters of the tree marginal with the idea of being able to choose (sets of) parameters for initialising a tree filter that did not only prevent the filter from failing but would help inform the prior selection of parameters (and possibly hyperparameters) for the ensemble model to follow which has $6 \times K_T \times \mathcal{F}$ parameters at the leaves of the tree.

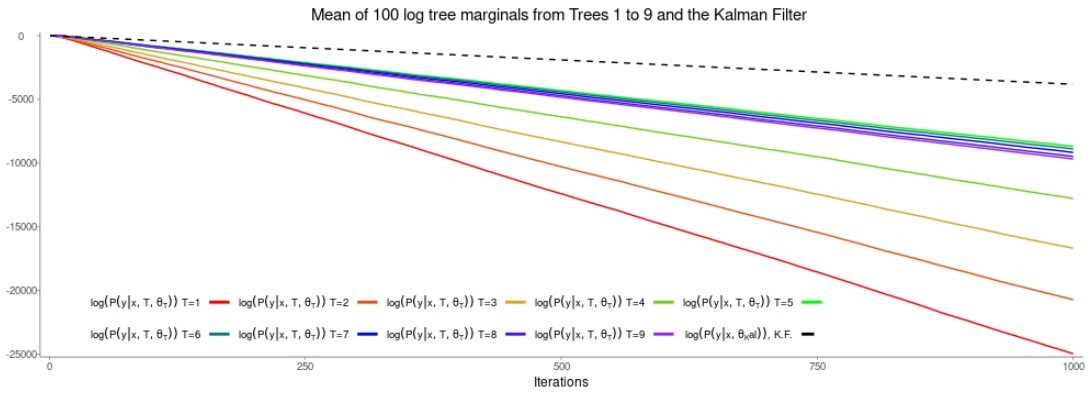


Figure 4.5.6: Tree marginal for each of the experimental trees and for the Kalman filter

Figure 4.5.6 is composed of the mean of the log marginals for the tree filter over 100 samples of length 1000 from the data generator T_{sim}. The linear form of the marginal is a combination of being smoothed over 100 data sets and the linear log form of the marginals themselves shown in Equations (4.32) and (4.33).

The model with the best performance (highest log marginal) is the Kalman filter. Again this is due to the parameters used at the model but crucially in this case because the Kalman filter receives all of the data. The next best performing model is T_5, the model that exactly matches T_{sim}, the data generator, and it receives,

50% of the data each of the leaves, on average. the next 4 best performing models are, in order, T_6, T_7, T_8, T_9 and these receive, respectively, $60\%, 70\%, 80\%$ and 90% of the data at leaf 2. The following 4 models, again in order of degrading performance are T_4, T_3, T_2, T_1 with, respectively, $40\%, 30\%, 20\%$ and 10% of the data being provided at leaf 2.

The effect of the data can best be seen in Figure 4.5.7. Again the dashed lines represent the marginal for leaf 3 and the solid lines the marginal for leaf 2. T_1 has a positive marginal at leaf 2 but this model is receiving only 10% of the data. If this was a truly streaming model with very large number of data points this suggests that with enough data this leaf model would eventually, on average, have probability 1 based on only 10% of the data, clearly nonsense. Similarly, leaf 3 of T_1 gets 90% of the data but fares the the worst out of all the leaf marginals.

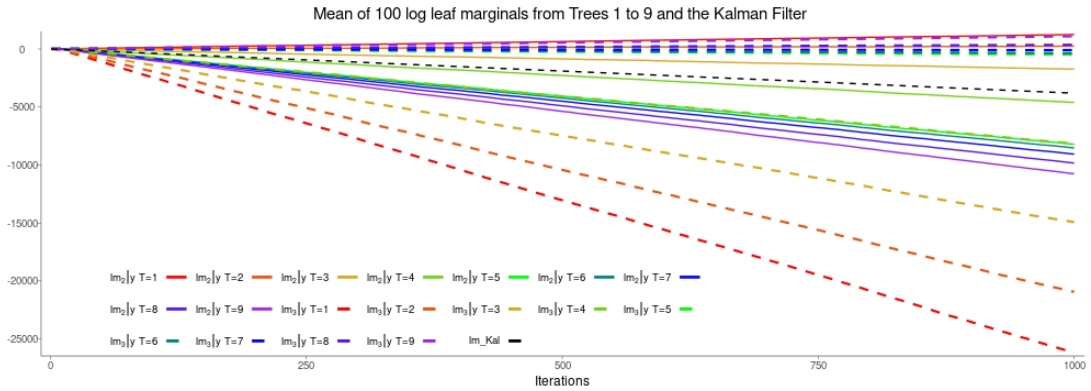


Figure 4.5.7: Leaf marginals for each of the leaves of the 9 experimental trees and for the Kalman filter.

Recall that the log tree marginal (TM) is the sum of the log leaf marginals (LM). For T_5 these values at $t = 1000$ are $LM_{5,2} + LM_{5,3} = TM_5 = -8215.047 - 471.7267 = -8686.774$. Each of these leaves receives 50% of the data but leaf 3 contributes $LM_{5,3} = (TM_5 - LM_{5,2})/TM_5 \approx 95\%$ to the good performance of the tree model while $LM_{5,2}$ contributes only $\approx 5\%$ to the good performance of the model.

Returning briefly to Figure 4.5.4, the lower two panels show that, like the MSE parameter $\hat{\Sigma}_{t|t,b}$ the scale parameter of the marginal, $1/[a]_{t,b,b}$, is a stochastic variable that reaches its steady state, although a but faster than the MSE parameter. The similarity between these two parameters is striking but as Figure 4.5.8 shows they are not the same.

The bar charts in Figure 4.5.8 were constructed in the same manner as those in Figure 4.5.5 in that 1000 bins were allocated to each of the charts but in the case of Figure 4.5.8 there are no spaces between the values that the scale

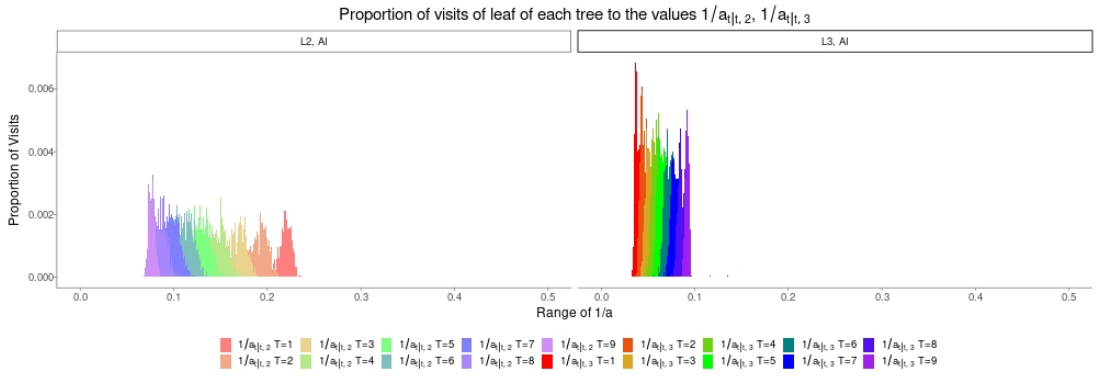


Figure 4.5.8: Distribution of visits of the leaf marginal parameter $1/[a]_{t,b,b}$.

parameter attains (even at 2000 bins no spaces between values were present). The distributions of the parameters have a similar form in that they tend towards a central peak value and appear to be roughly symmetric about that value. In both L2:AI and L3:AI the leaf that is most updated ($T_{9,2}$ in L2:AI, and $T_{1,3}$ L3:AI) have the narrowest distributions and the highest peaks but in the range of values in L2:AI is much broader than in L3:AI.

The final parameter to be considered is the residual parameter of the tree filter, $d_{t,b,b}$. Figure 4.5.9 shows that, on average the Kalman filter explains the data most completely because the residual, on average, is zero. As this is an optimal estimator and receives all of the data this is not surprising.

Figure 4.5.9: The residuals of the leaf marginals, $[d]_{t,b,b}$.

What is surprising is that, again, the model that matches the data generator, T_{sim}, exactly, T_5, seems to contain more residual information than all the other leaves. That is, it has the highest and lowest residual values corresponding to leaf 3 (dashed lines) and leaf 2 (solid lines). The models that seem to perform

the best are T_1, T_9 which reflects their performance in the comparison of tree estimates but not in the leaf or tree marginals where T_1 was the worst performer and T_9 the middle performer.

4.6 Summary and Conclusion

This chapter introduced the tree filter: an on-the-fly conditional model that presents several sets of filters with alternate parametrisations. It was shown that this model reduces uncertainty around the level estimates of an observed stochastic process by adding additional explanatory information at the internal nodes of the tree model. The marginal likelihood of the tree filter was derived and this was compared to the marginal likelihood of the Kalman filter. A calibration study was performed with the intention of finding *a priori* values for setting the multiple leaf filter parameters. Simulation studies were performed with the intention of showing the properties of the fixed two-leaf tree filter under different leaf parametrisations. While the it was shown that the bounds around the level estimates were tighter than those produced by the Kalman filter it was also shown that incorrect parametrisation could lead to spurious results over over a long enough run of data.

In the next chapter tree model learning and adaptation is explored with the aim of removing the dependency of the initial filter parametrisations and internal specifications. The random tree model is then extended to an ensemble of random tree models to produce a tree probability weighted mixture of estimates, predictions and their associated bounds.

5 Tree Filter Ensemble

5.1 Introduction

The previous chapter introduced the tree filter which is an approach to filtering that allows one to specify independent filters for a signal conditional on some deterministic explanatory process x^t. Each leaf of the tree supports a latent process Z_b^t that is intermittently updated by observations $Y_{\eta_{x_t,T}}$ from process Y^t assigned to it by the rules and structure of a fixed regression tree.

This chapter considers how one would search for tree models of the tree filter type in a streaming environment. Section 2.3.2.2 presented a particular method by Chipman et al. (1998) for sampling random trees and then searching for a (smaller) collection of better tree models that can be assessed by an expert for suitability for the experiment under investigation. Alternate tree models are proposed and compared using the integrated likelihood of the current tree and the proposed tree, the prior for each of these models and the proposal distribution within a Markov chain Monte Carlo sampling process. The search for these models is over a fixed sample which limits the number of possible rules at nodes and allows one to ensure that there is enough data in each of the leaf models so that model summaries at these nodes make sense.

The streaming setting presents several challenges to this approach because the sample, while not infinite, is large enough to be considered countably infinite and of a random size. This "axiom" of streaming modelling was discussed, along with others, in Chapter 2, Section 2.5. Not knowing the sample size and also not being able to store any data means that one must be able choose covariates and sample splitting rules (threshold values) for proposed models based on the stream(s) of data on-the-fly. Choosing a window of data creates a dependency on the window size and the data therein. An empirical distribution of the known covariates does not make sense because it assumes that these so-called[1] distributions are either

[1] so-called because if the covariate process is assumed deterministic then they are not random and hence are functions not distributions

independent or, if not independent, are then complex enough to form so that excess memory and processing power is required to maintain these distributions. Not-to-mention that to form or adapt these so-called distributions requires enough data, initially, to get a frequency count and subsequently an excess of data to adjust the frequency count.

This chapter begins by describing how the tree sampling approach of Chipman et al. (1998) is adapted to the streaming setting. This requires a change to some basic assumptions for modelling data which includes an additional parameter which are the bounds of the known covariate processes. The addition of this parameter means that the calculation of the transition proposals needs updating. Some additional proposal moves are suggested and described. This section also includes a description of the Markov chain Monte Carlo approach (MCMC) that will form the basis for proposing and choosing models.

The next part of this chapter will demonstrate the streaming model over three different simulated datasets using a single tree only. It is not expected that a single tree will perform particularly well nor that the MCMC chain will converge to the target posterior but this is a stepping stone between the fixed tree filter and the full mixture of tree models that will be presented and demonstrated in the sequel to the first two parts of this chapter. The particular focus will be on the effect of the filter parameters and how they can be adapted now that the tree model is changing.

This chapter continues by presenting a mixture of tree models over an approximation to the distribution of the trees. This is the third constituent of the model represented by $\mathcal{F}(\cdot)$ in Equation (1.2). A central motivation of this thesis is to demonstrate that MCMC can be used in the streaming setting but with some modifications of the method to take into account that new data is being added to the joint distribution of the observed process. One of these modifications is that multiple chains can be run at the same time and that by forming a mixture over the distribution of the tree models the difference between the current target posterior, $\mathsf{p}\left(T \mid Y^t, Z^t, x^t\right)$, and the target posterior at $t+1$, $\mathsf{p}\left(T \mid Y^{t+1}, Z^{t+1}, x^{t+1}\right)$, will be small enough to be negligible.

The continuation of this part of the chapter will present a comparison between the proposed model and some existing models that are known to converge to the target posterior. To do this will require two comparisons, the first is the case where the data are assumed fixed and the established models are run, according to their recommended specifications, over the fixed data set. The proposed model will be run but, unlike the established models, Bayesian Additive Regression Trees

(BART) (Chipman et al. 2010) and Dynamic Trees (DT) (Taddy et al. 2011), the data will be seen only once. In the second comparison the established models will used in a (pseudo) streaming data setting where new data will be added to the sample at each iteration. In the case of DT their model has already been adapted to the streaming setting by Anagnostopoulos and Gramacy (2013) but in the case of BART, new data is added and for each new data point the algorithm is rerun over the entire data set from start.

5.2 Tree Model Randomization and Sampling

The second constituent of Equation (1.2), represented by $T(\cdot)$ is a regression tree model as typified in Section 2.3.2.2 developed by Chipman et al. (1998) (BCARTMS). The purpose of the fixed tree model, among other things, is to provide a method for structuring and proposing a set of bases for the Kalman filters. However, as mentioned in Section 2.4 single and fixed trees are limited in their usefulness in the streaming setting. The costs of large trees in the streaming case are the increase in complexity of the tree model, the difficulty in filter model parameter specification and the more filters there are the fewer the number of updates there are per filter hence further increasing the dependency of the analysis on the leaf filter parameter specifications.

The consequences of depending on a single main effect and rule are exacerbated in the streaming setting because the nature of the model is changing with time: the relative[2] importance of a covariate or feature may be changing with time and the nature of the phenomenon under study may be changing in time. It is unlikely that a single threshold value at the root node will be able to maintain its explanatory strength for an unknown, but assumed large, duration (number of data points). The need for real-time and any-time inference and prediction means that one cannot stop the analysis, attempt different root or subtree node specifications and then catch-up the analysis.

Thus there is a need, in the statistical streaming setting, for model specification and choice "on-the-fly". One way to achieve this is to propose alternative models, compare these and choose the model that has the greatest odds of being the correct model, at least for some small duration. There is, however, the difficulty of specifying the alternative model because one is only learning about the model and phenomenon at the same time. A way around this is to increase the number of model proposals and choices from some (limited) set of possible models so that the chosen models and their specification can attempt to keep up with the stream

[2]Importance is relative to the chosen covariates, not all possible covariates.

of incoming data.

This section will present a method for proposing new tree models and choosing between an existing tree model and the new proposed model. Proposal models are randomly generated by a version of the prior tree model sampling process of BCARTMS. Tree model samples are compared and chosen via MCMC. The likelihood of each of the models is based on the tree filter marginal of Section 4.3 calculated at each instance t when a new data point y_t is presented to each model thereby providing an instantaneous likelihood ratio over single data point. The jump proposal probability or transition kernel between the current tree and the new tree is based on that of BCARTMS but now including proposals for the additional step changes between models. Marginalising over the leaves of each tree means that a reversible jump approach (Green 1995) between models is avoided.

The subsection that follows will concentrate on describing and demonstrating the tree prior sampling process and the modifications that have been made to this process in light of the streaming setting. This will be followed by description of the MCMC algorithm used in the model and a demonstration of its application. The random tree model will be compared to the fixed tree models of Section 4.3.

5.2.1 Tree Prior

In the streaming setting one might interpret the tree prior as a form of automated prior elicitation based on: some hyperparameters, α and β, that control the tree structure; the chosen covariates and the range of parameters and functions available for choosing and adjusting the parameters of the filters at the leaf models. However, in the streaming setting there is no fixed size dataset thus removing the possibility of a finite number of trees. The number of covariates is finite and, if the leaf parameters are set as was done in the Calibration Study, these too may be finite. Further, for each covariate $x_j \in \mathbb{R}$ there are an infinite number of splitting coordinates $c_j \mid x_j$. Generating a grid-like structure as was done in Denison, Mallick, et al. (1998a) is not impossible but this adds a further specification that may well not suit the analysis for more than a limited duration, not-to-mention the additional overhead of setting up and maintaining the grid. Further, using the methods suggested by Bifet, Gavald, et al. (2018) and described in Section 2.5, to window or "reservoir" explanatory variables would not only create temporal dependencies but could render the inference spurious because BCARTMS rules 3, Section 2.3.2.2 requires a specific number of data points per leaf, and BCARTMS rules 2, Section 2.3.2.2 requires that every leaf can be reached by the data, which would be difficult if not impossible to maintain in the streaming setting.

The complications that result from the streaming setting suggest that the tree prior of BCARTMS requires some tweaking before being used for model selection. The remainder of this section will present modifications to choosing the leaf threshold values in the prior function, Algorithm 2.1, Line 6 and to modifying the tree prior jump steps so that the tree prior sampling process is more suitable for streaming.

5.2.2 Modification of Allowable Coordinates

Calculation of the tree prior is almost exactly the same as in Equation (2.46), that is, a single traversal of the internal and terminal nodes of the tree for each t. The difference between the method of BCARTMS (and Bayesian CART Algorithm (Denison, Mallick, et al. 1998a) (BCART)) and this method come about because the allowable rules in the fixed dataset setting are not suitable in the streaming setting. This section will describe how $\mathsf{p}(\mathit{RULE})$, that is, the allowable rules from Algorithm 2.1, Line 6 are computed in the streaming setting.

A rule for allowability that remains the same as that of BCARTMS (and BCART) is Algorithm 2.1, BCARTMS rules 1 i.e. there must be an available rule, so that if all the possible values have been exhausted for a particular covariate then that covariate is not allowable. This relates primarily to categorical and finite ordinal sets, not used in this document, but possible in general.

The streaming data setting is characterized by fact that the number of data points, N, is considered unknown but large enough so that using a multiset of data to limit the range of threshold choice for each covariate is not possible. Using the reservoir, discretisation or windowing methods such as done by Hoeffding Trees (Bifet and Gavaldà 2009) creates additional temporal dependencies on the stream of data because the covariate values are subject to either distributional assumptions or other summaries that rely on frequencies of feature value occurrence rather than the values themselves Section 2.5.

Ignoring the fact that the range of covariate values can vary over time can also render the model spurious because some combination of threshold values chosen at different times from different reservoirs, windows or distributions might block all possible updates to a leaf at a later time thereby creating a measure over an impossible set. That is, the support of the observation process is spurious because there are latent measures included in the tree that have probability zero for every process $Y^t \in \mathcal{Y}$.

To remedy this situation the method for choosing the allowable set of covariates and thresholds that can be used for proposing new trees must be updated for

the streaming setting. The idea is that the tree prior acts a method to store all the information necessary to choose an appropriate filter. Each internal node, η, for the duration of its existence, must hold all information about the x_j assigned to it. Let η_j be an internal node with assigned rule (x_j, c_j). Then $I(\eta_j)$ is the subset of I_T that contains all rules that relate to covariate x_j in tree T. Each x_j is assumed independent of every other $x_k, k \neq j$ for all t, a standard assumption in regression. Thus for each T, $I(\eta_j) \cup I(\eta_k) = \emptyset$ and by the term "store all information necessary" used earlier in this paragraph it is meant that each $I(\eta_j)$ must be structured in a way that allows for any value c_j of x_j to be used in the tree, but not necessarily at every node, and that every $b \in K_T$ can be reached by any x_t at every t.

All the information that $I(\eta_j)$ needs are the bounds of x_j. This simple addition to the tree model means that each time any η_j splits on some c_j, a subset of x_j is created, where the bounds of the subset are created by the neighbours of η_j in $I(\eta_j)$. Alternatively, if p, the number of covariates, were large enough or if the tree was small enough then one can enforce that each x_j is unique in the tree resulting in an orthogonal basis at each leaf where the origins of the bases at each leaf are the coordinates of the thresholds in $\mathrm{P}_A(b), b \in K_T$. In either case choosing some set or collection of sets that allows/forbids certain rules is simply a case of forming an allowable set of configurations for the MCMC chain and, in essence, is no different to the rules imposed by BCARTMS or BCART other than the fact that an infinite number of covariate values can be included in the model.

In this chapter the rule for allowable values is where the bounds are assumed known for each x_j. It has been assumed without loss of generality that all x_j are bounded in $(0, 1)$ (but this could have been any interval) and assuming that the bounds are known is consistent with the notion that the x_j are known. Bearing in mind that the min and max values of some random sequence are also statistics one could also place a distribution over each covariate dimension with hyperparameters and learn these but this likely adds additional complexity with little possible gain and runs similar risks to the windowing, distributional and discretisation methods of Section 2.5.

Each $I(\eta_j)$ forms an induced subgraph of the tree, H_j, where the nodes are the defined by the values η_c of x_j and the edges are formed by considering the paths of the nodes in base tree graph G. Singleton nodes (nodes in H_j that have no ancestors nor descendants in G) are initially kept but discarded once a randomly chosen new node is added to H_j if the new node does not connect any singleton node to any other nodes in H_j. Randomly choosing some node in H_j by choosing some $\eta \in I_T$ of G and then uniformly choosing some x_j to form $I(\eta_j)$ forms a

clique of H_j if there are ancestors or descendants of η in G and then H_j is formed into a directed graph by having the edges connect the nodes according to node order. As each node has a value c_j associated with it and as it is a directed graph these values form an ordering of the interval $x_j(B) = (0, 1)$. Thus if a threshold is part of a clique then choosing a new threshold means choosing a value that fits into some interval of the existing partition of $x_j(B)$ creating a new subinterval of $x_j(B)$. After a new threshold is added, the allowable set, the set of rules $I(\eta_j)$, is updated so that the ordering of the induced subgraph H_j is maintained for the entirety of the existence of x_j in the tree.

5.2.3 Tree Prior Simulations

The above description of forming an allowable set for choosing covariate rules in the streaming setting highlights an essential difference between the streaming setting and the other tree based approaches to regression: the tree prior must be considered as a nonfinite dimensional stochastic process. In BCARTMS, the prior tree sample was introduced as a "tree-generating stochastic process" (Chipman et al. 1998, p. 938) and the prior distribution over the terminal nodes was produced for several parameter settings. The tree generating density from BCARTMS is:

$$\mathsf{p}\left(\, SPLIT(\eta) \,\right) = \alpha(1 + \delta_\eta)^\beta \tag{5.1}$$

The tree-generating stochastic process in this document is redefined as a reflected random walk with a probabilistic upper boundary. The states of the random walk are the number of leaves, K_T, the lower boundary is always 2 and the upper probabilistic boundary is derived from the fact that Equation (5.1) is a power law so that as δ_η increases, $\mathsf{p}\left(\, SPLIT(\eta) \,\right) \to 0$ and hence so does $\max_{\delta_\eta \to \infty} K_T$. Figures 5.2.1 and 5.2.2 re-present a single example of those provided by BCARTMS.

5.2.4 Target Posterior

In the nonstationary streaming data environment training a tree and then using this tree to make estimations and predictions on the stream of data without modification would be of little value. A single tree model would either have to be very large (such as some of those described in Section 2.5) or would have to be retrained over some batch of data at frequent but probably arbitrary intervals which would interfere with, if not negate, the streaming data approach.

The proposed approach does neither of these. In this case, before the streaming

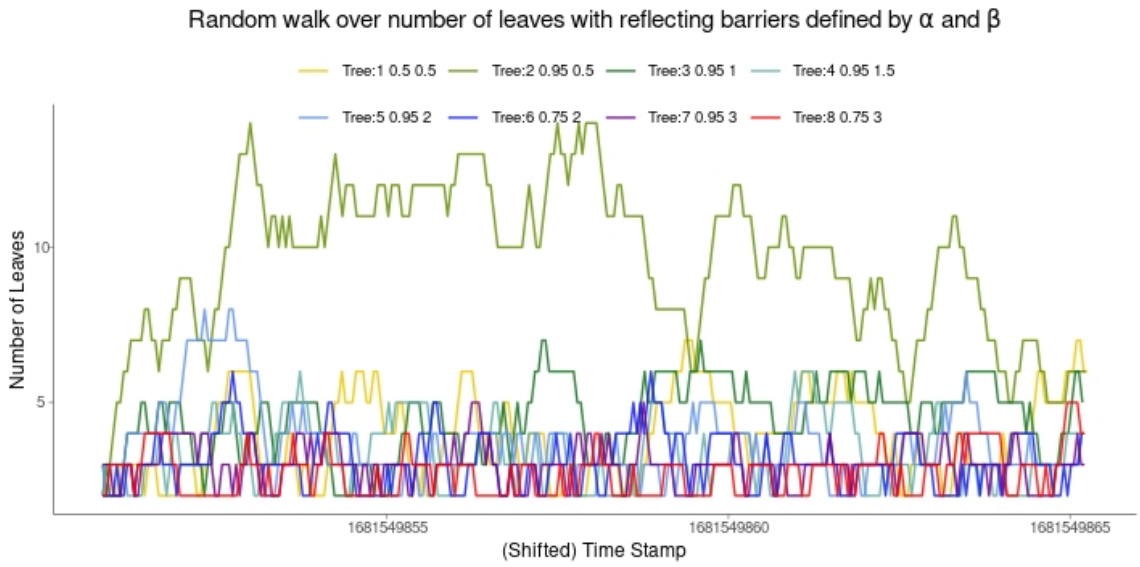


Figure 5.2.1: The tree generating process as a random walk with probabilistic reflective upper barrier.

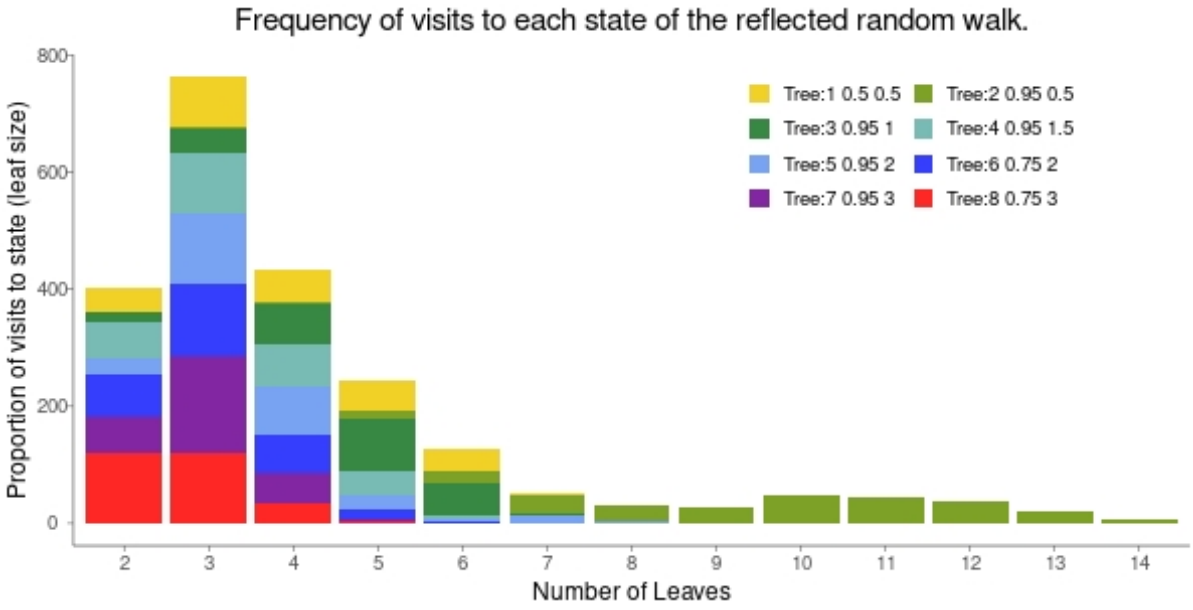


Figure 5.2.2: Frequency of visit to each state conditional on the tree depth and parameters (α, β).

algorithm is begun, the tree is initialised to be a weak learner which is a single internal node and two leaf models. Each leaf model is a Kalman filter and together, the weak learner and the Kalman filters make up the Tree Filter (Section 4.3). However, at each index t the tree is copied and a proposed change to the tree is made so that it is possible for x_t to choose a different filter for y_t to update than in the original tree. The change to the tree constitutes a form of automatic prior model elicitation from the set of covariates, their threshold values and leaf parameters for the set of filter models at the leaves. After some leaf $b \in L_T$ has

Figure 5.2.3: The expected size of a tree conditional on the tree depth and parameters (α, β).

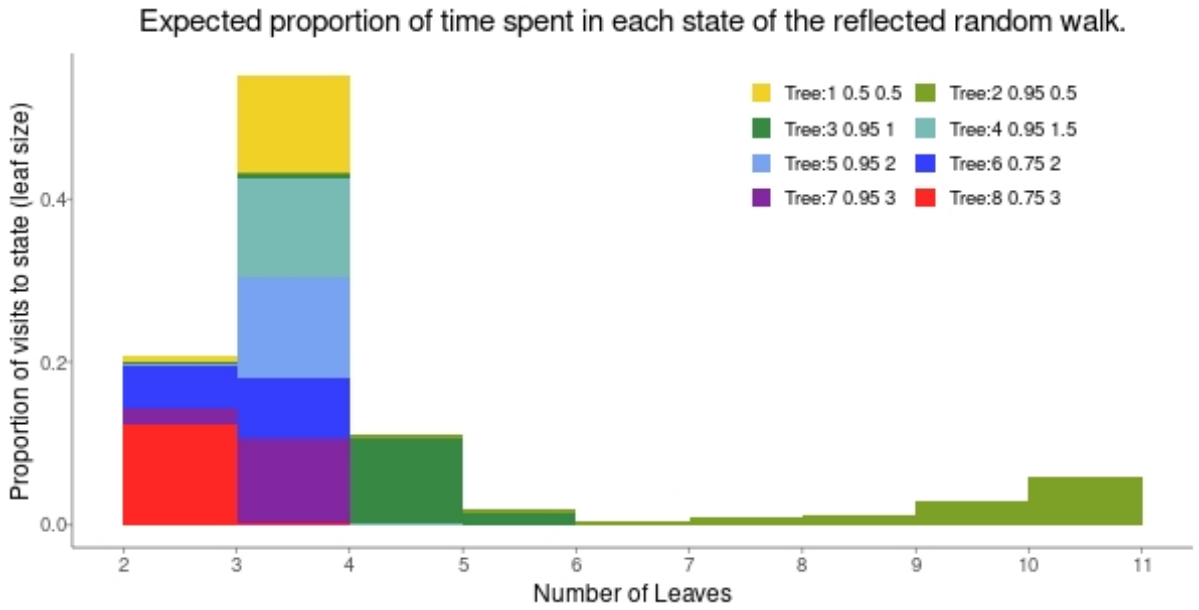


Figure 5.2.4: Expected time spent in each state conditional on the tree depth and parameters (α, β).

been chosen and the filter at that leaf updated, the two trees are compared via an MCMC ratio and the better tree, the one with the greater odds of performing better at this iteration, is accepted and assumed to be able to perform better at the next iteration. This process of state estimation via the tree filter and model prediction via the MCMC process is carried out once for every data point, $d_t = (x_t, y_t)$, presented to it from the stream of data.

The weakness of this approach is immediate from the above description: a single data point is unlikely to present enough evidence for adequate model comparison and for every t, a new data point suggests that, in the MCMC setting, one should restart the sampling process for model selection from $t = 1$, contradicting the previous statements that in the streaming setting one cannot stop, restart

and play catch-up with the streaming data.

A partial solution to the above issue proposed by this modelling framework is that the latent processes $Z_b^t, b \in L_T$ are assumed to be Markovian so that at every t, the available history of the observation process is stored in the latent space. Having multiple latent processes by having multiple leaf models from which to choose at each t provides a larger history of the latent process to support the observation process. Thus the data accrued in the tree marginal likelihood provides a more complete history of the observation process than a single data update.

However, new data is new evidence even if it is a single data point. This means that the joint distribution of the sequence of data

$$\mathsf{P}\,(\,Y_1 \leq y_1, Y_t \leq y_2, \ldots, Y_t \leq y_t\,)$$

is not finite and evidence of the distributional properties of the stochastic process up to index t may not necessarily hold for the joint distribution of the process

$$\mathsf{P}\,(\,Y_1 \leq y_1, Y_2 \leq y_2, \ldots, Y_t \leq y_t, Y_{t+1} \leq y_{t+1}\,)$$

after the new data $Y_{t+1} = y_{t+1}$ has arrived. In the general statistical setting one works with a finite data set and the problem of a changing data set does not generally arise. If a new data point were to be added to the data set then the MCMC process would be rerun over the entire sample. Clearly this cannot be done in the streaming setting if contemporaneous modelling and inference (training and learning) are to be maintained.

The best one can do is to "hope" firstly, that the target distribution, the posterior represented by $\mathsf{p}\,(\,T, Z_t \mid x^t, Y^t, \theta\,)$ at t, is the correct distribution. Secondly that the new target distribution represented by $\mathsf{p}\,(\,T^{true}, Z_{t+1} \mid x_{t+1}, Y_{t+1}, \theta\,)$, where T^{true} is some "true" tree model, is not too far way from the proposed distribution $\mathsf{p}\,(\,T^*, Z_t \mid x_t, Y_t, \theta\,)$ that results from the MCMC model choice process.

From the outset it was clear that a single tree would not resolve this issue. Much in the same way that Chipman et al. (1998) found that restarting the MCMC process provided better coverage of possible modes of the tree posterior space, having an ensemble of trees means that multiple searches are occurring at every instance t to provide better coverage of the tree posterior space and the space of latent processes. Chapter 6 will consider how to bring the posterior at instant $t+1$ closer to the posterior at t. Indeed, MCMC theory (viz. Section 3.3.2) guarantees for a fixed data sample and a uniform transition between model samples, the correct target posterior will be found but this requires a long enough run of the

Markov chain, a luxury that is not available in the streaming setting.

Reproduced here for convenience are the probability model equations:

$$\mathsf{p}\left(T, Y^t, Z^t \mid x^t, \theta_T\right) = \mathsf{p}\left(T \mid \xi_T\right) \mathsf{p}\left(Z_0 \mid T, \psi_T\right) \cdot \prod_{i=1}^{t} \mathsf{p}\left(Y_i \mid T, Z_i, \psi_T\right) \mathsf{p}\left(Z_i \mid Z_{i-1}, T, \psi_T\right) \tag{5.2}$$

$$\mathsf{p}\left(T, Z^t \mid x^t, Y^t t, \theta\right) \propto \mathsf{p}\left(T \mid x^t, \xi_T\right) \mathsf{p}\left(Y^t \mid x^t, Z^t, T, \psi_T\right) \cdot \mathsf{p}\left(Z^t \mid x^t, T, \psi_T\right) \tag{5.3}$$

$$\begin{aligned} \mathsf{p}\left(T, Z^t \mid Y^t, \theta_T, x^t\right) &= \mathsf{p}\left(T \mid Y^t, \theta_T, x^t\right) \mathsf{p}\left(Z^t \mid T, Y^t, \psi_T, x^t\right) \\ &= \underbrace{\mathsf{p}\left(T \mid Y^t, \theta_T, x^t\right)}_{\text{Ⓐ Posterior for tree model}} \times \underbrace{\prod_{b=1}^{K_T} \mathsf{p}\left(Z_b^t \mid T, Y^t, \psi_T, x^t\right)}_{\text{Ⓑ Product of Gaussians from the IKF}} \end{aligned} \tag{5.4}$$

$$\begin{aligned} \mathsf{p}\left(T \mid Y^t, \theta_T, x^t\right) &\propto \mathsf{p}\left(T \mid \Theta_T\right) \mathsf{p}\left(Y^t \mid T, \theta_T, x^t\right) \\ &= \underbrace{\mathsf{p}\left(T \mid \Theta_T\right)}_{\text{Ⓓ Prior for } T} \underbrace{\int \mathsf{p}\left(Y^t \mid T, Z^t, \theta_T, x^t\right) \mathsf{p}\left(Z^t \mid T, \theta_T, x^t\right) \, dZ^t}_{\text{Ⓒ Marginal for } T \text{ over latent process } Z^t} \end{aligned} \tag{5.5}$$

The focus of this section is to describe the MCMC approach for targeting the posterior of the model in Equation (5.2), specifically the left hand side of Equation (5.5) or Ⓐ. The terms in Ⓑ are a product of Gaussian densities where for each density the parameters are either $\hat{\mu}_{t|t,b}, \hat{\Sigma}_{t|t,b}$ if b is the leaf chosen by x_t at index t or $\hat{\mu}_{t|t-1,(b)}, \hat{\Sigma}_{t|t-1,(b)}$ for every leaf not chosen at index t, were (b) means every leaf but b. The random variable $Z_t = z_t$ is sampled from $\mathsf{N}(\hat{\mu}_{t-1|t-1,(b)}, \hat{\Sigma}_{t-1|t-1,(b)})$ at each iteration.

The integral in Ⓒ was the focus of the section titled, The Marginal Likelihood of the Tree, and the formula for calculating the log of this integrated likelihood was outlined there and is presented in full in Appendix A1.1. Ⓓ was the focus of Section 5.2.1, Tree Prior, where the difference in approach to sampling the tree in the streaming setting were presented.

The target posterior of the model, $\mathsf{p}\left(T, Z_t \mid x^t, y^t, \theta_T\right)$, is based on the prior model sample T at instance t and the likelihood of a single data point, y_t marginalised over the latent processes $Z_b^t, b \in L_T$ at the leaves that exist in T at each t. The initial tree at $t = 0$ is a weak learner with a single covariate x_j^R at the root, a uniformly chosen (or fixed) threshold value c_j^R and a linear Kalman

filter with known, but different ψ_b at each of the two leaves. The initial Gaussian state distribution at each leaf is initialised with parameters $\mu_{0,b}, \Sigma_{0,b}$. There is an additional initialisation parameter, $W_{0,b}$, which results from the recursive nature of the algorithm for calculating the tree marginal as shown in Equation (A1.6). This has assumed to be the same as $\Sigma_{0,b}$, both of which are initial guesses. Algorithm 5.1 is an extension of Algorithm 4.1 and will be used as a guide to the random tree model and to the MCMC procedure. The below bold type refers to the relevant line in Algorithm 5.1. These provide some additional detail for a more complete understanding of the algorithm and the streaming training and learning processes.

Initialisation: The tree is copied so that two trees, call them T and T^*, exist. A move (alternatively called step, jump or evolution) is proposed for T^* from the available moves which are change (Algorithm 5.2), grow (Algorithm 5.5) or grow-shift (Algorithm 5.4), named also by numbers 1,2 and 5 respectively. These are the only available moves when a tree is a weak learner. There are no NULL trees nor is there a single filter option at the root. The assumption here is that some explanatory variable x_j will provide more information about the observation process, even if this choice is strongly biased to one or the other leaf.

To generate the leaf parameters the method described in the Calibration Study is used. Some data can be used to get an idea of the standard deviation of the observation process Y^t for setting the range of tree model parameter V_b and likewise it is not unreasonable to use some existing data to set other limits such as the tolerance value $Tol = 0.0001$ that changes the boundary of every covariate variable from $[0, 1]$ to $(0, 1)$. Another way to see Tol is that it sets an absolute minimum size on the tree partition. If any interval created in $I(\eta_j)$ is smaller than Tol the move is automatically rejected. This prevents an endless run of partition refinement.

Other guesses for parameters H_b, F_b and W_b can also be based on expert knowledge or empirical observation. However these parameters' settings are guesses for the tree model as a whole and they are deterministically modified for each leaf as the tree model randomly changes form. Based on Section 5.2.3 one can get a rough idea how deep a tree might be so that setting the scale of parameters such as F_b can be guided.

Tree T^* proposal: (Algorithm 5.1 and line 4) The proposal moves have been described in Algorithms 5.2 to 5.6 of Section 5.2.4.1. In each of these algorithms on the right hand side the probability of that random component has been shown. Baring $\mathsf{p}\,(\,SPLIT(\eta)\,) = \alpha(1+\delta_\eta)^\beta$ all the other densities are discrete

Algorithm 5.1: Streaming algorithm for a random tree model.

Initialise: Set the weak learner with η^R, rule x_j^R, threshold c_j^R uniformly selected and set leaves b_2, b_3 with parameter sets ψ_2, ψ_3. Initialise parameters $\hat{\mu}_{0,b}, \hat{\Sigma}_{0,b}, A_{0,b}, d_{0,b}$ and ℓm_0

Result: At each $t \in 1, \ldots, ,$ $\mu_{t|t,b}$, $\Sigma_{t|t,b}$ and $\log \mathsf{p}\,(\,T, z_t \mid x_t, y_t, \theta_T\,)$

while *!STOP* **do**
- Sample $z_t \sim \mathsf{N}(\hat{\mu}_{t-1|t-1}, \hat{\Sigma}_{t-1|t-1})$
- Copy $T \to T^*$
- Choose move for T^* from Change, Grow, Grow-Shift, Prune, Prune-Shift
  - ▷ `If` $K_T = 2$ `only Change, Grow and Grow-Shift apply`
- Generate $Q(T) = \mathsf{p}\,(\,T,\ T^*\,)$
- Generate $Q(T^*) = \mathsf{p}\,(\,T^*,\ T\,)$
- **foreach** $b \in L_T$ & $b \in L_{T^*}$ **do**
  - Predict $\mu_{t|t-1,b}$
- **end**
- Wait for x_t
- Using x_t, choose the filter at leaf b from T and b^* from T^*
- Wait for y_t
- Using y_t, update the Kalman Filter at leaves b, b^*
- Output $\mu_{t|t,b}$, $\Sigma_{t|t,b}$ from T only.
- **foreach** $b \in L_T$ & $b \in L_{T^*}$ **do**
  - Update $A^*_{t-1,b} \to A_{t-1,b}$
  - Update $A_{t,b}$ according to whether b was chosen by x_t or not
  - Update $d_{t,b}$ according to whether b was chosen by x_t or not
  - Calculate $\ell m_{t,b}$
  - Get $\log \mathsf{p}\left(\, Z_{t,b} = z_{t,b} \mid \hat{\mu}_{t|t,b}, \hat{\Sigma}_{t|t,b}, \psi_b, \cdot \,\right)$
- **end**
- Get $\log \mathsf{p}\,(\,T \mid \cdot\,)$ and $\log \mathsf{p}\,(\,T^* \mid \cdot\,)$
- Get $\log Q = \log \mathsf{p}\,(\,T^*,\ T\,) - \log \mathsf{p}\,(\,T,\ T^*\,)$
- Calculate $\log q^*(T) = \log \mathsf{p}\,(\,T \mid \cdot\,) + \sum_{b=1}^{K_T} \ell m_{t,b} + \log \mathsf{p}\,(\,Z_{t,b} \mid \cdot\,)$ and $\log q^*(T^*) = \log \mathsf{p}\,(\,T^* \mid \cdot\,) + \sum_{b=1}^{K_{T^*}} \ell m_{t,b} + \log \mathsf{p}\,(\,Z_b \mid \cdot\,)$
- Calculate $\log \alpha_t\{T,\ T^*\} = \min\{\log q^*(T^*) - \log q^*(T) + \log Q,\ 0\}$
- Draw $\log u_t \sim \log \mathsf{U}(0, 1)$
- **if** $\log u_t \leq \log \alpha_t\{T,\ T^*\}$ **then**
  - Set $T \leftarrow T^*$
- **else**
  - Delete T^*
- **end**

end

uniform densities. Not shown in those algorithms is the calculation of the threshold probability. This involves considering $\mathsf{P}\,(\,LB_j \leq c_j \mid I(\eta_j)\,)$ on the left hand side and $\mathsf{P}\,(\,c_j \leq UB_j \mid I(\eta_j)\,)$ on the right hand side. Once $I(\eta_j)$ been chosen by uniformly choosing x_j, each of the previous probabilities is the conditional probability that the interval to the left (or right) is of the size $c_j - LB_j$ or $UB_j - c_j$

respectively. Setting Tol will affect this measurement so ideally Tol should be set much lower than the expected interval size.

The resulting transition ratios of Algorithm 5.1 and line 5 are:

$$Q(T) = \frac{\mathsf{p}\,(\,T^*,\,T\,)}{\mathsf{p}\,(\,T,\,T^*\,)} = \frac{\mathsf{p}\,(\,T \mid T^*\,)}{\mathsf{p}\,(\,T^* \mid T\,)} \tag{5.6}$$

where, for the change move,

$$\begin{aligned}\mathsf{p}\,(\,T^* \mid T\,) &= \frac{c_j - LB_j}{UB_j - LB_j} \times \frac{1}{ALLOWABLE\ p} \\ \mathsf{p}\,(\,T \mid T^*\,) &= \frac{c_{(j)} - LB_{(j)}}{UB_{(j)} - LB_{(j)}} \times \frac{1}{ALLOWABLE\ p}\end{aligned} \tag{5.7}$$

and for the grow and prune moves,

$$\begin{aligned}\mathsf{p}\,(\,T^* \mid T\,) &= \frac{c_j - LB_j}{UB_j - LB_j} \times \frac{1}{ALLOWABLE\ p} \times \frac{1}{K_T} \times \alpha(1+\delta_{b_{RN}})^{\beta} \\ \mathsf{p}\,(\,T \mid T^*\,) &= \frac{1}{2 \times |pairs|} \times 1 - \alpha(1+\delta_{b_{RN}})^{\beta}\end{aligned} \tag{5.8}$$

and the grow-shift and prune-shift moves,

$$\begin{aligned}\mathsf{p}\,(\,T^* \mid T\,) &= \frac{c_j - LB_j}{UB_j - LB_j} \times \frac{1}{|I_T|} \times \frac{1}{ALLOWABLE\ p} \times \alpha(1+\delta_{b_{RN}})^{\beta} \\ \mathsf{p}\,(\,T \mid T^*\,) &= \frac{1}{K_T} \times 1 - \alpha(1+\delta_{b_{RN}})^{\beta}\end{aligned} \tag{5.9}$$

where $ALLOWABLE$ means the number of covariates that are currently allowable for that particular tree based on the covariates and bounds of the part of the tree ancestral to the currently chosen node.

Note that these ratios should be reversed for the Prune and Prune-Shift moves. In the Change move subscript (j) means the covariate at the node before the change move was proposed. The difference between the proposal moves of BCARTMS and this approach are now clear because in the former approach one compares a discrete uniform chance of conditionally choosing a threshold value and in this case one is comparing the ratio of interval sizes produced by a uniformly chosen threshold value and the boundary from which that threshold is drawn.

Filter Predictions: (Algorithm 5.1 and line 8) At each of the leaves b in both T and T^* a one-step-ahead prediction is made using:

$$\begin{aligned}\hat{\mu}_{t|t-1,b} &= F_{t,b}\hat{\mu}_{t-1|t-1,b} \\ \Sigma_{t|t-1} &= F_{t,b}\Sigma_{t-1|t-1}F_{t,b}^T + W_{t,b}\end{aligned}$$

This prediction is not generally output and is currently only used to advance the state processes for each of the $Z(t, b)$. However, should one assume that x_t and y_t have some difference in arrival time for the same index t[3] one might want to make (one or several) predictions between each observation. This has not been done in this paper but is well within the capabilities of, and one of the reasons for choosing, the Kalman Filter.

Filter Estimate: (Algorithm 5.1 and line 13) A data point, $d_t = (x_t, y_t)$ arrives via a communication channel in an (again assumed) formatted state so that $x_t = (x_1, \ldots, x_p)$, $x_j \in (0, 1)$[4] and $y_t \in \mathbb{R}^n$. The explanatory vector x_t is then used to choose a leaf in both T and T^* and y_t is used to update the filter at that leaf. At this point the estimates of the state process:

$$\hat{\mu}_{t|t,b} = \mu_{t|t-1,b} + \mathrm{I_{t,b}}\, K_{t,b} e_{t,b}; \tag{5.10}$$

$$\hat{\Sigma}_{t|t,b} = R_{t,b} - \mathrm{I_{t,b}}\, K_{t,b} H_{t,b} R_{t,b} \tag{5.11}$$

are output.

Tree Marginal: (Algorithm 5.1 and line 15) The updating of the tree marginal is exactly the same as done in Section 4.3 only now at each t there may be different numbers of leaves. The marginal of each leaf keeps a recursive record of the data that is assigned to it for as long as that leaf exists. Thus, when a leaf is removed or added the information stored in that leaf must be maintained within the tree. In the case of the change move nothing, with respect to the tree structure, changes. In the case of the prune move if two nodes are removed the log marginal of each leaf is added and divided by two. For the grow move half the log marginal is allocated to each new leaf. This simple way of dividing and collecting the information stored in the leaf marginals ignores whether one leaf was updated more often than another but is based on the fact that when two leaves are added one does not know which leaf will be updated more often in the future.

In the case of the grow-shift and prune-shift moves sharing stored information is not quite so strait forward because there may not be parents and children from

[3]Each iteration of the algorithm has been indexed by t and it has also been assumed that each iteration of the algorithm shares a single $d_t = (x_t, y_t)$ but this not generally the case. Each index of the algorithm could be indexed separately by the arrival time of x_t and/or the arrival time of y_t. These two streams of data could, and probably would, have different but not dissimilar arrival times but this leads to a level of complexity that has been avoided here and by others (Bifet, Gavald, et al. 2018). It is assumed that for every index t, the x_t at the receiver (socket, preprocessor output connection, pipe etc.) explains the very next or simultaneous y_t at the receiver, an assumption that is dubious hence any causal inference should be avoided.

[4]and is correctly labelled so that each x_{tj} actually is from source $\mathcal{X}_j$

which to collect the information. Thus the approach here is to initialise the log marginal a one does in the **Initialise** step.

This is also true for the parameters $\hat{\mu}_{0,b}$ and $\hat{\Sigma}_{b,0}$. That is, in the case of prune and grow, $\hat{\mu}_{t|t,b}$ is sampled from both children and averaged or from the parent and $\hat{\Sigma}_{b,0}$ is initialised to $W_{b,0}$. For grow-shift both $\hat{\mu}_{t|t,b} = \hat{\mu}_{0,b}$ and $\hat{\Sigma}_{b,0} = W_{b,0}$.

A final item to note is the modification of the parameters $\psi_b = \{H_b, V_b, F_b, W_b\}$. Each of these can be modified by some deterministic function as the tree changes shape. While a seemingly obvious solution to this is to sample ψ_b for each of the leaves that need modification, in a non-stationary environment the hyperparameters for the distribution from which any of these parameters might be sampled would also need to be learnt "on-the-fly", not-mention the lack of identifiability of the parameters of the Kalman Filter.

Sampling V_b or W_b only seems an intuitive option but this does not alleviate the need for hyperparameter learning nor does it necessarily make sense when considering the tree structure as method of increasing homogeneity of the model space. That is, as mentioned in Section 2.3.2.1, one of the reasons for choosing tree models is that the partition of the model space better explains the data by making the data more homogenous within the partition. Thus one might expect in the random tree modelling case that as the tree changes the signal-to-noise ratio $r = W/V$ would increase for a given W (because V, the measurement noise, is getting smaller). But randomly drawing V does not guarantee this without some form of immediate adjustment to the distribution from which V might be drawn as the tree changes. This is one of the reasons for performing the Calibration Study of Section 4.4.

Indeed, as mentioned in that section, one of the intentions behind choosing a tree structure is to provide a set of alternate models and then, by using a set of calibrated parameters, one can view possible filter parameters more along the lines of explanatory variables rather than as random missing data as one usually does for parameters. Thus, modifying filter parameters becomes a case of explaining the models via adding and removing known variables as suggested by G. Box and Luceño (1997).

The Calibration Study has presented a solution to choosing prior sets of parameters for the filters. In the simulation studies that follow the details for each experiment will be explained. The general approach assumed in this document is based on the results of the calibration study and that showed that the two main factors that influenced the leaf models were F_b and H_b/V_b. For H_b/V_b, based on the discount factor approach of Harrison and West (1999), it has been assumed,

as described, that the intention of the tree model is explain the signal better by reducing the measurement noise. Thus, as the tree depth increases, V_b decreases and $r = W_b/V_b$ increases for a constant W_b. Similarly, as V_b decreases, H_b/V_b increases for a constant H_b. Thus the tree marginal does not lose its explanatory power as the tree model increases because if $H_b/V_b \to 0$ then, from the calibration study it can be seen that the effect of the marginal (the effect of the data in the tree) also tends to zero (Table 4.4.10).

The parameter F_b has possibly the largest effect on the streaming tree model behaviour and performance. The filters must either be updated with the probability according to the critical parameter as defined by Sinopoli et al. (2004) or this F_b must be limited to $(-1, 1)$ and preferably substantially below each of the limits of that interval. Guaranteeing that the updates conform to Sinopoli et al. (2004) is not the best option because even if the tree were limited to 2 leaves, random choices of the pair (x_j, c_j) could temporarily bias the update of a filter so that $A_t^{-1} \to 0$ and the algorithm crashes because of a divide-by-zero error. Thus best option found so far is to limit F_b to be between $(-1, 1)$ (and actually closer to $(-0.75, 0.75)$).

An example of a family of functions that have been used to deterministically change the values of V_b and F_b are:

$$\begin{aligned} F_b &= \max\{\text{sgn}F_b^* \delta_b^{-s}, \min_F\} & F_b^{-1} &= \min\{\text{sgn}F_b^* \delta_b^{s}, \max_F\} \\ V_b &= \max\{F_b^* \delta_b^{-s}, \min_V\} & V_b^{-1} &= \min\{F_b^* \delta_b^{s}, \max_V\} \end{aligned} \tag{5.12}$$

where s is predetermined scale at which the parameter values change, $\min_F, \min_V$ are the smallest values that are allowed for these parameters and $\max_F, \max_V$ are the largest parameters allowed. The sign of F_b is allowed to randomly change and V_b is always positive.

This above approach is only one possible way of doing things that seemed intuitive and practical after many tests on different approaches. It is important different models are proposed at each leaf, even if only slightly different else there seems little reason for specifying some conditional rule. The section that follows provides more detail on each of the proposal moves.

5.2.4.1 Modification of Tree Proposal Moves

Proposing new trees via the tree model and structure changes proposed by Chipman et al. (1998) is not ideal for the streaming setting because stopping and restarting the chain, due to the chain getting stuck in local modes, means there

will be pause between the current inference and the past inference so that information is lost. In other words, from a streaming point of view, stopping and starting a chain is equivalent to restarting the analysis. The streaming setting necessitates long runs of the Markov chains, which in BCARTMS was found not to be useful. Others such as Wu et al. (2007) and Pratola (2016) have suggested other modification steps but these are too complex for the streaming setting. Thus some alternative simple modification steps are proposed.

Firstly, following Pratola (2016), the swap move has been abandoned. The second change is based on the realization that the only time a tree can be pruned is when there are a pair of siblings available. For example in Figures 5.2.5a and 5.2.5b,

(a) A tree with only left internal nodes.

(b) A tree with only right internal nodes.

Figure 5.2.5

the chance of pruning either of the trees is 2/5 but the chance of growing either of them is 5/5 assuming that both grow and prune moves are equally likely to be chosen, which is not true for 2 leaf trees. To remedy this the prune-shift move was designed to prune the tree at a single leaf node and, because a reversible move for this is required, an additional move called the grow-shift move was included.

The change move forms a part of the all the other moves in some way. For the grow and new grow-shift moves because a new internal node is added this change move must be performed to some degree. For the prune and prune shift moves, because an internal node is removed $I(\eta_j)$ (specifically the boundaries of the nodes that have covariate x_j) must be updated although no new covariate or threshold value is added for these moves.

The calculation of the transition probability is written in the comments of this algorithm. Unlike in BCARTMS and BCART the choice of c_j depends on the bounds of the covariate at each node. While the tree is a random object the

bounds are considered deterministic and so are the bounds of each of the nodes in $I(\eta_j)$ much as the values x_j are considered deterministic: they are part of the history of the tree.

The set of *ALLOWABLE* covariates (covariates from *ALLOWABLE* nodes) changes as the structure of the tree changes and it is possible that at some index t some x_j is not in *ALLOWABLE* but at some other instant it is. Thus if x_j is not *ALLOWABLE* it is suspended from the set of allowable covariates but it is allowed back for the next transition. This too is considered in the same light as covariates are in general so these changes to *ALLOWABLE* do not form part of the transition probability. If $NULL$ is returned then the current tree is returned and the algorithm continues.

Algorithm 5.2: Choosing and modifying an internal node.

Result: Change: Tree T is modified with a new covariate x_j at a random internal node based on $I(\eta_j)$ or NULL is returned.

```
1  Uniformly choose η_RN from the internal nodes I_T                ▷ p( η_RN | T )
2  while TRUE do
3  |  Choose U(ALLOWABLE x_j)                                      ▷ p( x_j | ALLOWABLE,T )
4  |  if x_j ∈ALLOWABLE then
5  |  |  Get H_j, the induced subgraph of x_j, hence I(η_j)
6  |  |  Get new node boundary values (LB_j, UB_j) from I(η_j)
7  |  |  if (LB_j, UB_j) ≠ ∅ then
8  |  |  |  Choose new threshold c_j                              ▷ p( c_j | I(η_j), x_j, η_RN, T )
9  |  |  |  Reset boundaries of I(η_j)
10 |  |  |  break;
11 |  |  end
12 |  |  else
13 |  |  |  Suspend x_j from ALLOWABLE
14 |  |  |  if ALLOWABLE = ∅ then
15 |  |  |  |  return (NULL)
16 |  |  |  end
17 |  |  end
18 |  end
19 |  else
20 |  |  Suspend x_j from ALLOWABLE
21 |  |  if ALLOWABLE = ∅ then
22 |  |  |  return (NULL)
23 |  |  end
24 |  end
25 end
```

The algorithm for pruning and shifting is given in Figure 5.2.6a. The reverse move, grow-shift are shown in Algorithm 5.4 and Figure 5.2.6b. Figures 5.2.6a

and 5.2.6b provide a graphic demonstration of the above moves. In Figure 5.2.6a in the top graphic the chosen node, b_{RN}, is either $b = 4$ or $b = 5$ in the left tree and the result is the tree on the right. In the bottom graphic $b_{RN} = 6$ on the left and the result is the tree on the right. Similarly, in Figure 5.2.6b, the current tree is on the left and in the top graphic $\eta_{RN} = 2$ and the result is on the right. In the bottom graphic $\eta_{RN} = 2$ again and the result is on the right. Looking more closely at the top graphic it looks exactly the same as normal grow move and this is almost true except that rather than choosing a leaf to be a new internal node the internal structure of the tree is shifted. The bottom graphic shows this more clearly. These prune-shift and grow-shift moves are reminiscent of the "radical restructure" of Wu et al. (2007) except that the restructuring is not too radical because of the complexity of changing too much in the tree. For completeness the algorithms for the grow and prune moves follow. These are not fundamentally different from the original moves other than the fact that the set $I(\eta_j)$ and the leaf parameters must updated in each move.

Algorithm 5.3: Removing an internal node and a single leaf. Shifting internal subtree up one level.

Result: Prune-Shift: Tree T with a single internal and leaf node removed.

1 Choose b_{RN} from leaf nodes with no leaf sibling ▷ $\mathsf{p}\left(b_{RN} \mid T\right)$

2 Remove b_{RN} and $\mathrm{P}(b_{RN})$ from tree ▷ $1 - \mathsf{p}\left(SPLIT_{(b_{RN})}\right)$

3 Renumber subtree of $\mathrm{P}(b_{RN})$ starting with new parent which is $\tilde{\mathrm{P}}(b_{RN}) = \mathrm{Sib}(b_{RN})$

▷ b_{RN} `was single leaf so` $\tilde{\mathrm{P}}(RN)$ `is internal node`

4 Reset boundaries of $I(\eta_{\tilde{\mathrm{P}}(b_{RN})})$

▷ $I(\eta_{\tilde{\mathrm{P}}(b_{RN})})$ `is set of` x_j `induced by the rule at` $\tilde{\mathrm{P}}(b_{RN})$ `and` H_j

5 **for** $b \in Subtree(\tilde{\mathrm{P}}(b_{RN}))$ **do**

6 | Renumber leaf of tree based on internal node structure of subtree

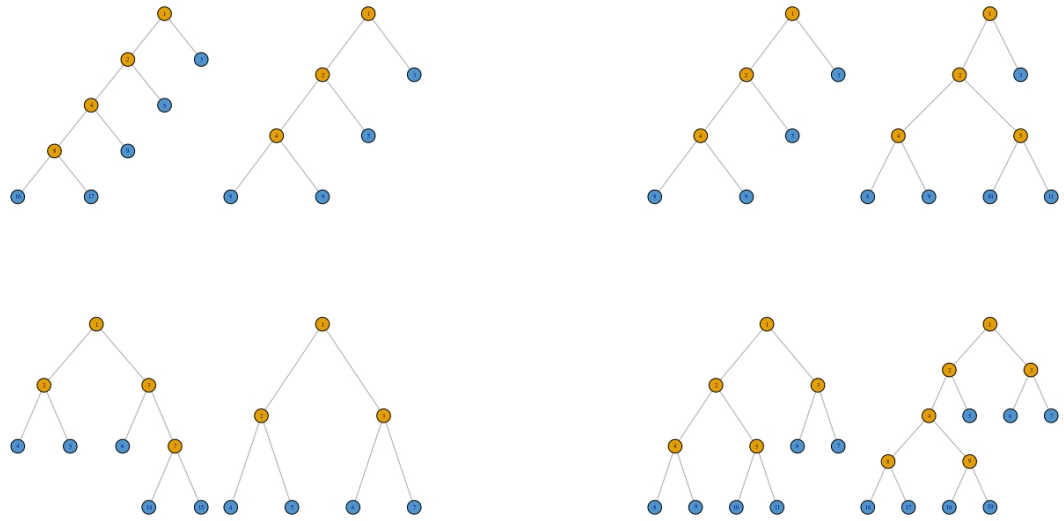
7 | Modify the ψ_b based on change in internal tree structure

8 **end**

Algorithm 5.4: Creating a new internal node and a single new leaf. Shifting internal subtree down one level.

Result: Grow-Shift: Tree T with a single new leaf node and a single new internal node.

Choose η_{RN} from any internal node ▷ $\mathsf{p}(\eta_{RN} \mid T)$
Choose to add either $\mathrm{C}_l(\eta)$ or $\mathrm{C}_r(\eta)$ ▷ $\mathsf{p}(Left) = 0.5$
if *left* **then**
    Add $\tilde{\mathrm{C}}_l(\eta)$
    Add $\tilde{\mathrm{C}}_r(b)$ a new leaf node ▷ $\mathsf{p}(SPLIT(\tilde{\mathrm{C}}_l(\eta)))$
    Shift $\mathrm{C}_l(\eta)$ (the current left child) down one tree level
    Update leaf nodes in subtree with root at $\tilde{\mathrm{C}}_l(\eta)$
    for $b \in Subtree(\tilde{\mathrm{C}}_l(\eta))$ **do**
        Renumber leaf of tree based on internal node structure of subtree
        Modify the ψ_b based on change in internal tree structure
    end
end
else
    Add $\tilde{\mathrm{C}}_r(\eta)$
    Add $\tilde{\mathrm{C}}_l(b)$ a new leaf node
    Shift $\mathrm{C}_r(\eta)$ (the current left child) down one tree level
    Update leaf nodes in subtree with root at $\tilde{\mathrm{C}}_r(\eta)$
    for $b \in Subtree(\tilde{\mathrm{C}}_l(\eta))$ **do**
        Renumber leaf of tree based on internal node structure of subtree
        Modify the ψ_b based on change in internal tree structure
    end
end
Update internal nodes by performing change move on $\tilde{\mathrm{C}}_l(\eta)$ or $\tilde{\mathrm{C}}_r(\eta)$ depending on which side was chosen
▷ $\mathsf{p}(c_j \mid I(\eta_j), x_j, \eta_{RN}, T)\,\mathsf{p}(x_j \mid ALLOWABLE,T)$
if $change \neq NULL$ **then**
    return T
end
else
    return NULL
end



(a) Pruning and shifting nodes from a single leaf.

(b) Growing, shifting nodes and adding a leaf from an internal node.

Figure 5.2.6

Algorithm 5.5: Adding two new leaf nodes and a single new internal node.

Result: Grow: Tree T with two new leaf nodes and a single new internal node.

Uniformly choose b_{RN} from any of the leaf nodes ▷ $\mathsf{P}(b_{RN} \mid T)$

Add $\mathrm{C}_l(b_{RN})$ $\mathrm{C}_r(b_{RN})$ as new leaf nodes ▷ $\mathsf{p}(SPLIT(\tilde{\mathrm{C}}_l(b_{RN})))^2$

Modify $\psi_{\mathrm{C}_l(b_{RN})}$ and $\psi_{\mathrm{C}_r(b_{RN})}$ based on $\psi_{b_{RN}}$

Delete b_{RN} from L_T

Add η_{RN} to I_T; Update η_{RN} by performing change move. ▷ $\mathsf{p}(c_j \mid I(\eta_j), x_j, \eta_{RN}, T)\,\mathsf{p}(x_j \mid ALLOWABLE,T)$

if $change \neq NULL$ **then**

  return T

end

else

  return NULL

end

Algorithm 5.6: Removing two leaves and their parent leaf and generating a replacement leaf.

Result: Prune: Tree T with two fewer leaves and new leaf node.

Choose b_{RN} from leaf nodes with two siblings ▷ $\mathsf{p}(b_{RN} \mid T)$

Get $\mathrm{P}(b_{RN})$ and delete it from I_T Add $\mathrm{P}(b_{RN})$ to L_T

Modify $\psi_{\mathrm{P}(b_{RN})}$ based on b_{RN}

Delete b_{RN} and $Sib(b_{RN})$ from L_T ▷ $1 - \mathsf{p}(SPLIT(\mathrm{P}(b_{RN})))$

5.3 Random Tree Demos and Simulations

This section aims to explore the performance of a single tree model that is evolving to keep pace with a stream of data. It is not expected that the a single tree will perform particularly well in terms of either estimation of the state or in finding the target distribution of the tree. The evolution of a single tree model is a stepping stone between the fixed tree model and the ensemble of tree models and the main intention of this section is to explore the effects of some of the model parameters now that the tree filter is training, learning, predicting and adapting to new data on-the-fly.

The simulation studies and demonstrations are performed over three models called the 2 Leaf Model, the CGM Model[5] and the the Friedman Model. Each of these are briefly described below.

Theory, the calibration study of Section 4.3 and the simulation studies of the same chapter have shown the importance of having the parameter $F_b \in (-1, 1)$ but ideally having the upper and lower limits of this parameter closer to -0.85 and 0.85 respectively. What has not been shown is how this parameter might change as the tree model changes. Similarly, other parameters W_b, H_b were shown relative to V_b for fixed F_b but what has not been shown is how any of these parameters, V_b, W_b, H_b, change as the tree model changes. That these parameters need to change in some way is assumed essential because the point of the tree model is to propose different, locally interactive[6] models that include different parameters and not only a different tree and rule structure so that the model can learn and adapt to the stream of data.

The final aspect of the tree model that also needs exploring are the tree parameters $(\alpha, \beta) \subset \xi_T$. Section 5.2.3 showed how the prior sampling process could be interpreted in the streaming setting as a reflected random walk with a probabilistic upper boundary. Rechecking this processes with the tree marginal and its parameters is necessary because the accumulation of data and its proportional effect relative to the tree prior in the streaming setting must taken into account.

The same caution and caveats regarding the choice of number of iterations and assumptions regarding future model behaviour described in Section 4.5 apply here. These are more applicable in the case of random model selection where the model object itself (the tree) can be affected by computational and other errors that result from the recursive nature of the algorithm.

[5] Inspired by the 5 leaf tree example in Chipman et al. 1998 but modified for the streaming setting.

[6] interactive in the sense of interaction of effects with the main effect at the root node

5.3.1 Modifying leaf parameters

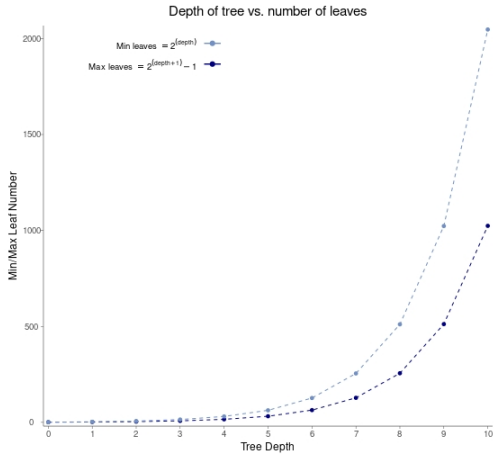
The calibration study of Section 4.4 presented a method of designing trees with different leaf model parameters that avoided estimation and sampling. This study was aimed at showing that one can, prior to beginning the streaming exercise, set the leaf parameters so that the model, firstly, will not fail due incorrect specifications of parameters that lead to, for example zero determinants or infinitely (very) large variances. Secondly, one can have an idea, at least at the leaf level, of the performance of the model. Thirdly, so that one can simplify the state estimation process to reduce memory and complexity of the algorithm for better streaming performance by treating leaf filter parameters in a manner akin to explanatory, rather than missing, data. Finally, for subsequent analysis, a base level of parameter performance is achieved for a fixed tree model so that studies for the evolving tree model can be better understood.

For the evolving tree model, the simulation study in Section 5.2.3 showed that for some parameter settings one might expect a large number of leaves in the tree model. Preliminary analysis has shown that the tree prior is not the sole determinant of tree size. Even if the model was set with $(\alpha = 0.75, \beta = 3)$, the most restrictive of the parameters used in that study, trees larger than 2 leaves are quite likely on average. Thus a method for adapting any of the parameters as the tree model grows is necessary to provide local (to the tree), alternative and interactive (interacting with the covariate at the root node) models.

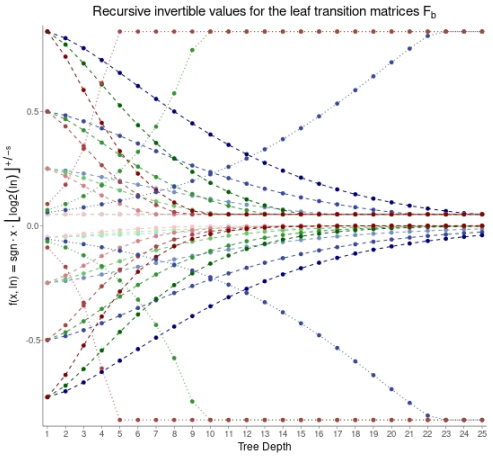
Deterministically adapting parameters is also necessary for the ensemble to explore the full space of the model. Unlike in BART where there is only a single variance parameter or in BCARTMS and Dynamic Trees where the variance parameter is marginalised out, each filter has up to 6 different parameters. In the streaming setting marginalisation over Z^t and T (Equation (4.18)) was relatively straightforward because of properties of the Gaussian distribution. Additional marginalization over any of the variance parameters as was done in Harrison and West (1999) would have produced a student-t distribution which is not amenable to easy recursive calculation.

The idea for deterministic parameter modification came from Harrison and West (1999) where there they provide a discount factor for the signal noise relative to the observation noise. However, in this case a discount factor is not enough because the relationship between the models is dynamic hence a slightly more involved set of functions are needed to modify the parameters with respect to the changing tree shape and each other.

Figure 5.3.1a shows the increase in the minimum and maximum number of leaves



(a) The change in the number of leaves as the depth of the tree increases.



(b) The change in size of F_b for several sizes and scales.

Figure 5.3.1

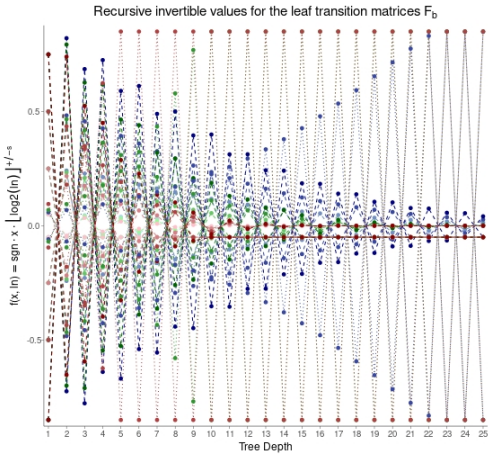
as the depth of the tree increases. The depth of the tree has been limited to 25. This is an unlikely depth of tree based on the prior study but not exceptional. There are two important things to notice from this figure. The first is that the number of leaves has an exponential growth shape with respect to depth. It is this shape that informed the choice for the type of function that would be used to control leaf parameter growth and decay. The second thing to notice is that as the tree gets deeper so the probability that a leaf is updated exponentially decreases. Alternatively, the deeper the tree gets the larger the effect of the alternate models (models not selected at each t) on the tree marginal. If one refers to Section 4.5, there it is shown how misspecification of leaf parameters can distort the tree marginal. More specifically, if a leaf is not updated often enough the tree marginal could appear to favour the wrong tree.

Figure 5.3.1b shows how the transition parameter decreases (increases) as a function of tree depth. The functions for adapting the transition parameter as the tree grows are:

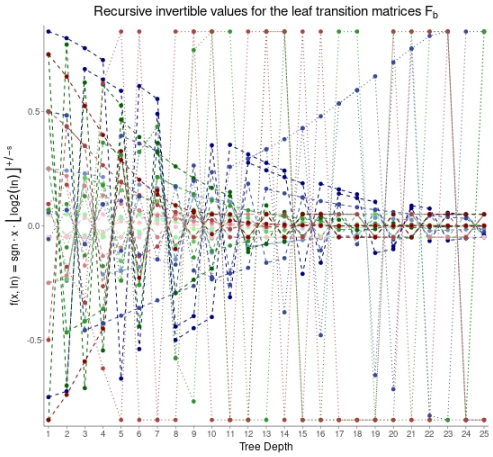
$$f(F_{\mathrm{P}(b)}, b, s, \mathrm{sgn}, \min_F) = \max\{\mathrm{sgn} \cdot F_{\mathrm{P}(b)} \lfloor \log_2(b) \rfloor^{-s}, \min_F\} \tag{5.13}$$

$$f^{-1}(F_{\mathrm{C_{l,r}}(b)}, b, s, \mathrm{sgn}, \max_F) = \min\{\mathrm{sgn} \cdot F_{\mathrm{C_{l,r}}(b)} \lfloor \log_2(b) \rfloor^{s}, \max_F\} \tag{5.14}$$

where $F_{\mathrm{P}(b)}$ is the transition parameter from the parent leaf in a grow move, $F_{\mathrm{C_{l,r}}(b)}$ is the transition parameter from either of the child leaves in a prune move, s is a scaling parameter that scales the rate of change of the parameter and $\min_F, \max_F$ are preset parameters that truncate the range of the parameter. The different colours in the graphs show scales $s =$ blue $= 0.05, s =$ green $= 0.1, s =$ red $= 0.2$. The dotted (for functions that read in reverse i.e. as tree depth



(a) The change in size of F_b when the sign of F_b is negative.



(b) The change in size of F_b when the sign of F_b is randomly allocated.

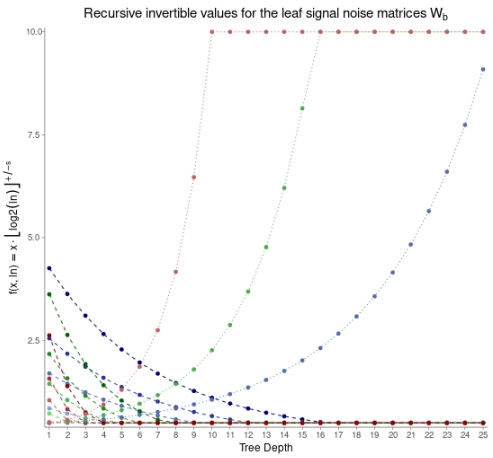
Figure 5.3.2

decreases) and dashed (for forward reading functions) lines are only there there as a visual aid because the depth count is a discrete value. The different values for F_b are $F_b = (0.05, 0.25, 0.5, 0.85, -0.05, -0.25, -0.5, -0.75))$ (shown by the different shades of line: there is only a single shade for the reverse direction), the minimum value is set to 0.05 and the maximum value to 0.85 and the sgn = 1 for all examples.

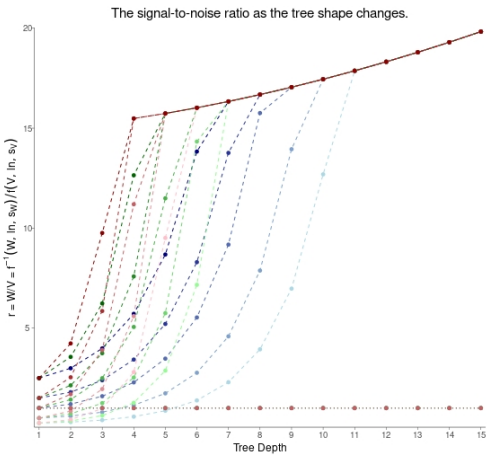
Figures 5.3.2a and 5.3.2b respectively show the effect of a negative only (sgn $= -1$) ancestry of F_b parameters and randomly changing the sign of the F_b parameter when all other parameters of $f(\cdot)$ and $f^{-1}(\cdot)$ are kept constant. The important things to notice are that the transition parameter is always bounded between the minimum and maximum values stipulated for F_b. Next, as the tree grows, the transition parameter always decreases in size at a rate that is set by s. Further, the larger (deeper) a tree gets the more restrictive the parameter F_b becomes. Glancing back at Equations (4.24) to (4.31) it is clear to see the intuitive appeal of this behaviour: as the tree gets bigger the contribution of each leaf gets smaller and, because the leaf that is updated is more influenced by H_b and V_b, the contribution of the updated leaf can be controlled independently from those that are not updated.

In the grow-shift and prune-shift moves there are no parent or child leaves to inherit parameters from so either a base parameter for each tree is set or a leaf is uniformly chosen and then that parameter is adopted for the new leaf and adapted based on $f(\cdot)$ or $f^{-1}(\cdot)$ from Equations 5.14 depending on whether it is a grow-shift or prune-shift move. The effects of this modification are not as consistent as those shown in Figures 5.3.1b, 5.3.2a and 5.3.2b because one is

either starting from a base parameter that has not been updated to the relative depth at the previous level or one is starting from a parent/child of an arbitrary depth.



(a) The change in size of W_b when $V_b = 1$ is fixed and the tree depth is changing.



(b) The change in the signal to noise ratio $r = W_b/V_b$ when $V_b = 2$ but changes with tree depth and W_b has a range of different sizes.

Figure 5.3.3

Figure 5.3.3a shows the functions in Equation (5.16). These functions are very similar to those in Equation (5.14) with the obvious necessary difference being that neither W_b nor V_b can achieve negative values. Again the different colours show different scales those being $s =$ blue $= 0.05, s =$ green $= 0.1, s =$ red $= 0.2$. The different values for W_b are $W_b = (0.5, 1, 2, 3, 5)$ (shown by the different shades of line: there is only a single shade for the reverse direction), the minimum value is set to 0.5 and the maximum value to 10.

$$g(W_{\mathrm{P}(b)}, b, s, \min_{W}) = \max\{W_{\mathrm{P}(b)} \lfloor \log_2(b) \rfloor^{-s}, \min_{W}\} \tag{5.15}$$

$$g^{-1}(W_{\mathrm{C_{l,r}}(b)}, b, s, \min_{F}) = \min\{W_{\mathrm{C_{l,r}}(b)} \lfloor \log_2(b) \rfloor^{s}, \max_{W}\} \tag{5.16}$$

A central difference shown in Figure 5.3.3a is that $g(\cdot)$ is used as the tree decreases in depth and $g^{-1}(\cdot)$ is used as the tree increases in depth. This seems counter intuitive because the signal noise seems to be decreasing as the tree becomes more refined and increasing as the tree becomes coarser. However one must bear in mind that as the number of leaves increases, the signal noise is getting stronger because the number of leaves is increasing which results in more of the variation of the observed process being explained by the tree. The aim here is to balance the effect of the increase in the number of models that are not updated with the

change in the contribution of each of these models at each t. Similarly, and in keeping with the ideas expressed in Chipman et al. (1998) and "BART: Bayesian additive regression trees", if the size of the signal in smaller trees is favoured then smaller trees in general might be favoured and this should help to prevent excessive tree growth.

Turning to Figure 5.3.3b one can see the overall effect of the signal-to-noise ratio as the tree is modified. In this diagram both W_b and V_b change as the tree depth changes but importantly, V_b changes in the opposite direction to W_b. That is, as tree depth increases, for W_b, $g^{-1}(\cdot)$ is used and for V_b, $g(\cdot)$ is used. As tree depth decreases, the reverse happens where for W_b, $g(\cdot)$ is used and for V_b, $g^{-1}(\cdot)$ is used. The overall effect though is that, in the forward direction that the signal-to-noise ratio $r = g^{-1}(\cdot)/g(\cdot)$ increases but in the reverse direction $r = g(\cdot)/g^{-1}(\cdot)$ is a constant value. In this demonstration the scale values for W_b, $s_W = (0.25, 0.5, 0.75)$ and the values for W_b were as before. Only a single value was used for $V_b = 2$ and the scale value was $s_V = 0.01$. The minimum and maximum values for W_b, $\min_W = 5$ and $\max_W = 30$ and minimum and maximum values for V_b were $\min_V = 0.1$, $\max_V = 5$. The reason for changing the values of W_b is that these are an example of actual values used in the experiments that follow and because the values chosen in Figure 5.3.3a show the behaviour of W_b more clearly than those used in Figure 5.3.3b.

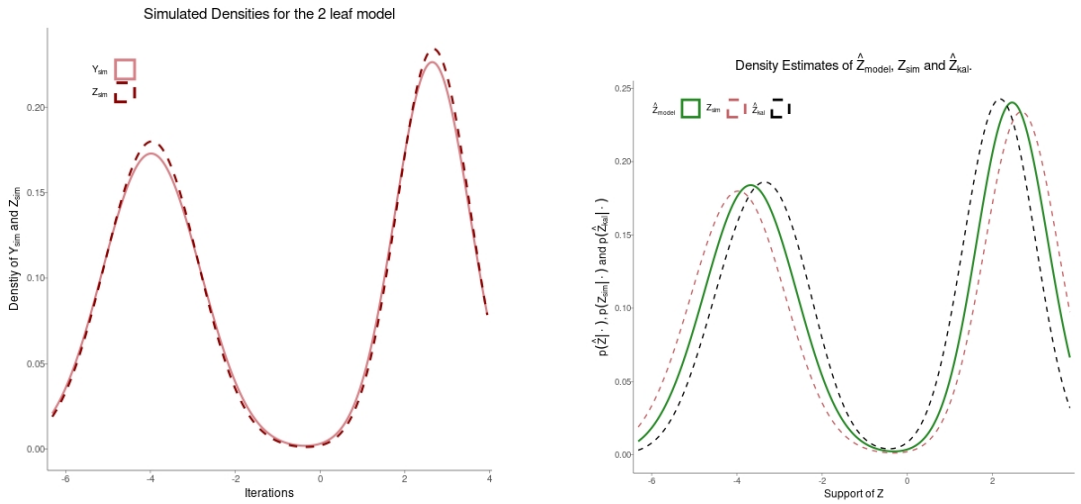
The next subsection turns to demonstrations of a single tree model for 3 different simulated data sets. The main result of these demonstrations is that the tree model learns from the data to improve the accuracy of the estimation of the mean level and the uncertainty of this estimate over the single tree filter. This is assessed by comparing the models to the Kalman filter.

5.3.2 The 2 Leaf Model

The 2 Leaf Model is the same as that in the simulation studies of Section 4.5. Again the Kalman filter is used as a benchmark because it is an optimal estimator, conditional on its parameters. The improvement shown here does not mean that the tree filter with random model search is in some way "more optimal" than the Kalman filter. Rather the improvement in both accuracy and uncertainty estimation is that one is able to specify more than one set of parameters for several Kalman filters. The tree model is able specify particular conditional models (filters) with different parameters at each t. Thus it is a case choosing the right Kalman filter at each t not improving over the Kalman filter.

The improvement in the random tree model over the fixed tree filter is that

there, like the Kalman filter, the leaf filter parameters were fixed and incorrect specification of these meant that estimation was poor. In the random tree model the tree structure (i.e. number of filters) can change, different filter models can be selected from a deterministic range of parameters as described in Section 5.3.1 and the rules for choosing filters (i.e. threshold values in the tree structure) can be chosen so that the choice of filter models, not only the number of filter models, can be improved.



(a) (b)

Figure 5.3.4: Density estimates based on simulated and estimated parameters for the 2 leaf model.

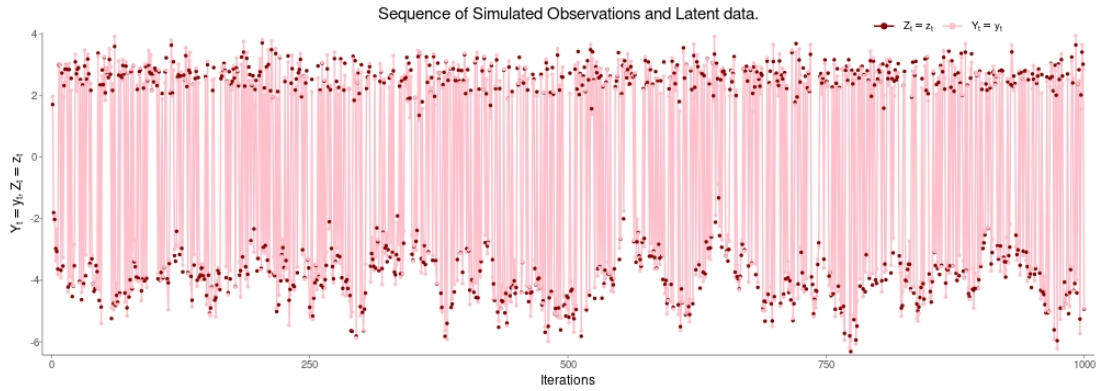
Figure 5.3.4a shows the density from which the data were generated and Figure 5.3.5a shows the sequence of data that were used in the simulation studies. The lines in Figure 5.3.5a show what the signal might look like if represented as a continuous line but the dots show the actual observed values.

The parameters used to initialise the tree model and used used for the Kalman filter are: These parameters are quite different to those used for the same data in

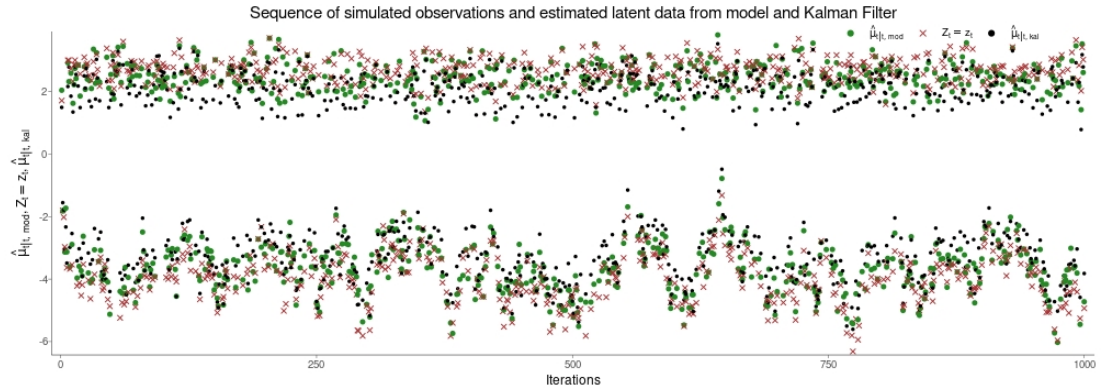
Model params for Random Tree, α=0.95, β=2						
Leaf	V_b	W_b	F_b	H_b	$Z_{0,b}$	$W_{0,b}$
2	1	4	0.7	1	0	10
3	1	6	-0.7	1	-4	10

Section 4.5.1. Arriving at this choice of parameters involved using the calibration study of Section 4.4, several trials of the functions in Section 5.3.1 and their parameter settings as well as a fair bit of experimentation.

Figure 5.3.4b shows that neither the Kalman filter nor the proposed (BTRS) model exactly match the simulated density although the both are not too far away but the BTRS model is closest. These estimated density plots do not show



(a)



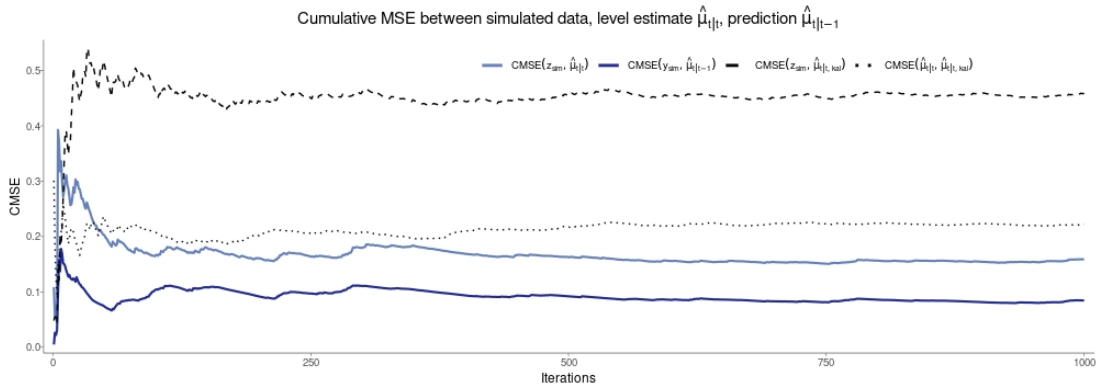
(b)

Figure 5.3.5: Sequences of data simulated from a 2 leaf model and then estimated by a random tree model.

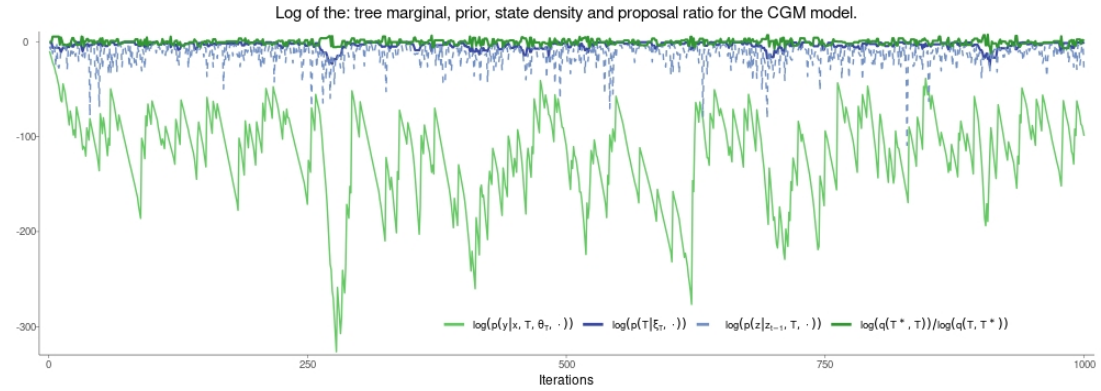
the full picture in that they calculate a "flat" density. That is, the y calculation is based on the data as a whole rather than considering the density at different time points in the sequence.

The sequential plot in Figure 5.3.5b addresses this to some degree by showing the points of the data. In this plot it can be seen that the BTRS tree model adapts to the sequence of data while the Kalman filter only does so to a lesser degree. For example, between iterations 0 to about 100 in the lower portion of the graph both the Kalman filter and the BTRS model are more similar at estimating the state variable but as the number of iterations progresses the BTRS model improves over the Kalman filter so that if one looks at the points in the lower portion of the graph between 900 and 1000 the BTRS model ($\hat{\mu}_{t|t,mod}$) has more points closer to the simulated points than the Kalman filter does.

This improvement in the BTRS model over the Kalman filter is most clearly seen in Figure 5.3.6a. Here the dotted line represents the cumulative MSE between the Kalman filter estimate and the BTRS estimate of the state process. For the first 400 iterations the CMSE of the Kalman filter and BTRS model seem fixed at a more or less constant difference apart but after these iterations there is an



(a) Cumulative error between the model and simulated stream and the Kalman Filter



(b) Marginal terms of a single random tree model for the the 2 leaf data simulation.

Figure 5.3.6

improvement in the BTRS model CMSE but none in the Kalman filter CMSE.

Figure 5.3.6b shows the different components that are used in calculating the acceptance ratio of the MCMC sampling process. The first thing to notice is that the tree marginal is no longer linearly decreasing as it was in Section 4.5.1. In fact, the marginal of the tree looks to be (quite wildly) changing around some constant value. A guess as to why this is happening is that the tree is changing shape and thus changes the information in the tree. By change in information it is meant that not only is the data accumulating in the model but the number of models (isomorphic to the number of leaves) is also changing which means that for each new data point added, the number of non-updated models is changing too.

As was shown in Section 4.5.1, the contribution of the nonupdated models, via their parameters and their contribution to the tree marginal, can have an effect to the extent of distorting not only the estimates of the state but also the estimate of the model itself. This changing marginal, itself an expected value of the tree model with respect to the latent data, shows how the tree model is searching the model space by using the latent processes and the new data. This is not to

say that each new data point causes the swings in the marginal but these swings suggest that the accumulation of data and the proposals of alternate models, their associated subprocesses and their parameters collaborate to search for a better model.

Other interesting features of this graph can be seen between iterations 250 and 300 and between 700 and 750. In both these cases the prior (the dark blue line) decreases quite sharply followed by sudden changes in the proposal distribution (the top dark green line) which are then followed by a change of direction in the marginal of the tree likewise in the prior. This suggests a form of hysteresis of the model is being reached but that a combination of prior, proposal moves and data are allowing the tree model move away from this fixed point.

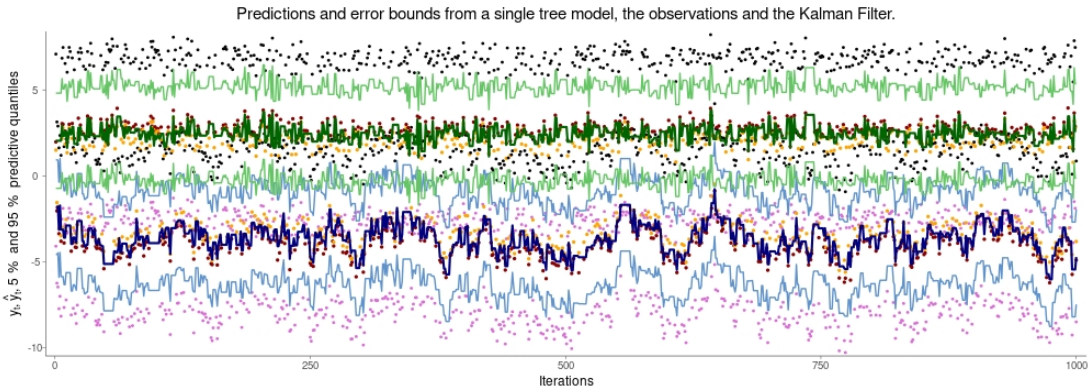


Figure 5.3.7: Estimation of the simulated stream of data including the error bounds.

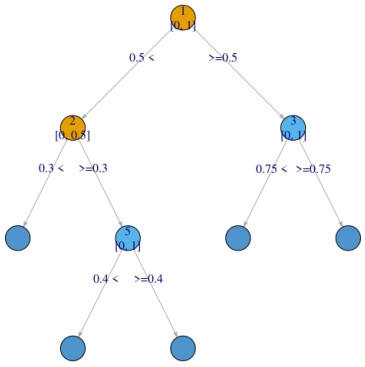
The final graph, Figure 5.3.7, shows the predictions of the observed process and the predictive uncertainty bounds. Due to the nature of the data this graph is quite difficult to read. The Kalman filter state estimate is represented by the orange dots and the simulated data but the red dots. The estimates of the state by the BTRS model are represented by the dark blue and green lines and their error bounds are, respectively, the lighter blue and green lines. The lower errors for the Kalman filter are both the lower set of pink dots. That is, the lower bound for the upper state estimate is the upper set of pink dots and the lower bound for the lower state estimate is the lower set of pink dots. The upper bounds are represented by the black dots with a similar ordering. This graph reflects the same information that was presented in Section 4.3: the tree filter model provides improved estimation of the uncertainty of the state estimates because the tree model is able to conditionally associate particular predictions with particular models.

The next section will present the same graphs but for the CGM inspired, 5 leaf model. This example is provided to show that the proposed model works for more

complicated data sets and it also raises some issues that are resolved in Chapter 5 where the CGM model becomes the main example model.

5.3.3 The CGM Model

The CGM Model is generated from a 5 leaf tree that was inspired by the example given in Chipman et al. (1998) but adapted to the streaming setting and only continuous covariates and thresholds have been used. This form of model was chosen because the density has 4 clear modes and one occluded mode as Figure 5.3.9a shows. The leaf parameters for the CGM data generator are included



(a) The adapted tree model from (Chipman et al. 1998)

Simulation leaf parameters.

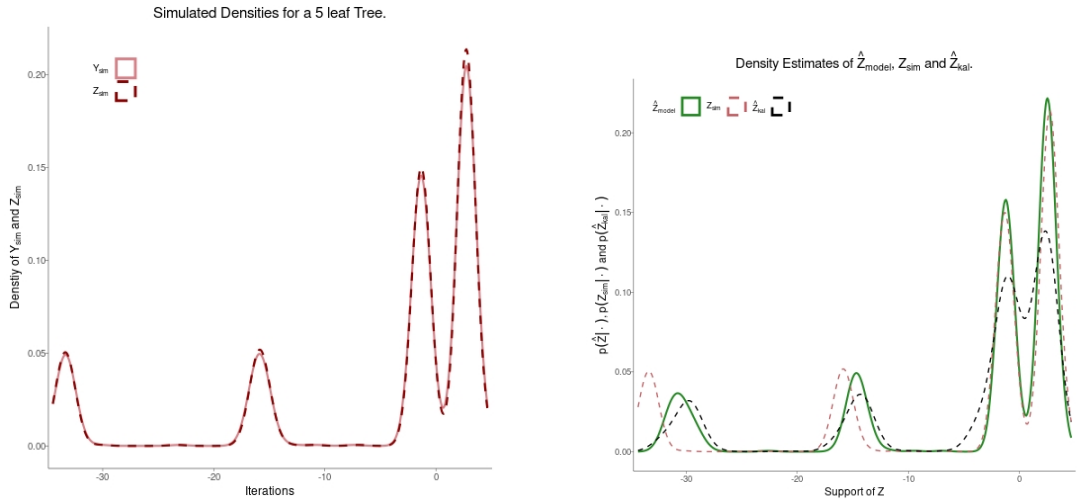
Leaf	V_b	W_b	F_b	H_b	$Z_{0,b}$	$W_{0,b}$	u_b	G_b
4	0.02	0.1	0.25	1	0	1	1	-1
10	0.03	0.15	-0.5	1	5	1	1	5
11	0.05	0.15	0.5	1	8	1	-1	8
6	0.05	0.01	-0.5	1	4	1	1	4
7	0.05	0.15	0.25	1	2	1	-1	25

(b) The leaf parameters for the 5 leaf tree adapted from (Chipman et al. 1998) for the streaming setting.

Figure 5.3.8

below Figure 5.3.8a. Figure 5.3.9a shows the estimation of the density for the CGM model. Note that the visible modes of the model on the left correspond to the data generated from leaves 7 and 11. The other three leaves generate the two modes on the right. Figure 5.3.9b shows estimations of the densities of the latent state based on their estimates of the parameters. The difference between the tree model and Kalman filter in this case is more pronounced than it was in the 2 leaf model. This is not surprising given the degree on nonstationarity of this data generator.

Figure 5.3.10a presents the sequence of data over 1000 iterations in both a discrete

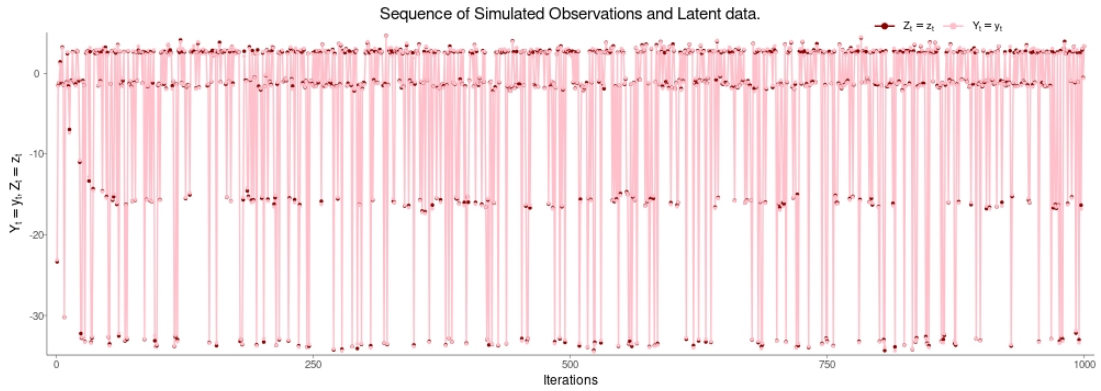


(a) (b)

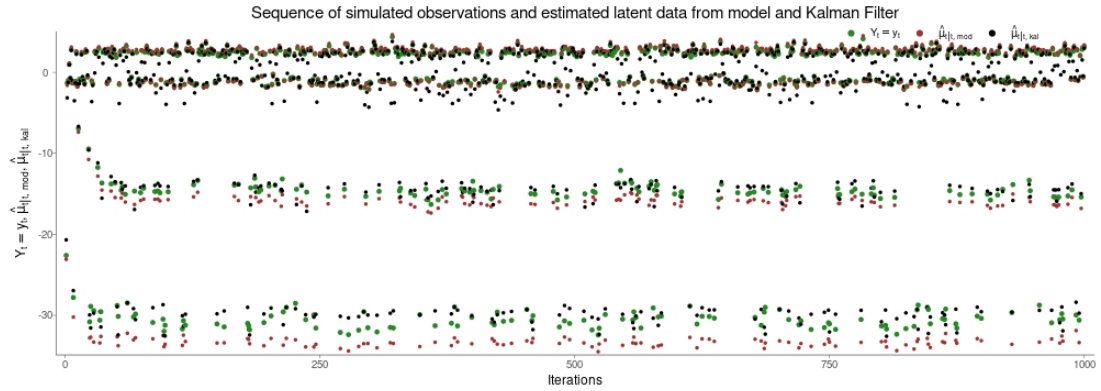
Figure 5.3.9: Density estimates based on simulated and estimated parameters for the 5 leaf, CGM model.

and continuous form. Figure 5.3.10b shows the sequence of estimates of the state produced by the tree model and by the Kalman filter. Not too much will be said about this here because a great deal of Section 5.3.5, Section 5.4.2 and Chapter 6 will delve into this model in some detail. Something to notice in Figure 5.3.10b is that the estimates of the state by both the Kalman filter and the proposed model (BTRS) do not converge towards the simulated data. The non-convergence of the model and Kalman filter can be seen more clearly in Figure 5.3.9a which shows the cumulative mean square error (CMSE) between simulated state and the the Kalman estimates of the state, $CMSE(z_{sim}, \hat{\mu}_{t|t,kal})$, between the simulated state and the BTRS model, $CMSE(z_{sim}, \hat{\mu}_{t|t,BTRS})$ and between the BTRS model and the Kalman filter estimates of the state, $CMSE(\hat{\mu}_{t|t,BTRS}, \hat{\mu}_{t|t,kal})$. In this case all three measures reach a more or less fixed state and do not consistently vary away from this state over the 1000 iterations. This in contrast to the 2 leaf model shown in Figure 5.3.6a but this is not a reflection of the model, only an example of the the behaviour of the model under particular specifications, only shown here to make the reader aware of them and to point out that the issue can be resolved through tree learning for the BTRS while it cannot, at least without stopping and restarting the model, for the Kalman filter. Section 5.3.5 will show how this is resolved.

Figure 5.3.12 shows the components of the acceptance ratio. Similarly to the two leaf model the tree marginal component varies slightly wildly about some constant rather then getting progressively lower. In contrast to the two leaf model, the log of the densities has greater variation than it does in the two leaf model. This means that there is greater variation in the samples that that are drawn from



(a)



(b)

Figure 5.3.10: Sequences of data simulated from the CGM model and then estimated by a random tree model.

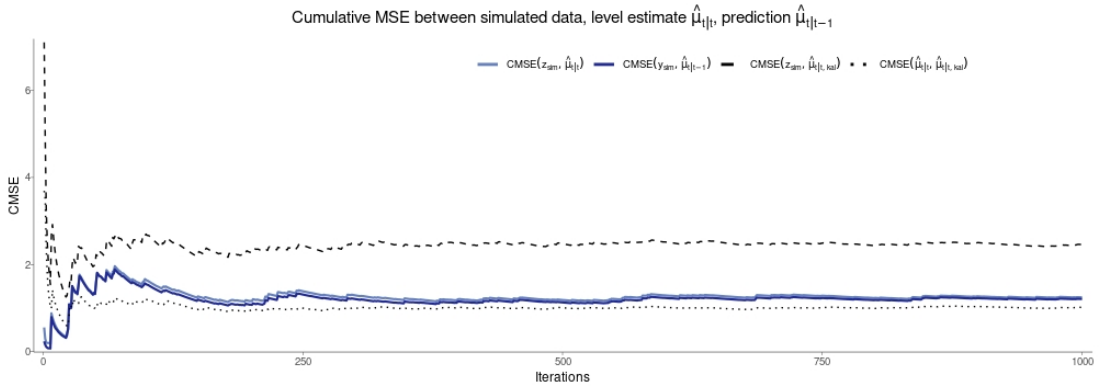
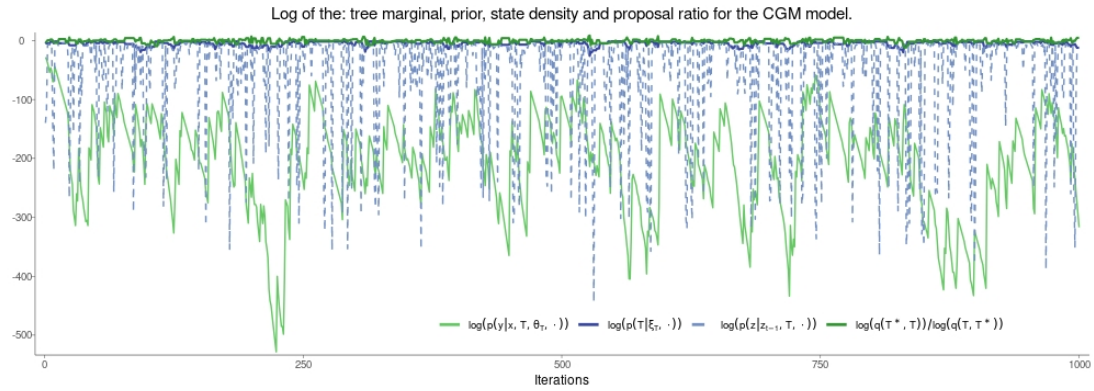


Figure 5.3.11: Cumulative error between the CGM model and simulated stream and the Kalman Filter

each leaf density which is not surprising considering the comparative range of possible modes in this model.

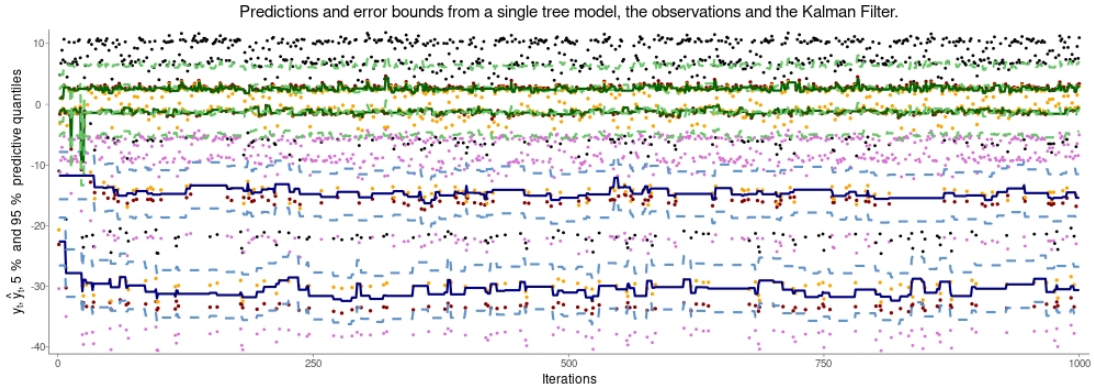
The final figure shown here is the sequence of predictions made by the model and the Kalman filter. Again this is a difficult graph to read because there is a lot being shown. The pink dots are the lower bounds of the Kalman filter predictions shown in orange and the black dots the upper bounds. The dashed lines are the



(a)

Figure 5.3.12: Marginal terms of a single random tree model for the the CGM stream simulation.

upper and lower bounds of the BTRS model predictions. They are shown dashed to emphasise that they are pointwise estimates taken from different models at each t and not a continuous upper and lower bound. What is interesting about this graph is the long intervals between the updates shown as straight lines in dark blue. These two sequences are updated less frequently than the two green sections (which actually consists of three leaves in Figure 5.3.8a). This means that, for the single tree model not only may there be some time before an estimate and prediction are updated but that when a prediction from the filter is updated it is likely to be more wrong than if the predictions were updated more frequently. This links back to Figure 5.3.12 where it was seen that there is greater variation in the log densities of the samples taken from the latent process as a part of the MCMC sampling procedure.



(a)

Figure 5.3.13: Estimation of the simulated stream of data including the error bounds.

The next section will briefly look at the Friedman model, a nonlinear function of 5 explanatory variables that has been used by other tree modellers most usually to show that a tree model can distinguish between "true" model covariates and

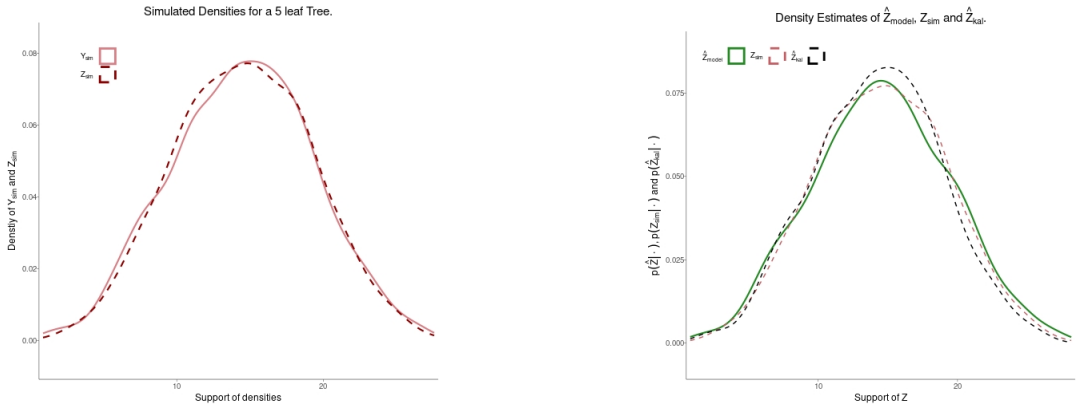
noisy covariates.

5.3.4 The Friedman Model

The Friedman model is a standard benchmark model that is used in many papers on tree modelling, originally in J. H. Friedman (1991) but also in both Taddy et al. (2011) and Chipman et al. (2010) to compare their approaches to the wider field of tree modelling. The data are generated from a function:

$$y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \mathsf{N}(0,1) \tag{5.17}$$

where each of the values $x_{i1}, \ldots, x_{i5}$ are generated from $\mathsf{U}(0,1)$ for $i \in 1, \ldots, 1000$. An additional 5 covariates whose values are not used in the model are generated to create some noise around the choice of covariate for the models.

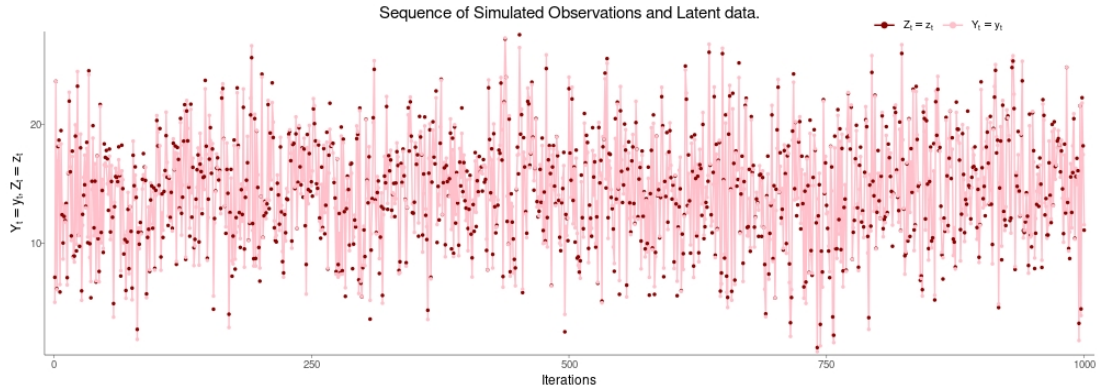


(a) (b)

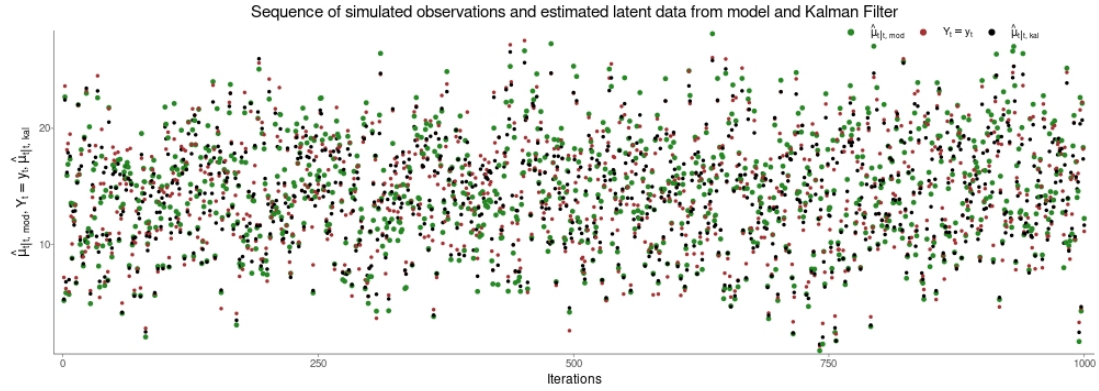
Figure 5.3.14: Density estimates based on simulated and estimated parameters for the Friedman model.

The estimated densities of the generated data and estimated data are shown in Figures 5.3.14a and 5.3.14b. The BTRS model seems to perform well in this case, possibly slightly better than the Kalman filter. Due to the nature of this data not much can be seen from the sequence of estimates of the generated data but turning to Figure 5.3.16a it is there shown that the Kalman filter has better estimates cumulatively over 1000 iterations. The cumulative predictions seem to out perform the cumulative estimates and it appears as though the estimates of the state level are converging towards the Kalman filter estimates. However, several runs of this model over several different numbers of iterations and different parameter settings showed that on average the estimates by the proposed model did not continue to approach the Kalman filter estimates although they did not get consistently worse either. It is not clear why this is the case and perhaps the

ensemble method will offer some improvement.



(a)

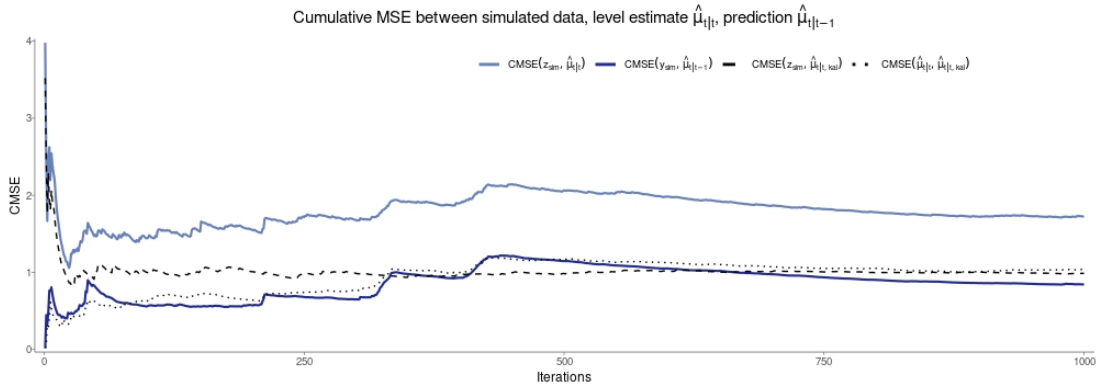


(b)

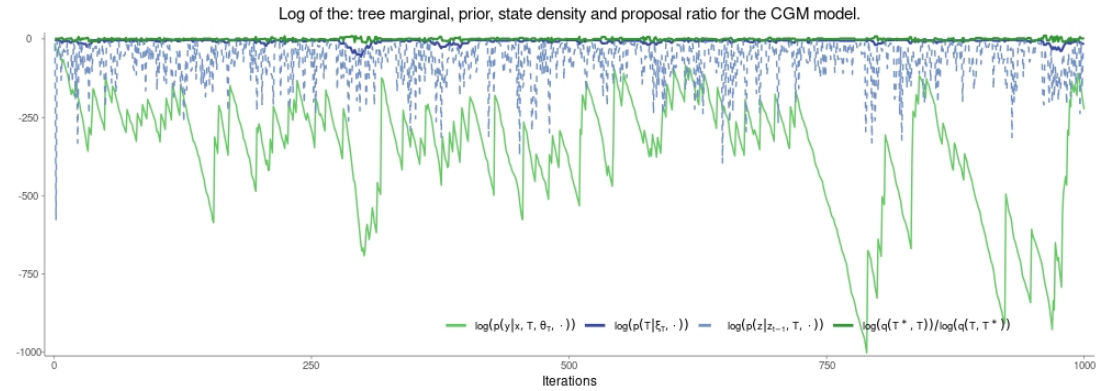
Figure 5.3.15: Sequences of data simulated from the Friedman model and then estimated by a random tree model.

Figure 5.3.16b shows some large deviations from what appears to be a fairly constant fixed level. However these cannot be linked to changes in the CMSE nor changes in the generated sequence so likely are a result of random recursive behaviour of the tree model evolution process. The bounds of the BTRS model in Figure 5.3.16c are again tighter than the Kalman filter bounds, where the lower bounds of the Kalman filter are in pink while the upper bounds are in black.

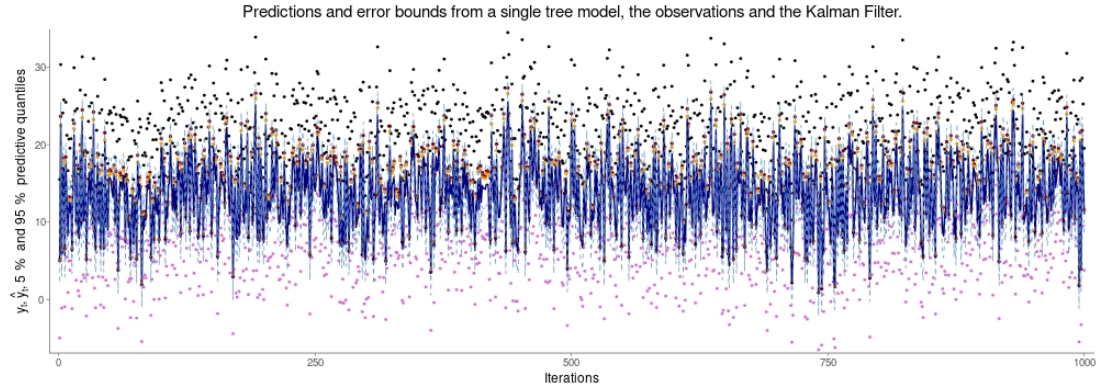
There is not too much more to be said about this example but it will be briefly revisited in Section 5.4.2 to see if the ensemble approach provides a better estimated level for this benchmark data set. However, it is important to note that a single tree model provides estimates for which the estimated density is extremely close the to estimated density of the generated data. While an ensemble of trees might track the latent level better at least one objective has been achieved: a single tree model can target a benchmark density using MCMC with only a single visit to each data point, with no storage or direct input and in constant time. However a single chain targeting a distribution is insufficient because:



(a) Cumulative error between the Friedman model and simulated stream and the Kalman Filter



(b) Marginal terms of a single random tree model for the the Friedman stream simulation.



(c) Estimation of the simulated stream of data including the error bounds.

Figure 5.3.16

- it has its scope limited by the number and type of models,
- a single chain is insufficient to explore the full model space,
- and it is unlikely that a single chain will adapt to new data, especially new data that is quite different from the data that has already been seen, as well as a collection of chains might adapt.

The next subsection will show the results of a simulation study that attempts to uncover which of the parameters have the most notable effect (or lack thereof)

on the single tree model performance and size.

5.3.5 Experimental Simulation Study

The simulation study in Appendix A1.3 shows the effect of different tree parameters $\xi_T = (\alpha, \beta)$ on a collection of different leaf model parameters $\psi_T = (V_b, H_b, F_b, W_b, \mu_{0,b}, \Sigma_{0,b})$. Data are generated from CGM model above. These data are compared with 5 independent Kalman filters with the same filter parameters settings found in the CGM leaf parameters.

The results of these experiments show:

- The fundamental validity of this approach: The independent Kalman filters (with fixed parameters) cannot converge on the conditional distribution while random generating trees doe converge more accurately to the true distribution.
- The largest effect on the accuracy of the model is the filter parameter H_b.
- Specifying ξ_T can keep tree sizes small but to keep tree size under control both ξ_T and ψ_T require 'correct' specification.

An additional study, not yet completed would be to look at the effects of the adaptations of the filter parameters as the tree changes shape, both under deterministic modification and sampling.

5.4 Tree Ensemble Modelling

The final constituent of the model Equation (4.1) of the Introduction is the ensemble $\mathcal{F}$ itself. Equation (4.1) is reproduced in Equation (5.18) for convenience where Θ is the collection of all model parameters, ψ_T is the collection of the model parameters for models $\mathcal{M}$ at the leaves of each tree and ξ_T are the tree model parameters for each tree.

$$Y^t \approx \mathcal{F}(x^t, \Theta, \varepsilon^t, t) = \sum_{\forall T \in \mathcal{F}} T(g(Z^t, \psi_T, t), x^t, \varepsilon^t, \xi_T)\pi(T \mid Y^t). \tag{5.18}$$

$\pi(T \mid Y^t)$ is shorthand notation for the posterior probability $\mathsf{P}(T \mid Y^t, \theta_T, x^t)$ using the density from Equation (5.4) where θ_T is the collection of parameters ξ_T, ψ_T for an individual tree.

This mixture model is a mixture of Gaussian densities produced by $g(Z^t, \psi_T, t)$ for every T which has mean $\hat{\mu}_{t|t,b_{x_t}}$ and variance-covariance $\hat{\Sigma}_{t|t,b_{x_t}}$ at each t where b_{x_t} is the leaf node chosen by x_t at index t in model T.

This ensemble model differs from the Bayesian Additive Regression Tree (BART) model of Chipman et al. (2010) in several ways. Apart from the fact that this model is a streaming model that produces estimates and predictions at index t and does not use Bayesian backfitting, the BART model of Section 2.4 is a additive model of mean values μ_i over a fixed set of data where $\mathsf{E}[Y \mid T, x]$ is the sum of the terminal nodes values with Gaussian error over the ensemble. That is, the expectation is taken with respect to the Gaussian distribution. In this proposed model $\mathsf{E}[Y_t \mid T, x_t, \cdot]$ is the expectation with respect to the posterior distribution of tree models. Each estimate or prediction is a component of the mixture and the weight of this component is provided by the (posterior) probability of that component considered over the support of all possible tree models.

However it is not possible to fully explore every possible tree model. While each tree is a discrete object it is possible that there are an infinitely large number of nodes and hence terminal nodes for each tree model not to mention that there are an infinitely large number of threshold choices for a finite number of covariates. Rather the space of tree models is approximated by the finite ensemble which provides the approximate probability of a tree:

$$\mathsf{P}^*\left(T \mid Y^t, x^t, \theta_T\right) = \pi^*(T \mid Y^t) = \frac{\pi(T \mid Y^t)}{\sum\limits_{T \in \mathcal{F}} \pi(T \mid Y^t)} \tag{5.19}$$

via an approximation to $\mathsf{p}(T, Z^t \mid Y^t, \theta_T, x^t)$ which was only defined up to a

constant of proportionality in Equation (5.3).

This section will firstly describe the predictive distributions for the latent variable Z^t and $Y(t)$. This will be followed by a demonstration of the full model for three different simulated data sets and a simulation study to assess the effect of the size of the ensemble on predictive accuracy. The final part of this section will compare the proposed model to two different models in two forms. The first two comparisons will be to compare this model (BTRS) against that of Dynamic Trees (DT) (Taddy et al. 2011) which has been modified in the R® package DynaTree (Gramacy et al. 2023) to deal with streaming data and against a version of BART (Chipman et al. 2010) that has been modified to predict a current data point based on the all data received up to that point. The next two comparisons will again be against DT and BART but this time the datasets will be considered fixed and the test will be to see how the proposed algorithm performs having only had a single pass over the data.

5.4.1 Predictive Distributions

Let $z^*_{x_t,T}$ be the predicted value for the random variable Z_t at each tree T chosen by x_t at index t. $z^*_{x_t,T}$ is predicted from T by the Kalman filter model at the leaf $b^T = b_{x_t,T}$ chosen by covariate vector x_t at every t. The prediction at the chosen leaf is $\mathsf{p}\left(z^*_{b^T} \mid T, \theta_T, Y^t, x^t \right)$ which has a Gaussian distribution:

$$\mathsf{p}\left(z^*_{b^T} \mid T, \theta_T, Y^t, x^t \right) \sim \mathsf{N}(\hat{\mu}_{t|t-1,b^T},\, \hat{\Sigma}_{t|t-1,b^T})$$

where

$$\hat{\mu}_{t|t-1,b^T} = F_{t,b^T}\hat{\mu}_{t-1|t-1,b^T} \qquad \text{and} \qquad \hat{\Sigma}_{t|t-1,b^T} = F_{t,b^T}\hat{\Sigma}_{t-1|t-1,b^T}F^T_{t,b^T} + W_{t,b^T}$$

where $\theta_T = (\phi_T, \xi_T)$ is the collection of leaf and tree model parameters for tree T.

For the ensemble model the prediction z^*_t for Z^t is taken with respect to the marginal distribution of the tree models that is approximated by Equation (5.19). This has value:

$$\begin{aligned}
\mathsf{p}\left(z^*_t \mid \Theta, Y^t, x^t \right) &= \sum_{\forall T} \mathsf{p}\left(z^*_{b^T} \mid T, \theta_T, Y^t, x^t \right) \mathsf{p}\left(T \mid Y^t, x^t, \theta_T \right) \\
&\approx \sum_{T \in \mathcal{F}} \mathsf{p}\left(z^*_{b^T} \mid T, \theta_T, Y^t, x^t \right) \mathsf{P}^*\left(T \mid Y^t, x^t, \theta_T \right) \qquad (5.20)
\end{aligned}$$

where now $\Theta = (\Psi, \Xi)$ the collection of all ensemble model parameters. The distribution of each prediction component $\mathsf{p}\left(z^*_{b^T} \mid T, \theta_T, Y^t, x^t \right)$ is Gaussian so

that the distribution of the prediction from the ensemble is a Gaussian mixture with mean:

$$\mathsf{E}\left[\, Z_t^* \mid \Theta, Y^t, x^t \,\right] = \sum_{T \in \mathcal{F}} \mathsf{P}^*\left(T \mid Y^t, x^t, \theta_T\right) \hat{\mu}_{t|t-1,b} \tag{5.21}$$

and variance

$$\begin{aligned}\mathsf{Var}\left[\, Z_t^* \mid \Theta, Y^t, x^t \,\right] = \sum_{T \in \mathcal{F}} \mathsf{P}^*\left(T \mid Y^t, x^t, \theta_T\right) \times \\ \left(\hat{\Sigma}_{t|t-1,b^T} + \left(\hat{\mu}_{t|t-1,b^T} - \mathsf{E}\left[\, Z^*(t) \mid \Theta, Y^t, x^t \,\right]\right)^2\right)\end{aligned} \tag{5.22}$$

where $\hat{\Sigma}_{t|t-1,b^T}$ is the tree (chosen leaf) model error and $\left(\hat{\mu}_{t|t-1,b^T} - \mathsf{E}\left[\, Z^*(t) \mid \Theta, Y^t, x^t \,\right]\right)^2$ is ensemble model error.

To get the predictive distribution for y_{t+1} from the current set of data that includes observations up to y_t, but also including the new explanatory vector x_{t+1}, condition on each of the tree models $T \in \mathcal{F}$ and on the prediction z_t^* to get:

$$\begin{aligned}\mathsf{p}\left(y_{t+1} \mid \Theta, y^t, x^t, x_{t+1}\right) &= \sum_{\forall T} \mathsf{p}\left(y_{t+1} \mid T, \theta_T, y^t, x^t, x_{t+1}\right) \mathsf{p}\left(T \mid y^t, x^t, x_{t+1}, \theta_T\right) \\ &\approx \sum_{T \in \mathcal{F}} \mathsf{P}^*\left(T \mid Y^t, x^t, x_{t+1}, \theta_T\right) \times \\ &\qquad \int \mathsf{p}\left(y_{t+1} \mid z_{b^T}^*, T, \theta_T, y^t, x^t, x_{t+1}\right) \mathsf{p}\left(z_{b^T}^* \mid T, \theta_T, y^t, x^t, x_{t+1}\right) dz_{b^T}^*\end{aligned} \tag{5.23}$$

The integral in Equation (5.23) is the integral of the product of two Gaussian distributions where the terms have distributions:

$$\begin{aligned}\mathsf{p}\left(z_{b^T}^* \mid T, \theta_T, y^t, x^t, x_{t+1}\right) &\sim \mathsf{N}\left(\hat{\mu}_{t|t-1,b^T},\, \hat{\Sigma}_{t|t-1,b^T}\right) \\ \mathsf{p}\left(y_{t+1} \mid z_{b^T}^*, T, \theta_T, y^t, x^t, x_{t+1}\right) &\sim \mathsf{N}\left(H_{t,b^T}\hat{\mu}_{t|t-1,b^T},\, V_{t,b^T}\right).\end{aligned}$$

However, the distribution of y_{t+1} is the conditional distribution of y_{t+1} on $z_{b^T}^*$.

To get the predictive distribution of y_{t+1} consider the Kalman filter model in the following form:

$$\begin{aligned}y_{t+1} = Hz_t + v_t &= HFz_{t-1} + Hw_t + v_t \\ &= HFz_{t-1} + \varepsilon_t\end{aligned}$$

where ε_t takes into account the error of the signal and the error of the noise.

Meinhold and N. D. Singpurwalla (1983) show that the distribution of ε_t is $\mathsf{N}(0, V + H\Sigma H^T)$ when one is conditioning both the state and the error on the past sequence of observations and not conditioning the observations on the state. That is, they show that joint distribution of the independent error and state processes gives rise to the conditional distribution of the state on the residual errors and hence the Kalman filter equations are a result of the properties of joint the Gaussian distributions.

Therefore the predictive distribution for the observation sequence from the chosen leaves that takes into account both the noise error and signal error under the Markov assumption of the observation sequence is:

$$\mathsf{p}\left(y_{t+1} \mid T, \theta_T, y^t, x^t, x_{t+1} \right) \sim \mathsf{N}\left(H_{t,b^T} \hat{\mu}_{t|t-1,b^T},\, V_{t,b} + H_{t,b^T} \hat{\Sigma}_{t|t-1,b^T} H_{t,b^T}^T \right)$$

The predictive distribution for y_{t+1} from the ensemble is then also a mixture of Gaussian distributions with mean and variance:

$$\mathsf{E}\left[y_{t+1} \mid \Theta, Y^t, x^t, x_{t+1} \right] = \sum_{T \in \mathcal{F}} \mathsf{P}^*\left(T \mid Y^t, x^t, \theta_T \right) H_{t,b^T} \hat{\mu}_{t|t-1,b} \tag{5.24}$$

$$\begin{aligned} \mathsf{Var}\left[y_{t+1} \mid \Theta, Y^t, x^t, x_{t+1} \right] = & \sum_{T \in \mathcal{F}} \mathsf{P}^*\left(T \mid Y^t, x^t, \theta_T \right) \times \\ & \left(H_{t,b^T} V_{t,b} H_{t,b^T}^T + \hat{\Sigma}_{t|t-1,b^T} + \left(H_{t,b^T} \hat{\mu}_{t|t-1,b^T} - \mathsf{E}\left[y_{t+1} \mid \Theta, Y^t, x^t, x_{t+1} \right] \right)^2 \right) \end{aligned} \tag{5.25}$$

The next section demonstrates the ensemble and this is followed by a comparison between the proposed model and two other established Bayesian models, Dynamic Trees (DT) and Bayesian Additive Regression Trees (BART).

5.4.2 Ensemble Model Demonstration

5.4.2.1 Introduction

Section 5.3.3 presented an adaptation of the 5 leaf tree provided by Chipman et al. (1998) to the streaming setting. There it was shown that a single tree does an adequate job of getting near to the the target distribution. However, this single tree required some tuning of parameters to achieve this accuracy and to get estimates and predictions that are an improvement over the Kalman filter. This section aims to demonstrate that by using an ensemble of trees the settings for parameters can be somewhat relaxed, that this will not negatively affect the accuracy of the model and that the mixture of trees will more closely approximate the target distribution.

This demonstration will be begin by describing how one sets

$$\Psi = 2 \times \mathcal{F} \times (V_b, H_b, F_b, W_b, \mu_{0,b}, \Sigma_{0,b})$$

leaf parameters for the initial ensemble weak learners. Then, as before, the results for 1000 iterations will be presented graphically with associated commentary. The data sets to be used to demonstrate the ensemble approach and for comparison will be those used in Section 5.3 but with a specific focus on the CGM model.

5.4.2.2 Choosing parameters

The calibration study and experiments in Sections 4.5.1 and 5.3.5 provided an insight into ranges of parameters that might be used by the ensemble but in all those cases the models were tuned to specific parameters because only one model was being used at a time. One objective of the ensemble is to generalise the range of choice of initial parameters so that more of the model space can be explored and so that the ensemble can be more adaptive to new data.

However, one must also accept that huge ranges of possible parameters may not help the tree models to converge towards the target distribution. The calibration study and deterministic modification of parameters was one approach to help inform a prior choice of deterministic parameter values in the same way that one might have a prior choice of explanatory variables (i.e. assuming 'as if' these parameters and covariates were known (N. Singpurwalla 2006)). Ideally this prior choice of parameters would be accompanied by a prior distribution but due to the number of parameters, and the consequent number of hyperparameters that might be associated with prior distributions this has not been done. Further, parameter inference and model identification is not the main focus of this thesis so prior distributions over parameters, other than the initial parameters of the latent state, would not be used further in this document.

The method to choose parameters requires that one have either some existing response values or some expert input. It is also necessary to point out that the parameters themselves do not necessarily need to reflect the parameters of the "true" model. This was shown in Section 4.5.1 where the "true" model did not necessarily outperform the known wrong models. There it was shown that more accurate predictions depend on the appropriateness of the selected parameters to the data not necessarily whether the parameters for modelling were the same as the parameters that generated the model. One of the purposes of this thesis is to show that tree filter models can learn to make better estimates and predictions than unconditional filters. There is no interest in finding the "best" filter because

this filter may not exist. Therefore, in choosing parameters one must consider what is best for the tree model, not what might look like the "true" filter.

The following list will explain how choices of each of the parameters have been approached:

- A fundamental rule from theory and the calibration study is that F_b for every leaf must be in $(-1, 1)$. It is also assumed that as the tree gets larger, each individual leaf contributes less to the information contained in the set of alternative models. That is, in Equations (4.11) and (4.12) it was shown how F_b predominantly affects the state and in Section 4.5.1 it was shown that excessive contributions from the state, particularly nonupdated models, can upset model performance. So reducing the effect of the individual state estimates and predictions has an intuitive reasoning.
- It has been shown that the effect of the signal-to-noise ratio (SNR) has a profound effect on the performance of the BTRS model. If one fixes the noise level of the initial weak learners to be low enough and the signal level of these learners to be high enough to provide a large enough (SNR) then one can rely on the relationship between W_b and V_b to adjust according to the relationships described by the tree modification functions in Section 5.3.1. The vague terms low enough and high enough depend to some extent on the problem so some initial investigation and expert input might be helpful.
- The parameter H_b has been shown in both Section 4.4 and Section 5.3.5 to have a marked effect on the predictive capability of the model. Equation (4.26) showed how this parameter affects the residual of the tree marginal which is is used to compare trees. The tree model in general is sensitive to this parameters so only a small range has been used in the experiments.

5.4.2.3 Demonstrations

This demonstration will be for 10, 25, 50 and 100 trees. Not all the graphs will be shown because with a large number of trees it is difficult to see what is going on in a graphical sense. Ensembles will be compared in two ways: the first is to show the accuracy of the ensemble estimates and predictions as a function of the number of trees and the second is to assess if the number of trees helps in bringing the model closer to achieving the target distribution.

Before this, however, an example of the ensemble using ten trees is provided. Also provided are a table of the leaf parameters generated according to the method described above (Table 5.4.2) and a table showing the functions that manipulate the leaf parameters and their arguments (Table 5.4.1).

Functions and arguments for adpating parameters.										
move	fname	func	argn1	argv1	argn2	argv2	argn3	argv3	argn4	argv4
grow	Z0	runif	n	1	min	-40.00	max	10.00		
	W0	runif	n	1	min	1.00	max	15.00		
	V	Vfunc	ln	2	scale	0.25	min	0.10		
	W	IVfunc	ln	2	scale	0.25	max	15.00		
	F	Ffunc	ln	2	scale	0.15	sn	1.00	min	0.05
	H	Vfunc	ln	2	scale	0.03	min	0.85		
prune	Z0	runif	n	1	min	-40.00	max	10.00		
	W0	runif	n	1	min	1.00	max	15.00		
	V	IVfunc	ln	2	scale	0.01	max	5.00		
	W	Vfunc	ln	2	scale	0.15	min	3.00		
	F	IFfunc	ln	2	scale	0.01	sn	1.00	max	0.95
	H	IVfunc	ln	2	scale	0.03	max	1.15		

Table 5.4.1: Functions that modify the parameters as the trees change shape (grow or prune, grow-shift or prune-shift) for the CGM model

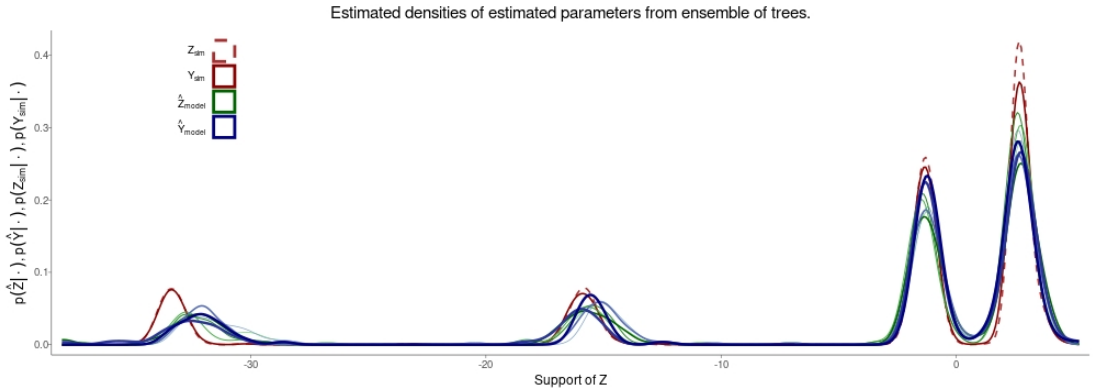
Parameter values, SNR and H/V for 10 leaf tree.												
LN	PN	0	1	2	3	4	5	6	7	8	9	10
2	V	1.00	0.250	0.444	0.639	0.833	1.028	1.222	1.417	1.611	1.806	2.000
	W	20.00	10.000	11.111	12.222	13.333	14.444	15.556	16.667	17.778	18.889	20.000
	F	0.85	-0.400	0.450	-0.500	0.550	-0.600	0.650	-0.700	-0.750	-0.800	-0.850
	H	1.00	0.900	0.900	0.917	0.950	0.983	1.017	1.050	1.083	1.117	1.150
	Z0	-30.00	-30.556	0.556	5.000	-35.000	5.000	-35.000	5.000	-12.778	-26.111	-17.222
	W0	15.00	6.111	5.000	12.778	5.000	7.222	13.889	7.222	9.444	8.333	10.556
	R	20.00	40.000	25.000	19.130	16.000	14.054	12.727	11.765	11.034	10.462	10.000
	H/V	1.00	3.600	2.025	1.435	1.140	0.957	0.832	0.741	0.672	0.618	0.575
3	V	1.00	0.250	0.444	0.639	0.833	1.028	1.222	1.417	1.611	1.806	2.000
	W	20.00	10.000	11.111	12.222	13.333	14.444	15.556	16.667	17.778	18.889	20.000
	F	0.85	-0.400	0.450	0.500	0.550	0.600	0.650	0.700	0.750	-0.800	0.850
	H	1.00	0.900	0.900	0.917	0.950	0.983	1.017	1.050	1.083	1.117	1.150
	Z0	-30.00	-3.889	-30.556	-35.000	-35.000	-21.667	-8.333	-21.667	-30.556	-26.111	-8.333
	W0	15.00	11.667	13.889	9.444	15.000	5.000	8.333	13.889	6.111	13.889	9.444
	R	20.00	40.000	25.000	19.130	16.000	14.054	12.727	11.765	11.034	10.462	10.000
	H/V	1.00	3.600	2.025	1.435	1.140	0.957	0.832	0.741	0.672	0.618	0.575

Table 5.4.2: Parameters for the initial weak learners of a 10 tree ensemble.

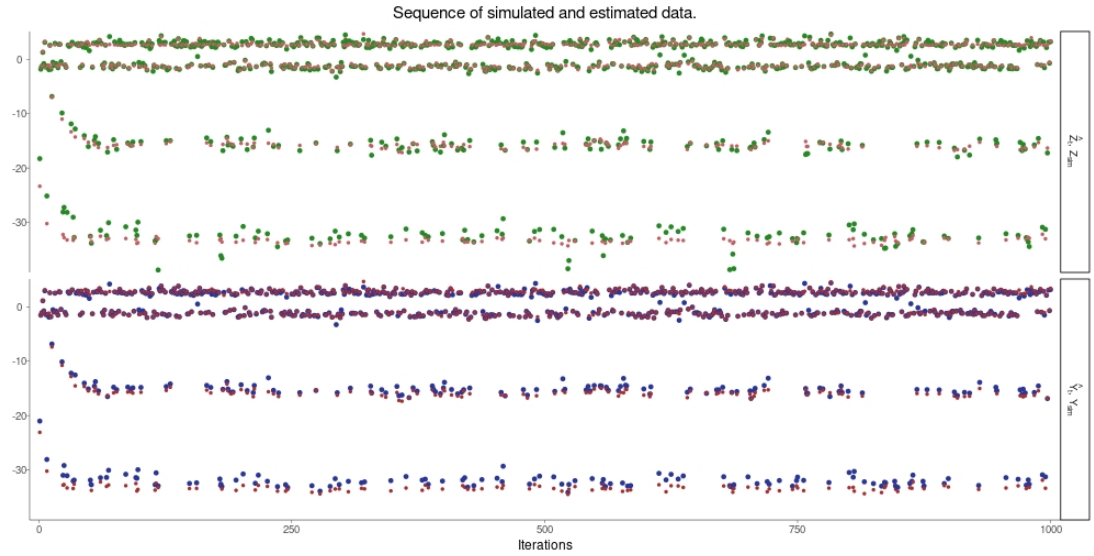
The first figure, Figure 5.4.1a, shows a density estimate of the sampled values. There are several lines because the intention is to show that the proposed method adapts to the "true" density as the number of iterations increases, although it does start out quite close. Examples of the density estimates were taken at $250, 500, 750$ and 1000 iterations and show the change in the density between these examples. For example, the lightest blue line shows the density of the predictions for the observations up to 250 iterations and the next light blue line shows the density for the same samples but from iteration 201 to iteration 500 and so on.

Figure 5.4.1a needs to be read in conjunction with Figure 5.4.1b where the estimated sequence is shown. It is clear that while the proposed algorithm gets close

to the simulated density it does not quite match the target density exactly. Some



(a) Density estimates based on simulated and estimated parameters for the CGM model using 100 tree ensemble.



(b) Estimates of the latent state and predictions for the observation sequence for the CGM model by a 10 leaf ensemble.

Figure 5.4.1

interesting observations are:

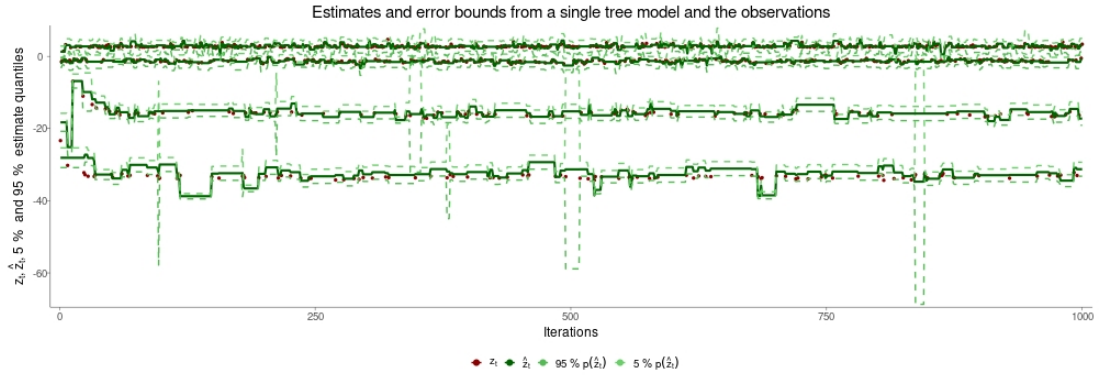
- The inability of the BTRS model to match the target density is similar to the behaviour of the Kalman filter. In Appendix A1.3 it was shown that different parameter settings for different Kalman filters produced vastly different results. In this model there are several parameters initially set and some functions provided that modify these parameters and the tree finds the the best set of parameters from which to produce estimates and predictions.

- The state estimation process is more adaptive than the predictive process. This is most clearly seen in fig. 5.4.1a where the state (in green) begins to intermingle more with the simulated state than the predictions for the

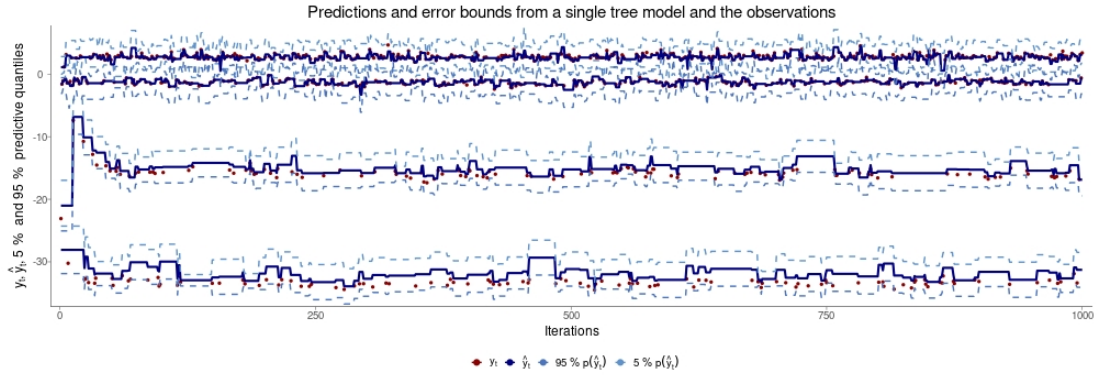
observations do. There seems to be a constant difference between the predictions and observations in the lower graph and this seems to persist once initial convergence in the first few iterations has been completed.

- There is more variability in the state estimates than in the predictions. This is likely due to the influence of the observation parameter,H, which is able to squeeze (scale) the predictions after the state parameter has provided the transition from the previous state to the current state. It seems that a correction of the available H parameters might cure this but after many experiments this effect persists. This will be explored more in Chapter 6.

The next figures, Figures 5.4.2a and 5.4.2b, show the sequences from Figure 5.4.1b but with the estimates and predicated values joined by lines and the piecewise 5% and 95% bounds shown by dashed lines[7]. It is clear that the bounds for the state estimate are tighter than the bounds for the predictions which is not obvious from the estimated densities in Figure 5.4.1a but consistent with theory.



(a) Estimates and pointwise error bounds for the level estimates.



(b) Predictions and pointwise error bounds for the observation stream.

Figure 5.4.2

The next figures will focus on the output from the ensemble of the components of

[7]The extreme lines shown in Figure 5.4.2a are due to unusual predictions and estimates which caused bound values to be grouped in the wrong set when graphing.

the tree marginal, the number of leaves in the tree and some other MCMC related output. Figure 5.4.3 shows, from top to bottom, the log of the non-normalised tree density; the number of leaves for each tree in the ensemble; the log of the tree marginal, the log of the densities of the estimated state and the log of the tree prior.

Perhaps the first thing to notice is that the colours, each one representing a different tree from 1 to 10, are changing for $\mathsf{p}(T \mid y^t, z^t, \cdot)$. This means that the MCMC algorithm is mixing well among all trees and that not one tree dominates the search of the model space. This mixing behaviour is essential to prevent degeneracy of the algorithm and is one reason why this approach might be preferred over sequential Monte Carlo (SMC) methods. It is also interesting because it highlights a difference between this approach and the approach of Chipman et al. (1998) because there the algorithm had to stopped and started repeatedly to avoid local degeneracy. The next thing to notice is the relationship between

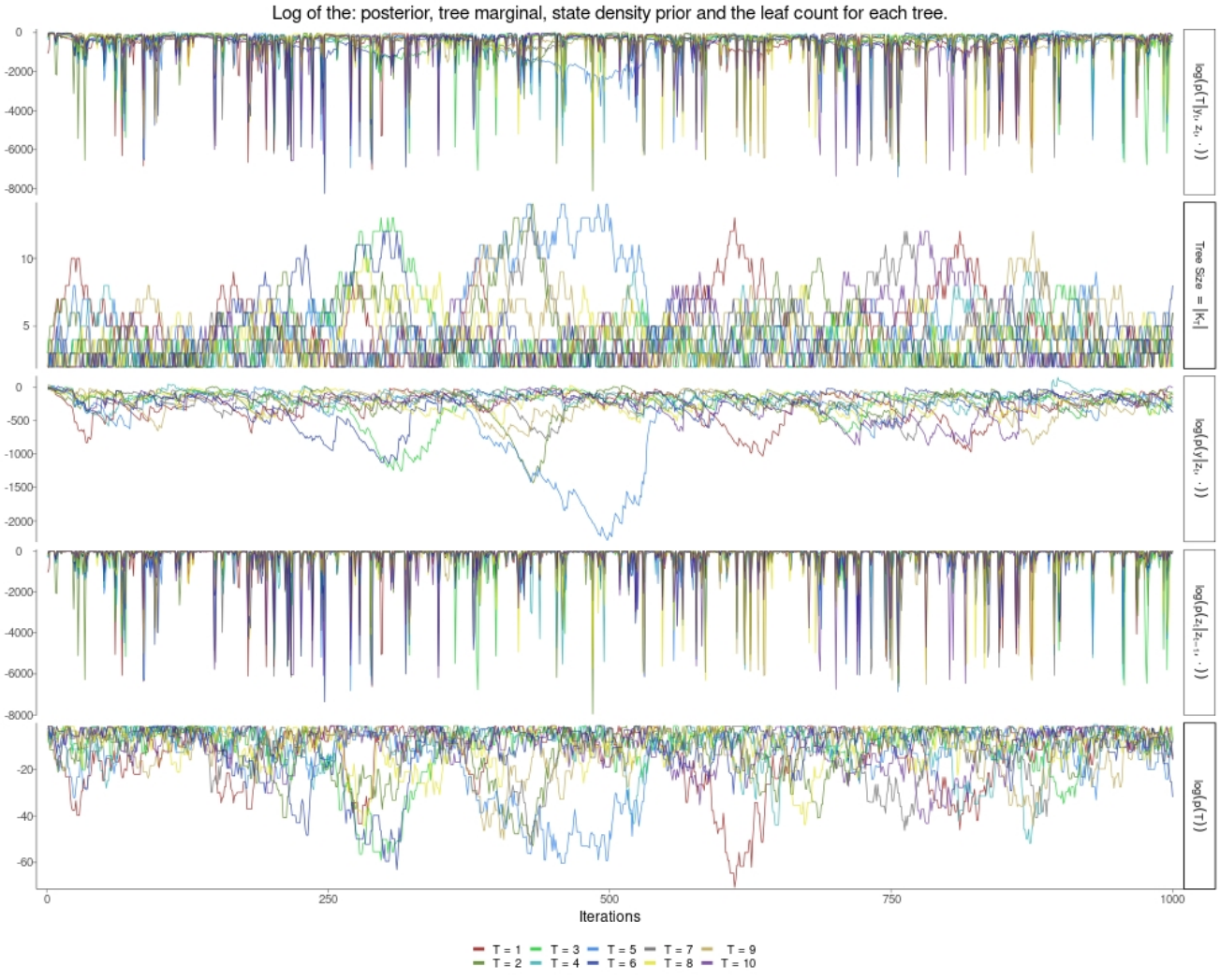


Figure 5.4.3: Components of the tree marginal and the number of leaves.

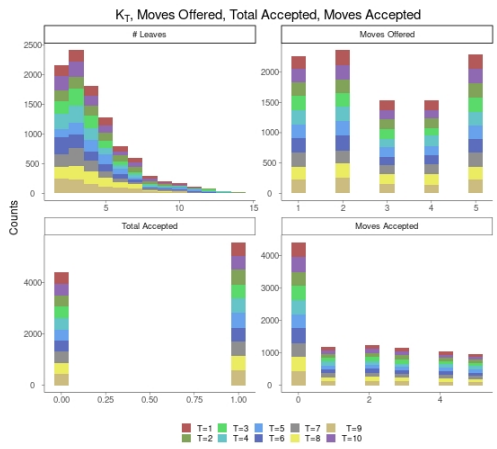
the different graphs. Focussing on $T = 5$, the light blue tree, one can see that as the number of leaves increases, the log of the tree density decreases, the tree marginal also decreases and the prior for the tree works in concert with these two components. This shows that the tree is learning and adapting to new data. This pattern is repeated by other trees over other iterations which shows that the ensemble is learning and adapting as the data changes and as the tree explores the model space but also that large trees are unlikely.

Finally, notice that there is a strong correspondence between the worst cases of the log of the tree density and the log of the state density. This shows that if a tree is estimating the state poorly this is reflected in the probability of the tree. What is is not clear is if a poorly estimating tree precipitates a change in the tree size or its evolution.

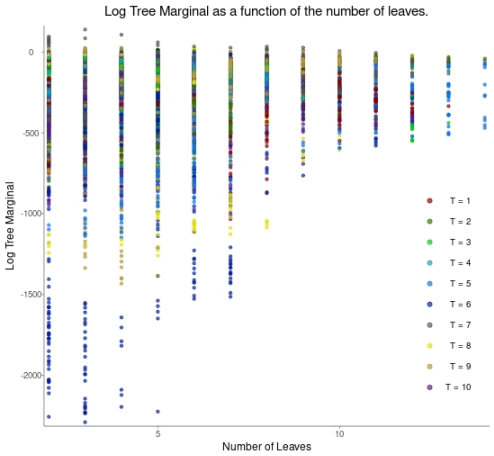
The next figures show some more information about the MCMC process. The top left graph of Figure 5.4.4a reflects the behaviour of the tree prior in the streaming setting. In Section 5.2.3 the effect of the prior in a streaming setting was shown but without taking into account the effect of the data. That is, the prior is meant to act as a penalty against ever expanding trees but it was not certain if the tree prior would be overwhelmed by the data. Figure 5.2.2 showed that for parameters $\xi_T = (\alpha = 0.95, \beta = 2)$ the expected number of leaves was between 3 and 4 for the duration of the experiment and Figure 5.4.4a confirms this. Thus, even in the streaming data setting where the analysis is not over a compact data set, the tree sampling prior process designed by Chipman et al. (1998) is an effective method of controlling tree growth which does not require the excessive tree sizes of other methods (Domingos and Hulten 2000; Hulten and Domingos 2003; Bifet and Gavaldà 2009) nor does it require the greedy search approach of Breiman et al. (1984).

The number of moves offered is a reflection of the fact that the prune and prune-shift moves can only be offered when tree has more than 2 leaves. The graph to the left shows that there are many iterations where all trees have only 2 leaves so it is reasonable that the number times that these two moves is offered is less than than the other moves.

The bottom left graph shows that the number of moves accepted is greater than the number of moves rejected. This must be true if the average number of leaves for the duration of the process is greater than 2. Thus it might be said that the tree is learning the "true" number of leaves. This is interesting because while the CGM model has 5 leaves, the posterior density has only 4 modes which would suggest that the number of filters actually needed to generate the data is 4 not 5.



(a) Number of visits to each leaf state, proportion of moves offered, proportions of moves accepted and proportions of different types of moves accepted according to tree colour.



(b) Taken from (Chipman et al. 1998), this plots hows the log integrated likelihood of the tree models as a function of the number of leaves.

Figure 5.4.4

The expected value for the distribution of the leaves is 5.33 but the most likely number of leaves is 3. Thus, a possible interpretation for the reason that more moves are accepted than rejected is that the trees are trying to find the average size of tree that best suites the data but the fact that the prior mean is in this same range makes this an uncertain conclusion.

The final graph of the 4 set shows that despite moves 1,2,5 (change, grow and grow-shift) being offered more times, the actual mixing of the moves is about uniform. Slightly more grow moves seem to be offered than prune moves but again this is possibly because the true number of leaves being sought is between 3 and 5 and not 2.

Figure 5.4.4b is an adaptation of the graph from Chipman et al. (1998). Each point on the graph represents a value for the tree marginal that is attained at each number of leaves. The colours again represent the different tree numbers 1 to 10. Interestingly, a tree with a few number of leaves can have a large tree marginal and a tree with large number of leaves can have a low tree marginal but these are exceptional cases and, if one compares the trees by colour between this graph and Figure 5.4.3 it is likely that one excursion of the tree has produced this range of values. In general, as the number of leaves increases the range of the tree marginal deceases. Note that this is different to the example shown in "Bayesian CART model search" because there the final results over 10 experiments are shown ans here all results of the evolution of the tree are shown.

Before presenting some numerical comparisons the final graph shows the cumu-

lative CMSE between the 10, 25, 50 and 100 tree ensemble demonstrations. As

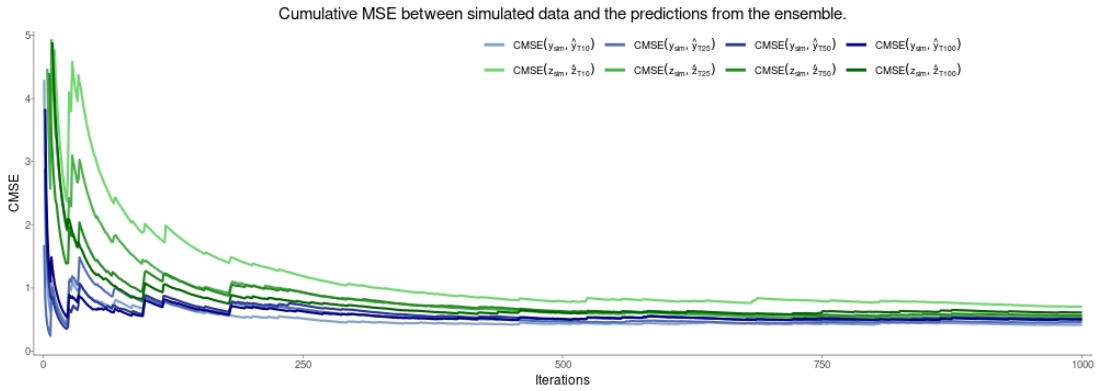


Figure 5.4.5: The CMSE between the simulated state and the estimated state as well as simulated observations and the predictions.

the key shows, the green lines represent the estimate CMSE and the blue lines the prediction CMSE. Interestingly, while it is clear that the prediction CMSE is lower in general for all ensemble sizes, there does not seem to be much differentiation between different ensemble sizes. In fact, if one compares the single tree from Figure 5.3.11 to these ensembles there seems to be little difference between a single tree and the ensemble. This is quite likely due to the fact that the Kalman filter is an optimal estimator and once a tree is estimating well it may well continue to do so.

Less optimistically it may mean that a single leaf is being chosen repeatedly and so the Kalman filter at one leaf is being updated repeatedly. A few things suggest that this is not the case: firstly Figure 5.3.12 shows that tree sizes are varied enough and change rapidly enough so that a single leaf is unlikely to remain chosen repeatedly. (i.e. the chains mix well) and Figure 5.4.6 shows that the range of leaves chosen (in $\log_2$ scale) by the tree to estimate and hence predict vary enough so that one leaf is not being repeatedly updated.

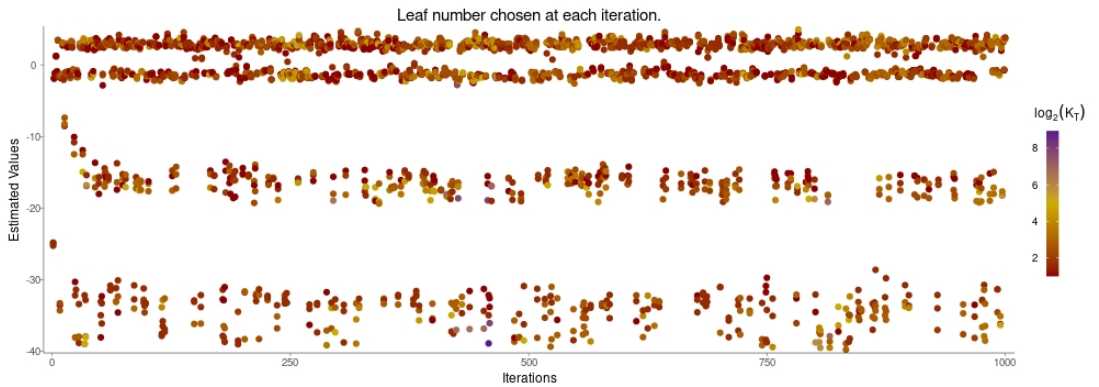


Figure 5.4.6: The $\log_2(K_t)$ of the chosen leaves for three trees: $1, 5, 10$

The final part of this section will present a numerical comparison between the

different ensembles. The first table will present a series ov values for the CMSE and their differences at different iteration numbers. The second table will compare the distributions using the Kullback-Leibler Divergence and the Wasserstein metric.

The Kullback-Leibler divergence also called the relative entropy is way for telling how close two distributions are. It is not a formal distance measure but is often used in statistics. The measure is defined as:

$$\mathcal{D}_{KL}(\mathsf{P} \parallel \mathsf{Q}) = \int_{\mathcal{Y}} \mathsf{p}\,(\,y\,) \log\left(\frac{\mathsf{p}\,(\,y\,)}{\mathsf{q}(y)}\right) \mu(dy) \tag{5.26}$$

but to use it over the discrete samples requires summing the integral over the empirical distributions obtained from the data generated Y_{sim} or Z_{sim} for Q and from the simulated Y_{BTRS} or Z_{BTRS} for P.

The Wasserstein distance, also known as the earth movers distance, is a true distance measure and ideally suited to statistical measurement because the result can be interpreted as the cost of transforming the proposed measure into the base measure. This measure is defined as:

$$\mathcal{W}_p(\mathsf{P},\, \mathsf{Q}) = \left(\frac{1}{n}\sum_{i=1}^{n} \left\|Y_{(i)} - X_{(i)}\right\|^p\right)^{1/p} \tag{5.27}$$

where $Y_{(i)}$ means the i^{th} order statistic (the data must be sorted) and in this case $p = 2$ means that the Euclidean norm is being used.

KL = Kullback-Liebler, WS=Wasserstein								
Dist.	$\mathcal{F} = 10, Y$	$\mathcal{F} = 10, Z$	$\mathcal{F} = 25, Y$	$\mathcal{F} = 25, Z$	$\mathcal{F} = 50, Y$	$\mathcal{F} = 50, Z$	$\mathcal{F} = 100, Y$	$\mathcal{F} = 100, Z$
KL	83.5303	94.5762	93.2218	86.0637	90.2173	86.1748	94.2132	93.7195
WS	0.5556	0.6646	0.5840	0.5655	0.6063	0.5732	0.6080	0.6145

Table 5.4.3: Two measures of the differences between the the proposed distribution and the generated data.

The results in Table 5.4.3 are perhaps, in one way, disappointing because the increase in the number of trees should make the model more accurate. However, as has already been discussed the Kalman filter is an optimal filter and already producing estimates an predictions with a low MSE/CMSE. Also, the tree has been shown to get close to the target posterior when new data from a fixed distribution is being provided. Perhaps the ensemble will help capture target distributions that changing with time?

The final table confirms what was presented in Figure 5.4.5. These figures show that at the given iterations the CMSE is already quite low. Increasing the number

of trees seems to make the CMSE slightly worse, at least in the case of this example.

CMSE at Iterations 250, 500, 750, 1000								
Iteration	$\mathcal{F} = 10, Y$	$\mathcal{F} = 10, Z$	$\mathcal{F} = 25, Y$	$\mathcal{F} = 25, Z$	$\mathcal{F} = 50, Y$	$\mathcal{F} = 50, Z$	$\mathcal{F} = 100, Y$	$\mathcal{F} = 100, Z$
250	0.5176	1.1876	0.6643	0.9284	0.7278	0.9316	0.6303	0.7765
500	0.4368	0.7901	0.4715	0.6084	0.5069	0.6332	0.5097	0.6106
750	0.4344	0.7929	0.4637	0.5474	0.5030	0.5780	0.5200	0.6325
1000	0.4186	0.7034	0.4596	0.5313	0.5114	0.5643	0.4992	0.6110

5.4.3 Ensemble Model Comparison

5.4.3.1 Introduction

The purpose of this section is to compare some well known Bayesian tree regression methods with the proposed model. The methods that will be used are Dynamic Trees (DT) (Taddy et al. 2011) and Bayesian Additive Regression Trees (BART) (Chipman et al. 2010). These approaches to regression are known to converge to the target distribution but use different methods to achieve this goal. DT is a sequential Monte Carlo (SMC) or particle filtering method that has already been adapted to the streaming data setting (Anagnostopoulos and Gramacy 2013). BART is not a streaming method but uses a form of Markov chain Monte Carlo (MCMC) based on Bayesian back-fitting (Hastie and Tibshirani 2000) to perform tree based regression and it is the descendant of the CGM model (Chipman et al. 1998) that has formed much of the basis for this thesis.

The substance of both of these approaches has been covered Chapter 2 and, with particular respect to SMC, in Section 3.3.3 but a brief summary of each will begin each of the subsections that deal with the respective comparisons: DT and BART versus the proposed model (BTRS). The differences between the proposed approach and these methods will be highlighted and, where relevant, weaknesses in both the proposed and existing approaches will be pointed out.

Following the comparison of approaches is a section that compares the results of the modelling data generated by the CGM model. For the most part the comparison is graphical because it is necessary to consider the entire length of the stream and not only the end result. Where sensible some comparative numbers have been provided.

5.4.3.2 Dynamic trees comparison

DT is a particle filtering method, at heart a state-space model, that uses trees as particles to produce predictions based on models at the leaf nodes of trees

that have been chosen at each t by x_t. This cloud of trees, recommended by the authors to consists of 1000 particles, produces posterior weighted predicted values from local leaf models that are then averaged over the particle set so that a mean result forms the predicted value for observation y_t at each t.

This model does not produce an estimate of the state process and, with a focus on an automatic regression, marginalises out the leaf model parameters, μ_η and σ_η (η is some leaf node). The marginal likelihood for the data is available at each t by an approximation over all particles in the particle set.

To learn the particle sets their approach uses SMC but first resamples particles based on the predictive probability weight of the particle given the new observation. New trees are then propagated by creating three candidate trees from the moves stay, prune and grow at the node that has already been chosen by the data. Trees are equivalent above the parent of the chosen terminal node so posterior probabilities for the new tree are only calculated over subtrees rooted at the parent node. The new trees are propagated by sampling from the candidate trees proportional to the prior probability of each move and the marginal likelihood of the tree based on the new data. The statistics at the chosen node are then updated.

This method is different from the proposed method in the following ways:

- The marginal likelihood of the tree in the proposed (BTRS) model is taken over the latent state. Another way to see this is that at each t the BTRS model produces a product of averages over all conditional models where the average is with respect the previous latent processes $\mathsf{p}(z_{i-1,b})$. Thus the entire history of the tree and its data is taken into account.
- BTRS produces an estimate of the state level at each t and predictions for the observed process.
- Parameters are not marginalised over which might allow for additional parameter learning but this also creates a problem with parameter setting and model identification.
- Each chain in the ensemble produces a stochastic process which mirrors to some degree, the output of the Kalman filter. As shown in Section 4.5.1, there are parallels between the tree likelihood scale and residual parameters and the state variance and innovation of the Kalman filter. However, the DT model discards its particles and hence loses information about the process that it is learning.
- The MCMC method produces a chain of samples about the process under

study rather than just predictions for the state process. These samples, both for the state and observation processes can be used to derive many properties of the posterior distribution while it is unclear if the particle set that is available from the SMC algorithm provides the same amenity.

- Nodes chosen for evolution of the tree in the BTRS case are randomly chosen rather than chosen based on the data. Further, the proposed model provides a change move so that, it is hoped, preferred covariates can be recognised and variable selection learnt.
- The original DT model requires that there are minimum number of data points at each leaf before learning can begin and all the data that is received is stored in every particle, at least for as long as the particle exists. For a large enough data set this would cause computational issues. However, this has been partially resolved in Anagnostopoulos and Gramacy (2013) by retiring some of the data. However, this creates a window dependency for the particle set which might render any saved particle set for future analysis useless unless these retired particles are also saved. In the BTRS model the covariates are not used in the leaf models at all. The parameters are independent of covariate input so that no data, beyond the temporary storages of parameter values and the current data point, are stored in the model. The tree, including the bounds of the used covariates, stores all information necessary to sample the next tree. The output process via the tree marginal, like the Kalman filter, produces all the information about the streaming process without having to resort to additional, post-inference covariate and response manipulation.

In short, the proposed BTRS model is different to the DT model because it provides a much richer set of data, in itself a stream, that is a summary of the streams of data that can possibly be used to learn more about the processes under observation than that provided by the DT model.

However, the one unmistakeable advantage of the DT model is that its method of sampling, SMC, it is guaranteed to find to the correct posterior distribution at each t because at each t there is a large enough cloud of samples over which an ergodic average can be attained. Typically in SMC there are issues with degeneration of the sample cloud as all posterior probability tends to focus on a small number of samples, a consequence of importance sampling, but this does not appear to be the case in the DT model and there are solutions to this for, example, particle MCMC (Andrieu et al. 2010).

5.4.3.3 Bayesian additive regression trees comparison

Bayesian additive regression trees (BART) (Chipman et al. 2010) is a general additive model for regression using trees. It is a counter part to the classical approach of boosting using weak learners (Freund and Robert E Schapire 1997) for the Bayesian setting. It was developed by the same authors as Chipman et al. (1998) but focuses on regression although extension to this functionality are available.

BART provides a sum-of-trees estimate of the mean level at each of the tree's leaves where each estimate is shrunk in proportion to the ensemble size which is suggested to be 200 trees. Each estimate has Gaussian error that is again proportional to the ensemble size so that, as a whole, the estimates from the ensemble have additive error according to a single variance parameter σ^2.

BART uses a Bayesian backfitting algorithm, a form of Gibbs sampler (Hastie and Tibshirani 2000), that samples from successive draws of each of the trees. A residual is formed between each data observation y_t and, for every j^{th} tree, the sum of estimates of μ_t for y_t over all trees not including the j^{th} tree. This residual then takes the place of y_t in each tree marginal when each tree posterior is calculated. In this way the every tree shares information because every tree marginal contains the residual information from the ensemble.

BART provides set of MCMC samples from which one can find any desired function of the samples from the posterior distribution. BART has also been adapted to very wide data (large p) (Hernández et al. 2018) and for variable selection (Linero 2018). It is hoped that these adaptations of BART might be applicable to the BTRS model as it too provides a rich source of data that can be accessed by concurrent processes that act upon the output stream from the BTRS model.

The main differences between BART and BTRS are that:

- BART is not suitable for streaming analysis. It is a traditional MCMC algorithm in the sense that it requires multiple samples and multiple passes over the data to achieve stationarity.
- BART is a sum-of-trees model where estimates are shrunk in proportion to the ensemble size. The BTRS model is a mixture model where each estimate and prediction is weighted according to the posterior of the tree. The estimates and predictions are independent of the size of the ensemble other than the case where more trees means wider and faster exploration of the tree model sample space than fewer trees.
- BART shares information between all trees. This means that for each esti-

mate every tree must be visited by every other tree to calculate the residual. The BTRS model calculates a residual (innovation) at each chosen leaf based only on the data that exists in that leaf's latent variable prediction $\hat{\mu}_{t|t-1}$.

- While the marginal for the tree in BART is a function of the ensemble, the marginal for the tree in BTRS is a function of only the alternate models in that particular tree. Each tree in BTRS retains its independence making it embarrassingly parallel and amenable to concurrent computation.
- BART, like Chipman et al. (1998) (BCARTMS), requires that all data is stored with the model and that there are a minimum number of data points in each leaf before inference can be performed. The models in the leaves of BART also require that the covariates be used for regression calculations. As mentioned in Section 5.4.3.2, the BTRS model has none of these requirements but does require that the bounds of each covariate are provided.
- BART has a single variance parameter for all tree models so that the error for each estimate is Gaussian and homogenous. In the BTRS model one is able to specify a number of different variance, transition and observation parameters which offers great flexibility in modelling but at the price of complicated model design and specification.

BART is a successful modelling approach that is part of the inspiration behind the proposed approach to streaming regression in this document. While primarily a model for estimation, prediction can be performed by using a training set to learn an ensemble of trees and then predictions can be obtained from an out-of-sample test set.

One thing that neither of the above approaches, DT and BART, seem to offer is flexibility in the dimension of the observation. The Kalman filter and its variants are able to model any $Y \in \mathbb{R}^n$, $Z \in \mathbb{R}^m$ but with a consequential cost on performance. If, however, the components of $Y \in \mathbb{R}$ are judged to be independent the BTRS model offers a method for dimension reduction, in a similar way to the ensemble Kalman filter (Evensen and Leeuwen 2000). Both of the other other methods offer different model types at the leaves. The BTRS model can offer many different versions of the Kalman filter to deal with nonlinearity and there is no reason other models cannot be adapted to dynamic modelling as has been done by Harrison and West (1999) if it makes sense to do so.

5.4.3.4 Model comparison

The rest of this subsection will show how the proposed model, BTRS, can get close to, if not match the BART and DT models in finding the target posterior. MCMC theory precludes actually achieving the target posterior at each t because for each new data point the joint distribution for the data changes thereby invalidating any stationarity assumptions. However it will be shown, between this chapter and the next, that it is possible to get close enough to the true posterior at each t so that the difference between the achieved posterior and "true" posterior is negligible.

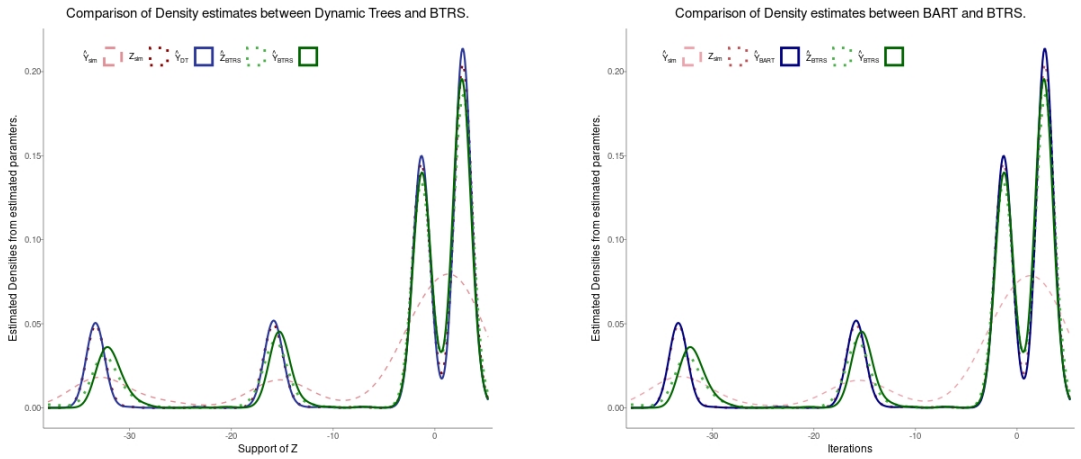
In their original formats, both BART and DT rely on the generation of a set of trees from some existing data (a training set) and then make some predictions based on new data (the test set). The proposed BTRS model performs estimation and prediction for the latent state process Z^t and the observed process Y^t at each index t without having been trained. That is, the BTRS model only sees the data once. To compare the proposed streaming model with the comparative models requires that two approaches are taken. The first approach is to assume that the data is fixed at size N. Then the original forms of the BART and DT models can be used against the proposed model with the disadvantage to the proposed model that it does not get to be trained nor does it get to revisit any of the data. The purpose of this comparison is that both BART and DT are known to converge to the true posterior distribution. Their respective MCMC (BART) and SMC (DT) methods hold and the proposed BTRS method of learning the model and outputting estimates on-the-fly can be tested against these.

The second approach involves creating a streaming-like setting This requires adapting the BART model where the DT model has already been adapted (Anagnostopoulos and Gramacy 2013). Adapting BART is done by allowing the model to estimate up to the current data point, make a prediction given a new covariates x_{t+1} and then to be provided with y_{t+1} so that the estimate up to the new data can be made. This is repeated 1000 times on new data after training the model. This modified BART model, along with the streaming DT model, are given a training set of data and then they perform prediction and re-learning on the test set. The proposed BTRS model only gets to see the test set.

The first figures, Figures 5.4.7a and 5.4.7b compare the estimated densities between the DT, BART and BTRS respectively. It is immediately apparent that the proposed model does not quite find the proposed distribution. As this BTRS model is the same as that used in Section 5.4.2.3 (as is the data) the reasons for not quite getting to the target distribution are discussed there. However, it is

worth noting that in these first examples the DT and BART models were run several times over the data and the predictions formed here are within sample predictions. In other words these predictions are not over a test set but over the training set.

The BTRS model only saw the data once, in sequence, and beyond the generalised parameter setting method described above, searched and learnt the models and made these predictions on-the-fly.



(a) DT and BTRS (b) BART and BTRS

Figure 5.4.7: Comparison of estimated densities between Dynamic Tree model (DT), the Bayesian additive regression tree model (BART) and the proposed streaming regression model (BTRS)

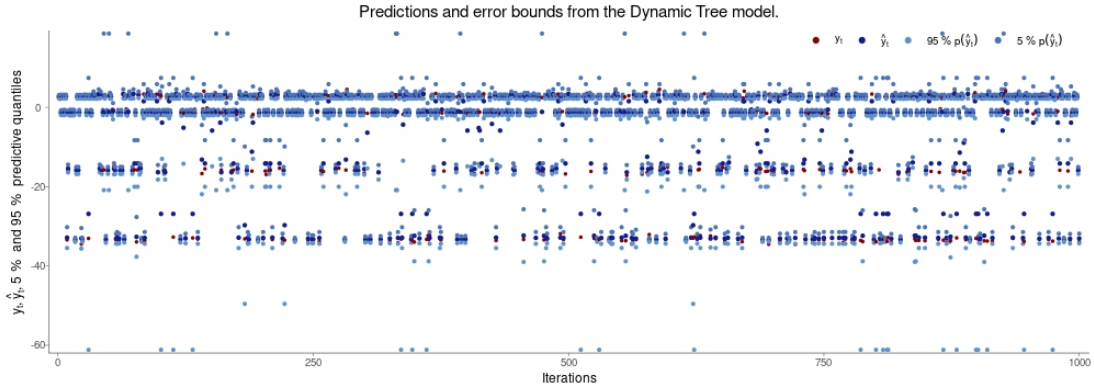
Figures 5.4.8a and 5.4.8b show the sequences of data that produced the above densities and these includes their point wise 5% and 95% error bounds. The bounds of the DT model seem tighter than that of the BART model.

Figures 5.4.9a and 5.4.9b show the comparison between the BTRS model and the streaming versions of both the BART and DT models.

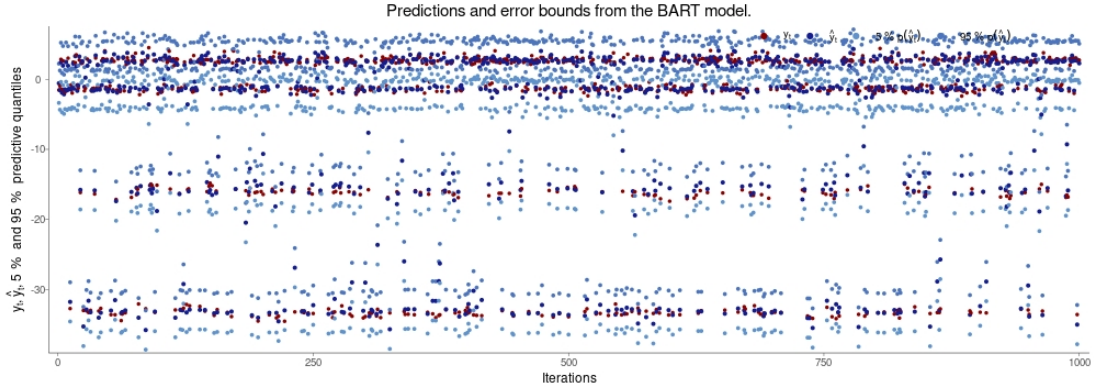
Figure 5.4.11 shows a comparison between the CMSE of all of the above models. It is clear that the BTRS model, while not quite finding the target distribution is clearly outperforming all versions of the above models when it comes to accuracy. The trade-off seems clear: accuracy over correctness. However, the next chapter will consider some simple modifications that will hopefully bring the BTRS model closer to the target distribution.

5.5 Summary and Conclusion

The aim of this chapter was to extend the tree filter model to one that can train, estimate, predict and learn on-the-fly. Markov chain Monte Carlo was used to

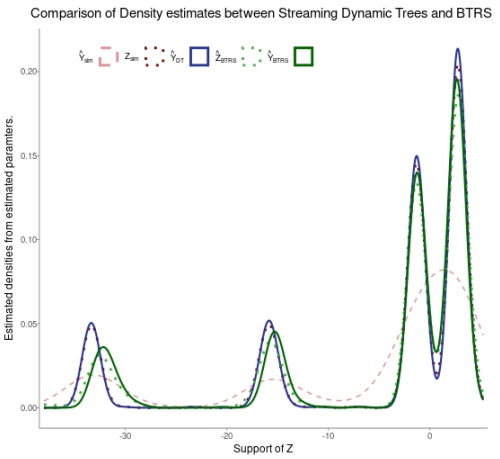


(a) Predicted values and their pointwise error bounds from the DT model.

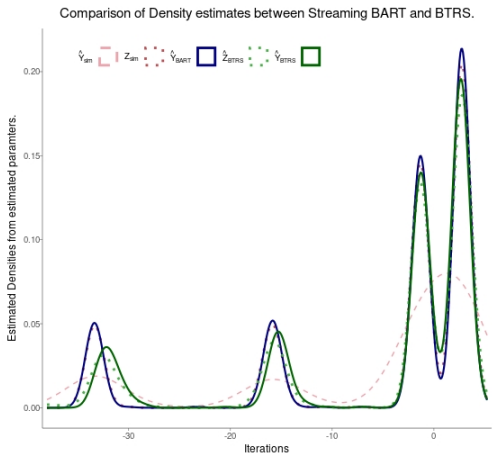


(b) Predicted values and their pointwise error bounds from the BART model.

Figure 5.4.8



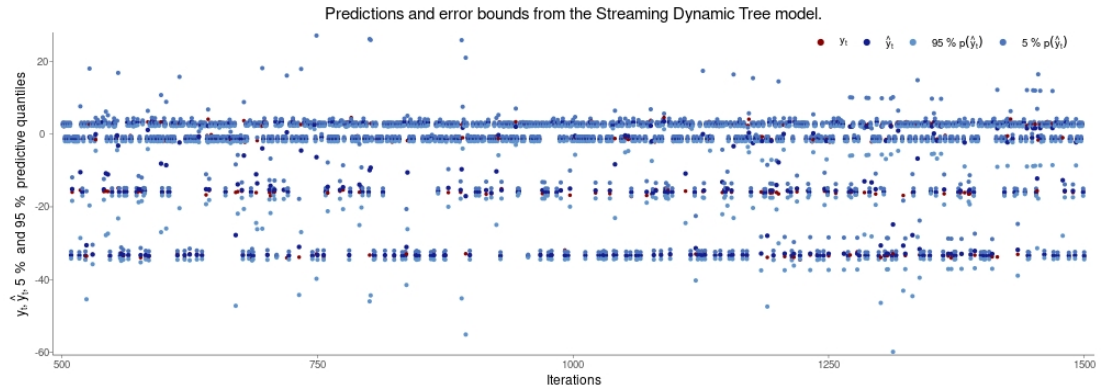
(a) sDT and BTRS



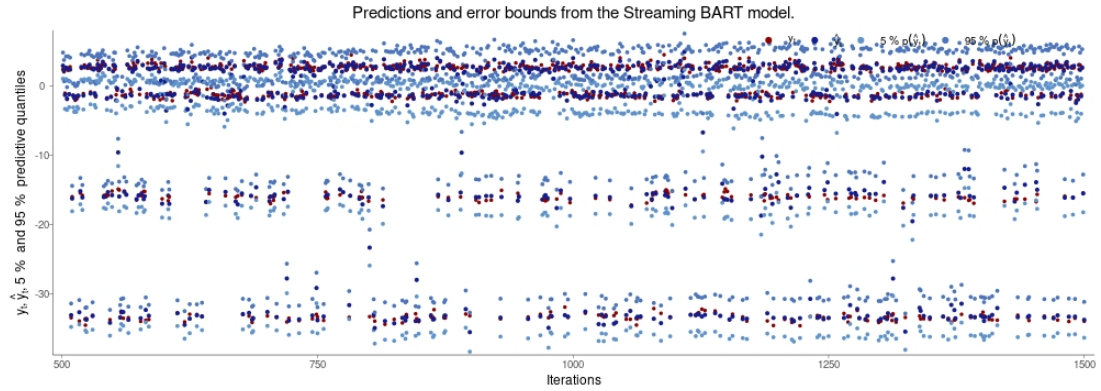
(b) sBART and BTRS

Figure 5.4.9: Comparison of estimated densities between streaming Dynamic Tree model (sDT), (streaming) Bayesian additive regression tree model (sBART) and the proposed streaming regression model (BTRS)

sample alternate tree models and this was combined with a deterministic method of adapting leaf filter parameters. The proposed model was compared with two state-of-the-art Bayesian approaches to tree modelling using simulated data sets. A real-world example has not been provided which limits an assessment of the



(a) Predicted values and their pointwise error bounds from the sDT model.



(b) Predicted values and their pointwise error bounds from the sBART model.

Figure 5.4.10

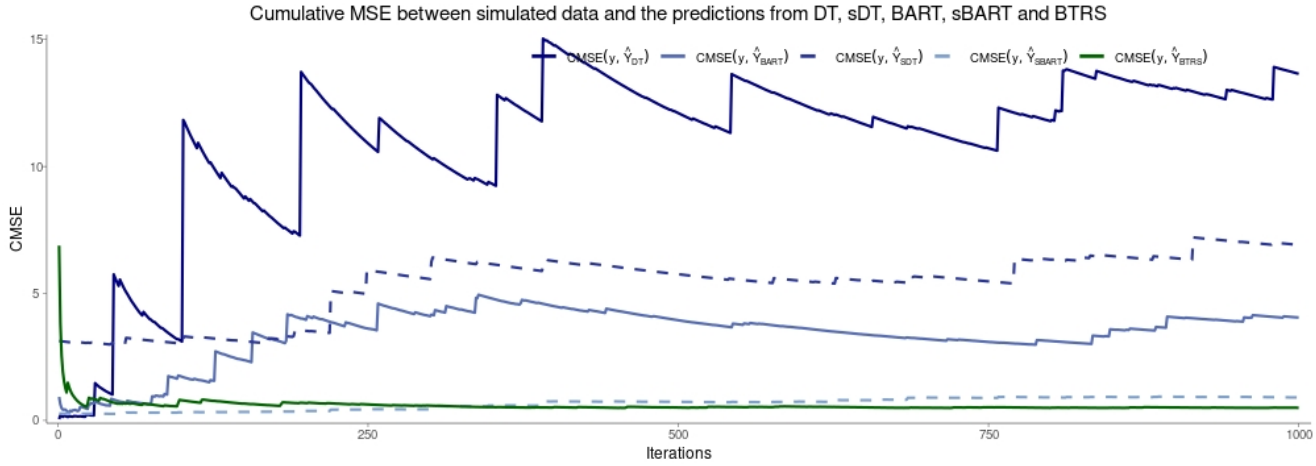


Figure 5.4.11: A comparison of the cumulative mean square error between the BTRS model, the DT and sDT models as well as the BART and sBART models.

model's applicability. Also not shown but demonstrated in preliminary studies was the performance of the model in $\mathsf{O}\left(|\mathcal{F}|K_T m^3\right)$. That is, the performance of the model is a constant in the model formulation and is independent of the input data stream.

The next section looks at some modifications to the proposed approach and shows an experiment using the Mackey-Glass equation.

6 Model Modifications

6.1 Introduction

This brief section will explore two modifications to the approach of the previous section. These modifications have not been thoroughly tested and offer demonstrations of methods that might improve the convergence of the streaming model to the target posterior.

The first modification is to use only the change, grow and prune moves. The second modification is to allow for a small number of additional evolution steps per iteration.

After these two modifications have been demonstrated and briefly explored a final example of a nonstationary data set will be presented. This data is generated by the Mackey-Glass differential equation. It is highly nonlinear and nonstationary and some demonstrations will show how this model copes with this type of data.

6.2 Modifications

6.2.1 Modification to Moves

The first modification that is explored is a return to an attempt at streaming modelling that was an earlier part of the research. The modification is to use only the change, grow and prune moves as proposal moves rather than all the moves that were used in the previous chapters. The reason for this is that the grow-shift and prune-shift are moves that seem to change the location or direction of the chain jump quite radically. These changes mean that it takes a while for the chain to resettle and find local modes to explore and this makes the dependence on the parameter modification functions stronger. Removing these moves suggests that changes are less severe so that the trees can learn the streaming process and adapt to new data in a slower fashion. This may come at a cost to performance and convergence in more extreme cases of nonstationarity.

The leaf parameters used for these experiments were exactly those used for the experiments in Section 5.4.2.3. They are reproduced in Table 6.2.1 for convenience. Only 10 trees were used in these experiments because if there are gains in increasing the number of trees then, for the CGM model that is used here, these might only be slight.

Parameter values, SNR and H/V for 10 tree ensemble.												
LN	PN	0	1	2	3	4	5	6	7	8	9	10
2	V	1.00	0.250	0.444	0.639	0.833	1.028	1.222	1.417	1.611	1.806	2.000
	W	20.00	10.000	11.111	12.222	13.333	14.444	15.556	16.667	17.778	18.889	20.000
	F	0.85	-0.400	0.450	-0.500	0.550	-0.600	0.650	-0.700	-0.750	-0.800	-0.850
	H	1.00	0.900	0.900	0.917	0.950	0.983	1.017	1.050	1.083	1.117	1.150
	Z0	-30.00	-30.556	0.556	5.000	-35.000	5.000	-35.000	5.000	-12.778	-26.111	-17.222
	W0	15.00	6.111	5.000	12.778	5.000	7.222	13.889	7.222	9.444	8.333	10.556
	R	20.00	40.000	25.000	19.130	16.000	14.054	12.727	11.765	11.034	10.462	10.000
	H/V	1.00	3.600	2.025	1.435	1.140	0.957	0.832	0.741	0.672	0.618	0.575
3	V	1.00	0.250	0.444	0.639	0.833	1.028	1.222	1.417	1.611	1.806	2.000
	W	20.00	10.000	11.111	12.222	13.333	14.444	15.556	16.667	17.778	18.889	20.000
	F	0.85	-0.400	0.450	0.500	0.550	0.600	0.650	0.700	0.750	-0.800	0.850
	H	1.00	0.900	0.900	0.917	0.950	0.983	1.017	1.050	1.083	1.117	1.150
	Z0	-30.00	-3.889	-30.556	-35.000	-35.000	-21.667	-8.333	-21.667	-30.556	-26.111	-8.333
	W0	15.00	11.667	13.889	9.444	15.000	5.000	8.333	13.889	6.111	13.889	9.444
	R	20.00	40.000	25.000	19.130	16.000	14.054	12.727	11.765	11.034	10.462	10.000
	H/V	1.00	3.600	2.025	1.435	1.140	0.957	0.832	0.741	0.672	0.618	0.575

Table 6.2.1: Initial leaf parameters for the 10 tree ensemble.

The functions that modify the leaf parameters as the trees change have been modified for each of the experiments. Tables 6.2.2 to 6.2.5 show the function settings for each of the 4 different experiments conducted on the model with only the change, grow and prune moves.

Functions and arguments for adapting parameters, Experiment 1.										
move	fname	func	argn1	argv1	argn2	argv2	argn3	argv3	argn4	argv4
grow	Z0	runif	n	1	min	-40.00	max	10.00		
	W0	runif	n	1	min	1.00	max	15.00		
	V	Vfunc	ln	2	scale	0.25	min	0.10		
	W	IVfunc	ln	2	scale	0.25	max	15.00		
	F	Ffunc	ln	2	scale	0.15	sn	1.00	min	0.05
	H	Vfunc	ln	2	scale	0.03	min	0.85		
prune	Z0	runif	n	1	min	-40.00	max	10.00		
	W0	runif	n	1	min	1.00	max	15.00		
	V	IVfunc	ln	2	scale	0.01	max	5.00		
	W	Vfunc	ln	2	scale	0.15	min	3.00		
	F	IFfunc	ln	2	scale	0.01	sn	1.00	max	0.95
	H	IVfunc	ln	2	scale	0.03	max	1.15		

Table 6.2.2: Function settings that exactly match those of Section 5.4.2.3.

As already mentioned, in Table 6.2.2 for Experiment 1, the parameters are no different to the parameters for the CGM model in Chapter 5. The aim here is

to show what the consequences might be if the model were simplified to only use these 3 moves.

Table 6.2.3 is Experiment 2. Here the idea is see if the the function choice and behaviour can also be simplified. Ideally one wants to have to specify as little as possible so that the ensemble learns the process. This has been done by removing all parameter modification functions except those that modify V_b, H_b because from previous investigations it was shown that these parameters are essential for maintaining tree learning and the signal-to-noise ratio as the tree changes.

Functions and arguments for adapting parameters, Experiment 2.										
move	fname	func	argn1	argv1	argn2	argv2	argn3	argv3	argn4	argv4
grow	Z0	constF								
	W0	constF								
	V	Vfunc	ln	2	scale	0.25	min	0.10		
	W	constF								
	F	constF								
	H	Vfunc	ln	2	scale	0.03	min	0.95		
prune	Z0	constF								
	W0	constF								
	V	IVfunc	ln	2	scale	0.25	max	5.00		
	W	constF								
	F	constF								
	H	IVfunc	ln	2	scale	0.03	max	1.05		

Table 6.2.3: Function settings for Experiment 1 where only the noise variance parameter and observation matrix are modified as the tree changes.

Experiment 3 reintroduces the functions that modify the state parameters F_b, W_b. This has been does to attempt to get some idea of the interaction effects between parameters that is specific to this modelling approach rather than the more general relationships that were shown in Section 5.3.1.

Functions and arguments for adapting parameters, Experiment 3.										
move	fname	func	argn1	argv1	argn2	argv2	argn3	argv3	argn4	argv4
grow	Z0	constF								
	W0	constF								
	V	Vfunc	ln	2	scale	0.25	min	0.10		
	W	IVfunc	ln	2	scale	0.10	max	15.00		
	F	Ffunc	ln	2	scale	0.10	sn	1.00	min	0.05
	H	Vfunc	ln	2	scale	0.03	min	0.85		
prune	Z0	constF								
	W0	constF								
	V	IVfunc	ln	2	scale	0.10	max	5.00		
	W	Vfunc	ln	2	scale	0.10	min	3.00		
	F	IFfunc	ln	2	scale	0.01	sn	1.00	max	0.95
	H	IVfunc	ln	2	scale	0.03	max	1.15		

Table 6.2.4: Function settings for Experiment 3 where functions that modify F_b, W_b have been returned.

Experiment 4 continues Experiment 3 by modifying the scale parameter of the functions that affect F_b, W_b.

Functions and arguments for adapting parameters, Table 4										
move	fname	func	argn1	argv1	argn2	argv2	argn3	argv3	argn4	argv4
grow	Z0	constF								
	W0	constF								
	V	Vfunc	ln	2	scale	0.2500	min	0.10		
	W	IVfunc	ln	2	scale	0.0001	max	10.00		
	F	Ffunc	ln	2	scale	0.0001	sn	1.00	min	0.05
	H	Vfunc	ln	2	scale	0.0500	min	0.85		
prune	Z0	constF								
	W0	constF								
	V	IVfunc	ln	2	scale	0.2500	max	5.00		
	W	Vfunc	ln	2	scale	0.0001	min	1.00		
	F	IFfunc	ln	2	scale	0.0001	sn	1.00	max	0.95
	H	IVfunc	ln	2	scale	0.0500	max	1.15		

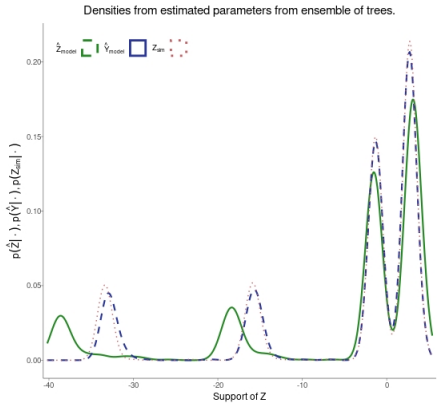
Table 6.2.5: Function settings for Experiment 4 where the scale parameters for functions that modify F_b, W_b have been decreased.

The first set of images, Figures 6.2.1a to 6.2.1d shows how the model approaches the density when only the change, grow and prune moves are used. These images should be compared to the images in Section 5.4.2.3.

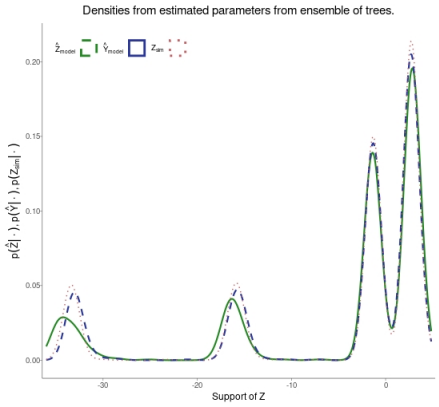
Figure 6.2.1a shows that there is a difference between modelling approaches and based on this image alone, one might assume that it is necessary to use all 5 moves. However, Figure 6.2.1b, which uses the function settings in Table 6.2.3, shows that, once the functions that affect F_b, W_b are removed, the model with only three tree moves could possibly perform better than the model with 5 moves. Figures 6.2.1c and 6.2.1d show that reintroducing the functions and the parameters based on Tables 6.2.4 and 6.2.5 weakens the estimation performance of the model. Figure 6.2.1d shows that there is even an effect on the prediction performance of the model as the scale parameters is further decreased.

Figures 6.2.2a to 6.2.2d show the sequences of estimates and predictions that correspond to the estimated densities in Figures 6.2.1a to 6.2.1d. There are some interesting things to notice in these sequences.

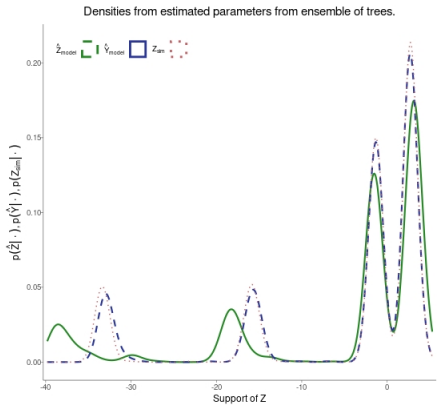
- The change in moves largely seems to affect the state estimates and not the predictions. In fact, it seems as though the predictions become more accurate with only 3 moves.
- Removing the functions that affect the state seem to show that the ensemble is now better at learning the state and also predicts well.
- Reintroducing the functions that modify the state parameters not only introduces a bias to the down side (or left) of the data but it also seems to



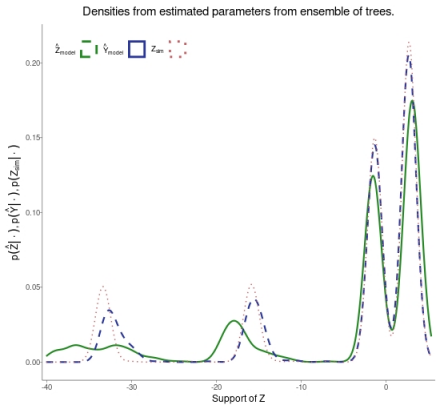
(a) Experiment 1.



(b) Experiment 2.

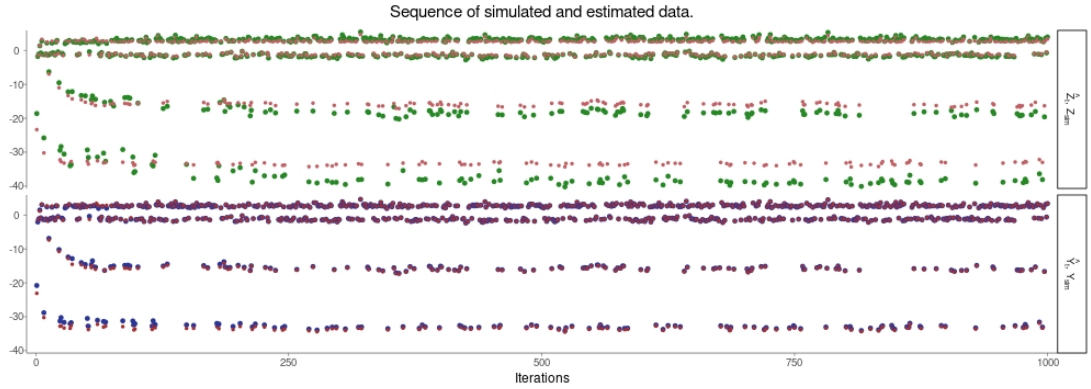


(c) Experiment 3.

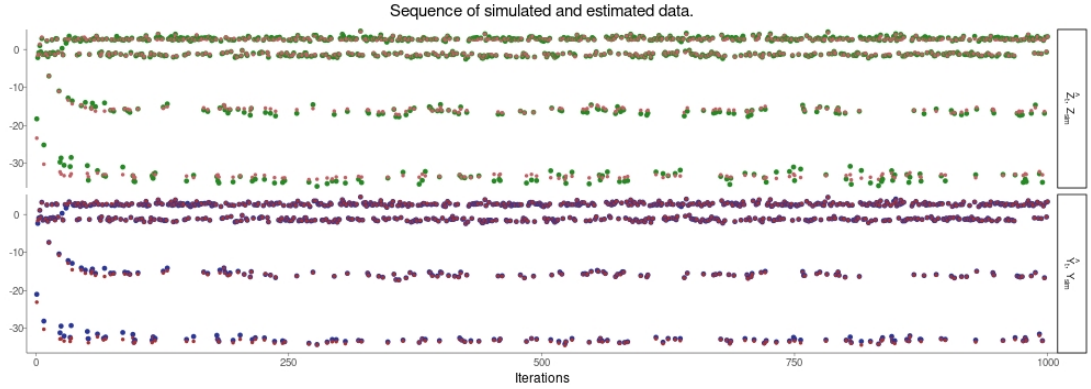


(d) Experiment 4.

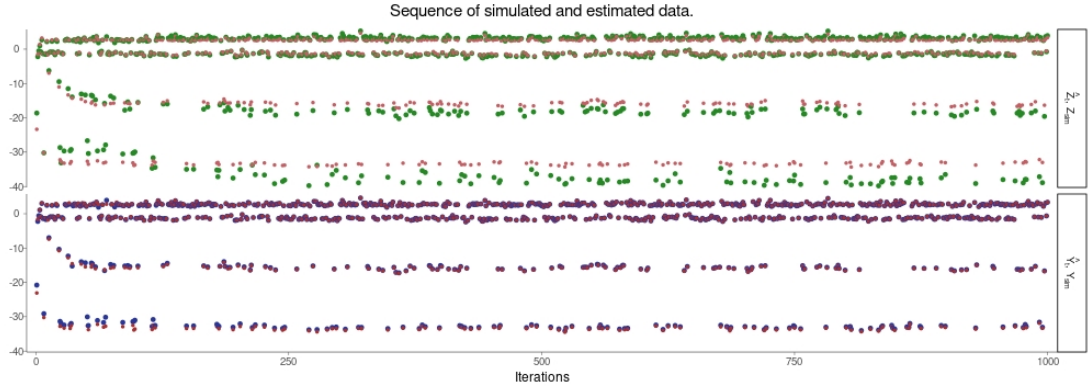
Figure 6.2.1: The estimated density when the only moves used are the change, grow and prune moves.



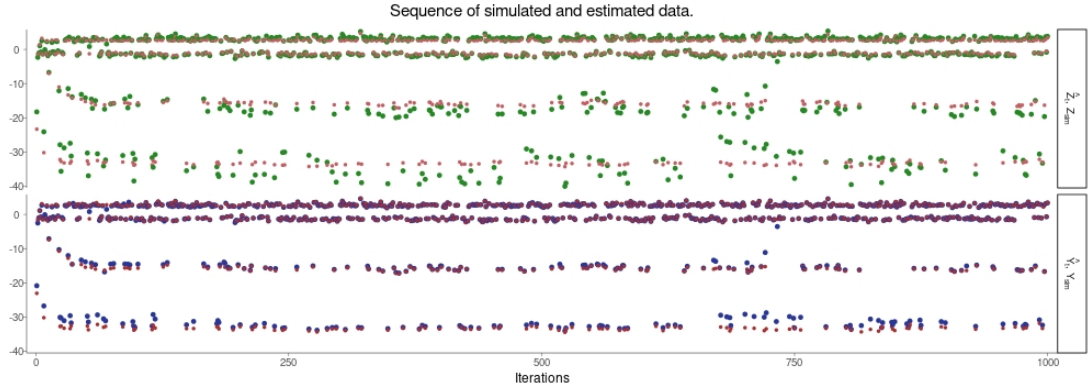
(a) Experiment 1.



(b) Experiment 2.



(c) Experiment 3.



(d) Experiment 4.

Figure 6.2.2: The sequence of estimates and predictions when only the change, grow and prune moves are used.

slow the convergence of the ensemble to the stream of data.

- Decreasing the scale parameter of the functions that modify F_b, W_b increases the variability of the estimates around the stream of data. Note that wrong or bad estimates seem to have more of an effect on the predictions in Figure 6.2.2d than in any of the other experiments.

The next section takes the same parameter and function settings as used in this section but now allows for M proposal moves per iteration. For comparison, the above parameters and the M proposal moves are repeated on models with 5 different types of moves.

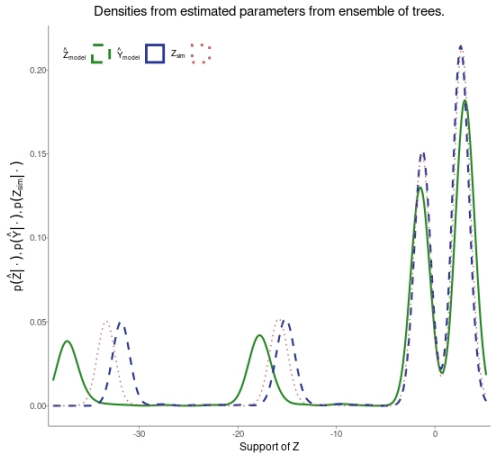
6.2.2 More proposals per Iteration

The next modification is intended to help with the convergence and accuracy. In this second modification each tree performs multiple evolution steps per data point. The idea behind this is to use each data point to locally explore the model space. In the usual MCMC approach several runs over the sample data would be performed so that averages of chains could be used get the ergodic summaries. It is generally assumed in that each Markov chain is independent and that, conditional on the each previous sample, the current sample is independent of all other sample points. Using a single data point but proposing several new trees for this data point provides an opportunity for each data point to visit a slightly different model, conditionally on the tree and independent of the previous data point. In this way several samples of tree models are proposed and accepted or rejected in a similar way to running multiple sample chains over a single data set. The cost of these additional steps should be fairly minimal because the ensemble is able to be parallelised over the independent trees so multiple evolution steps per data point means that within tree resampling avoids the setup and take-down cost of sharing data between trees, for example when new data arrives or when the posterior of the ensemble must be calculated.

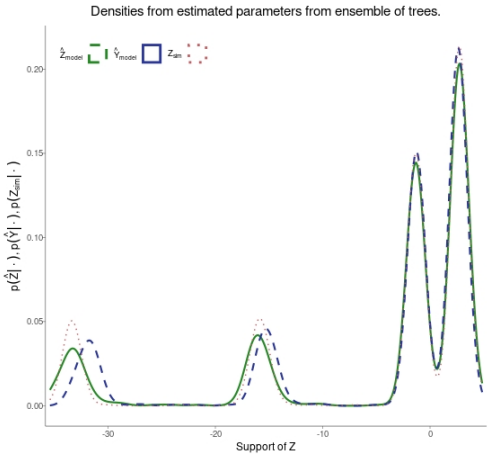
Another reason for this modification is that for the Kalman filter to converge it should be given the chance to move away from its initial value. This can only happen if the same filter is updated several times in a row which in general cannot be guaranteed in the random tree modelling case. Performing multiple samples per tree means that the probability that a leaf is updated is increased so that it is more likely that a filter can be iterated past its initial value. For all experiments that follow, the number of proposals per iteration is $M = 5$.

Figures 6.2.3a to 6.2.3d are the counterparts to Figures 6.2.1a to 6.2.1d and should be compared accordingly. A general effect to notice is that increasing the proposal moves "tightens" the estimates and the predictions by reducing the variability of the estimates. Figure 6.2.3b appears to be the best performing model for the CGM data so far. There seems to be a reduced discrepancy between the estimate density and the prediction density and the estimate density looks to be very accurate in terms of the location parameter which in general has been an issue. Figures 6.2.3c and 6.2.3d both look to have improved variability but do not have as good a location as Figure 6.2.3b with Experiment 3a, which has the largest scale parameters for W_b, F_b, faring the worst.

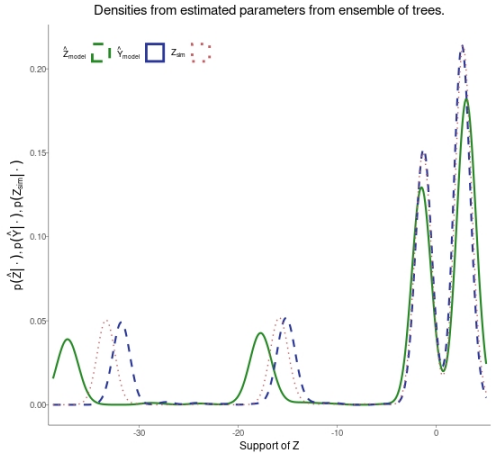
Figures 6.2.4a to 6.2.4d present the samples streams for the estimated densities in Experiments 1a, 2a, 3a and 4a. These graphs confirm the previous comments



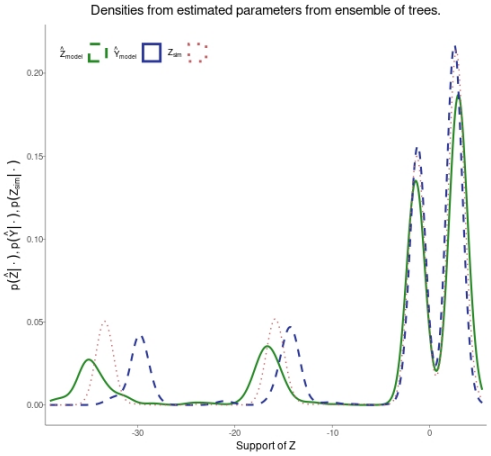
(a) Experiment 1a.



(b) Experiment 2a.

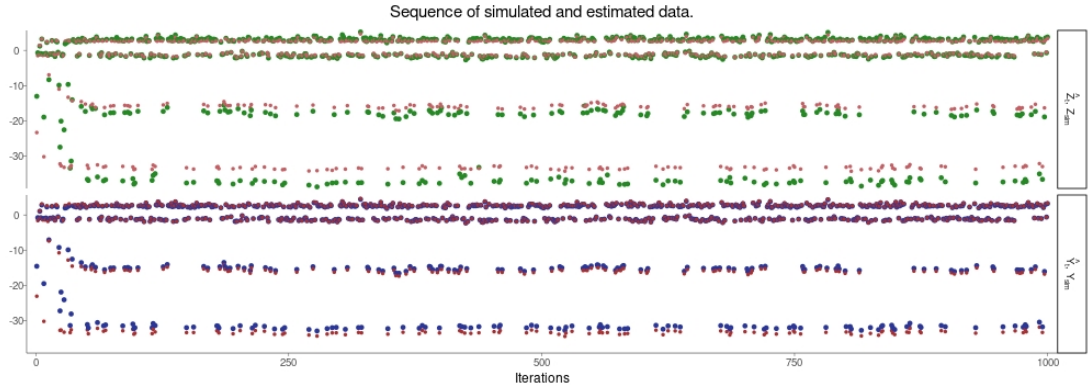


(c) Experiment 3a.

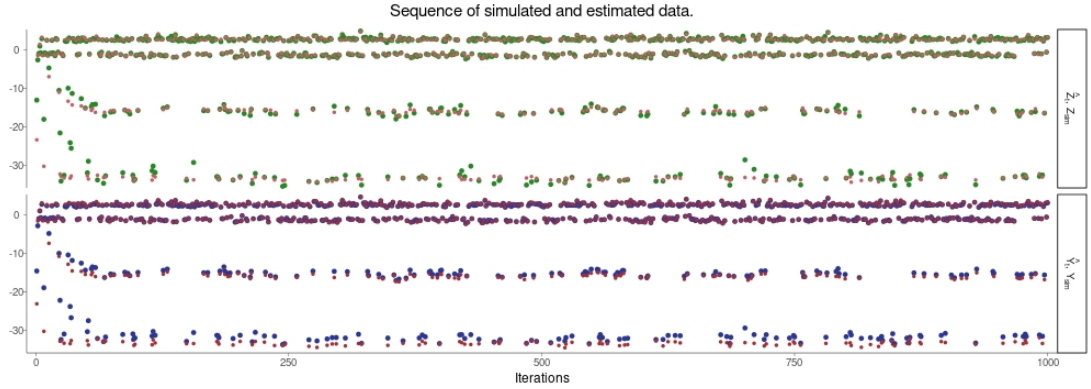


(d) Experiment 4a.

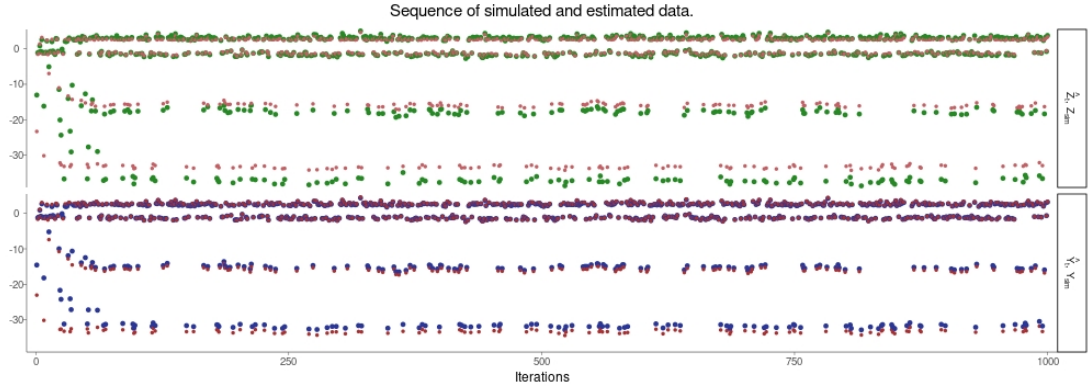
Figure 6.2.3: The estimated density when the only moves used are the change, grow and prune moves.



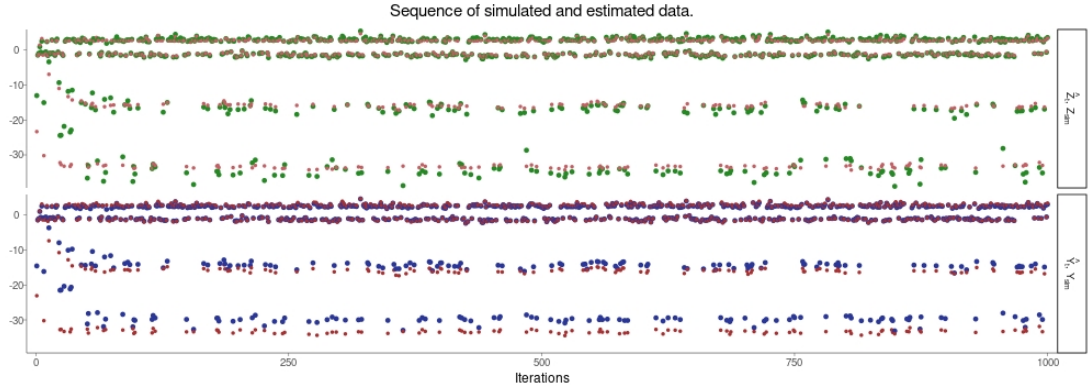
(a) Experiment 1a.



(b) Experiment 2a.



(c) Experiment 3a.



(d) Experiment 4a.

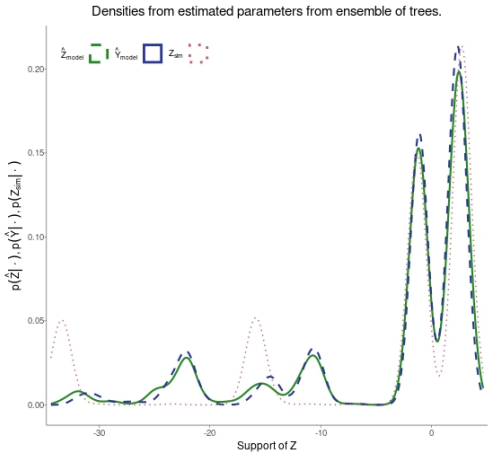
Figure 6.2.4: The sequence of estimates and predictions when only the change, grow and prune moves are used.

about the effects of increasing the number of evolution steps per iteration. However, notice that in the sequence for Experiment 1a there is a still a slight bias towards zero for the predictions and way from zero for the estimates. In Experiment 2a the bias seems to have been resolved for the estimates but exists to a lesser degree than 1a. for the predictions. Experiment 3a, which seemed to be the worst in terms of densities actually seems to have best predictions although the worst estimates. Experiment 4a goes some way to correcting the estimates but at the expense of a bias in the predictions.

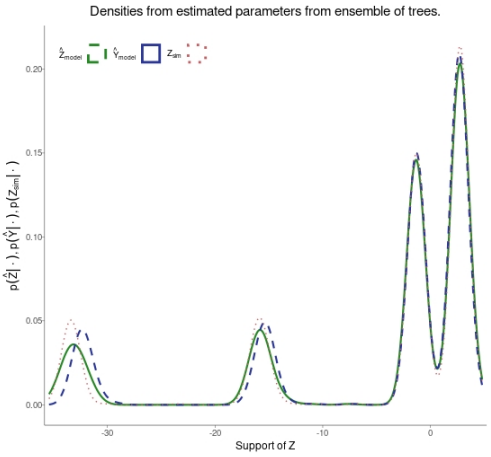
The densities in Figures 6.2.5a and 6.2.5b to 6.2.5d are estimated from sequences when all moves are used and there are $M = 5$ proposals per iteration (response and covariate). The parameters and modification functions are exactly those used for all experiments in this section.

It is immediately apparent that increasing the number of proposals detrimentally affects both the estimates and the predictions if the functions that modify F_b, W_b are retained. In contrast, using all the moves and doing multiple proposals per iteration when only the functions that modify V_b and H_b are used seems to produce the best output so far. Figures 6.2.6a to 6.2.6d show that by including multiple proposals and modifying the parameters F_b, W_b increase variability of the estimates and predictions and also cause the location of estimates and predictions to be offset. Figure 6.2.6b however shows that by removing the functions that modify the state parameters and increasing the number of proposals and using all the moves allows the ensemble model to find the target distribution and make accurate predictions.

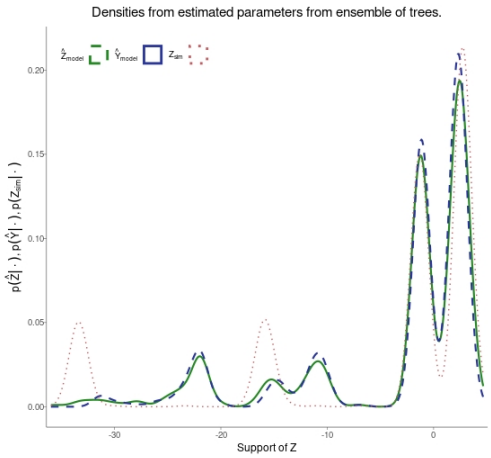
A tentative reason for the effects on the state shown here may be down to the fact that the trees are better at estimating the state with some general initial parameters and tree model learning rather than attempting to force parameter modification through deterministic functions. Further experimentation would be required to confirm this.



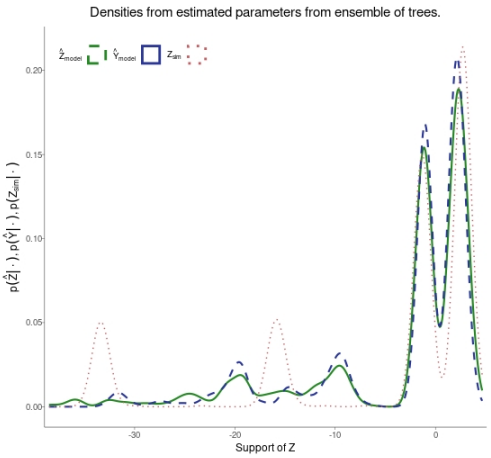
(a) Experiment 1b.



(b) Experiment 2b.

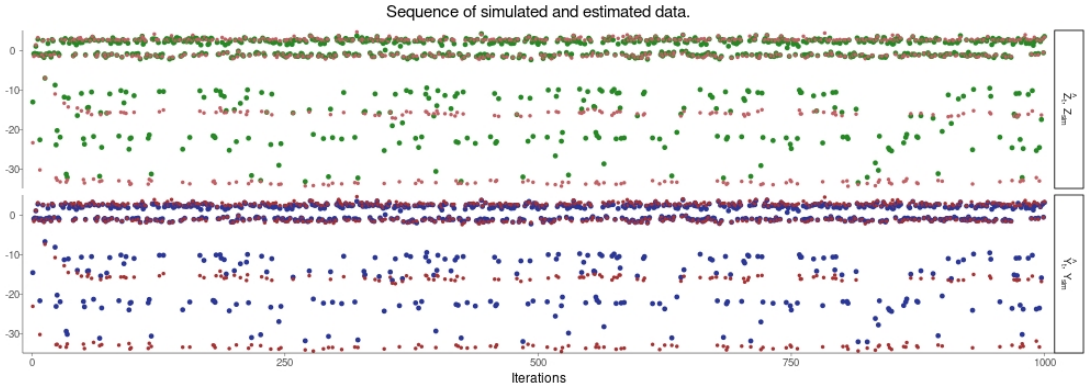


(c) Experiment 3b.

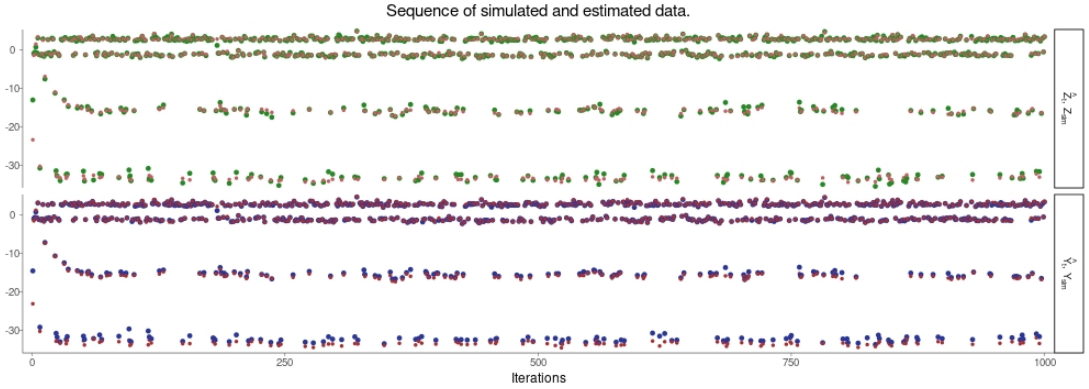


(d) Experiment 4b.

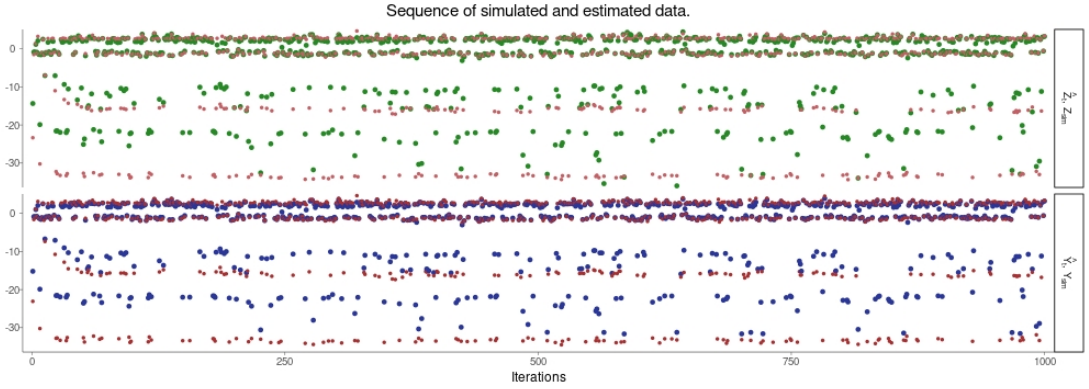
Figure 6.2.5: The estimated density when the only moves used are the change, grow and prune moves.



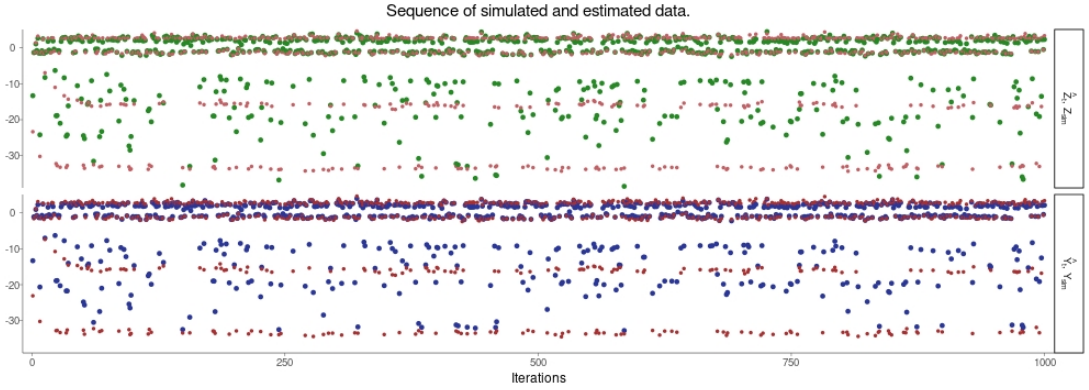
(a) Experiment 1b.



(b) Experiment 2b.



(c) Experiment 3b.



(d) Experiment 4b.

Figure 6.2.6: The sequence of estimates and predictions when only the change, grow and prune moves are used.

6.3 Mackey-Glass Experiment

This section briefly shows how the proposed model tracks a nonlinear and non-stationary stream of data. The iterations have been limited to 1000 but there is enough change in the stream to highlight some of the proposed algorithm's properties.

The Mackey-Glass function is a delay differential equation that is used in biomedical and physical sciences. The equations are defined as:

$$\frac{dG(t)}{dt} = \frac{\beta_0 \theta^n G(t-\tau)}{\theta^n + G(t-\tau)^n} - \gamma G(t) \tag{6.1}$$

This function was adapted for this experiment by converting it into a state space form and then by making the explanatory variables the previous points along the series.

In contrast to the experiments in Section 6.2, in this experiment the function modification parameters were kept at those that did not include modifying the state parameters. The initial ensemble parameters were modified.

Parameter values 1, SNR and H/V for 10 leaf tree.												
LN	PN	0	1	2	3	4	5	6	7	8	9	10
2	V	1.00	0.250	0.444	0.639	0.833	1.028	1.222	1.417	1.611	1.806	2.000
	W	20.00	10.000	11.111	12.222	13.333	14.444	15.556	16.667	17.778	18.889	20.000
	F	0.85	-0.400	0.450	-0.500	0.550	-0.600	0.650	-0.700	-0.750	0.800	-0.850
	H	1.00	0.900	0.900	0.917	0.950	0.983	1.017	1.050	1.083	1.117	1.150
	Z0	5.00	5.000	2.778	1.667	3.889	3.333	1.667	4.444	3.333	2.222	3.333
	W0	10.00	5.556	3.778	6.444	2.889	4.667	5.556	7.333	10.000	6.444	6.444
	R	20.00	40.000	25.000	19.130	16.000	14.054	12.727	11.765	11.034	10.462	10.000
	H/V	1.00	3.600	2.025	1.435	1.140	0.957	0.832	0.741	0.672	0.618	0.575
3	V	1.00	0.250	0.444	0.639	0.833	1.028	1.222	1.417	1.611	1.806	2.000
	W	20.00	10.000	11.111	12.222	13.333	14.444	15.556	16.667	17.778	18.889	20.000
	F	0.85	0.400	0.450	0.500	-0.550	-0.600	-0.650	0.700	0.750	0.800	0.850
	H	1.00	0.900	0.900	0.917	0.950	0.983	1.017	1.050	1.083	1.117	1.150
	Z0	5.00	3.889	0.556	1.667	0.000	4.444	3.333	3.889	1.667	3.333	3.333
	W0	10.00	3.778	10.000	3.778	2.000	4.667	3.778	4.667	5.556	5.556	8.222
	R	20.00	40.000	25.000	19.130	16.000	14.054	12.727	11.765	11.034	10.462	10.000
	H/V	1.00	3.600	2.025	1.435	1.140	0.957	0.832	0.741	0.672	0.618	0.575

Table 6.3.1: Parameters with high signal-to-noise.

A further aim in this experiments was to compare the case where there was one proposal or $M = 5$ proposals per iteration. Figures 6.3.1a to 6.3.1c show the densities for these 3 experiments:

1. Parameters 1 and only 1 move per iteration
2. Parameters 1 and 5 moves per iteration

Parameter values 2, SNR and H/V for 10 leaf tree.											
LN	PN	0	1	2	3	4	5	6	7	8	9	10
2	V	1.00	0.100	0.256	0.411	0.567	0.722	0.878	1.033	1.189	1.344	1.500
	W	20.00	1.000	2.000	3.000	4.000	5.000	6.000	7.000	8.000	9.000	10.000
	F	0.85	-0.400	-0.450	0.500	-0.550	0.600	0.650	-0.700	0.750	0.800	-0.850
	H	1.00	0.900	0.900	0.917	0.950	0.983	1.017	1.050	1.083	1.117	1.150
	Z0	5.00	5.000	2.222	3.333	1.667	0.556	2.778	3.889	0.556	3.889	2.778
	W0	10.00	2.000	7.333	9.111	3.778	9.111	3.778	10.000	10.000	6.444	2.889
	R	20.00	10.000	7.826	7.297	7.059	6.923	6.835	6.774	6.729	6.694	6.667
	H/V	1.00	9.000	3.522	2.230	1.676	1.362	1.158	1.016	0.911	0.831	0.767
3	V	1.00	0.100	0.256	0.411	0.567	0.722	0.878	1.033	1.189	1.344	1.500
	W	20.00	1.000	2.000	3.000	4.000	5.000	6.000	7.000	8.000	9.000	10.000
	F	0.85	0.400	-0.450	-0.500	-0.550	0.600	-0.650	0.700	0.750	0.800	-0.850
	H	1.00	0.900	0.900	0.917	0.950	0.983	1.017	1.050	1.083	1.117	1.150
	Z0	5.00	3.889	3.889	2.778	2.778	4.444	5.000	5.000	5.000	1.111	4.444
	W0	10.00	8.222	9.111	2.000	8.222	3.778	4.667	2.889	4.667	10.000	2.889
	R	20.00	10.000	7.826	7.297	7.059	6.923	6.835	6.774	6.729	6.694	6.667
	H/V	1.00	9.000	3.522	2.230	1.676	1.362	1.158	1.016	0.911	0.831	0.767

Table 6.3.2: Parameters with low signal-to-noise.

Functions and arguments for adapting parameters.										
move	fname	func	argn1	argv1	argn2	argv2	argn3	argv3	argn4	argv4
grow	Z0	constF								
	W0	constF								
	V	Vfunc	ln	2	scale	0.20	min	0.100		
	W	constF								
	F	constF								
	H	Vfunc	ln	2	scale	0.05	min	0.975		
prune	Z0	constF								
	W0	constF								
	V	IVfunc	ln	2	scale	0.20	max	2.000		
	W	constF								
	F	constF								
	H	IVfunc	ln	2	scale	0.05	max	1.025		

Table 6.3.3: Functions to modify parameters.

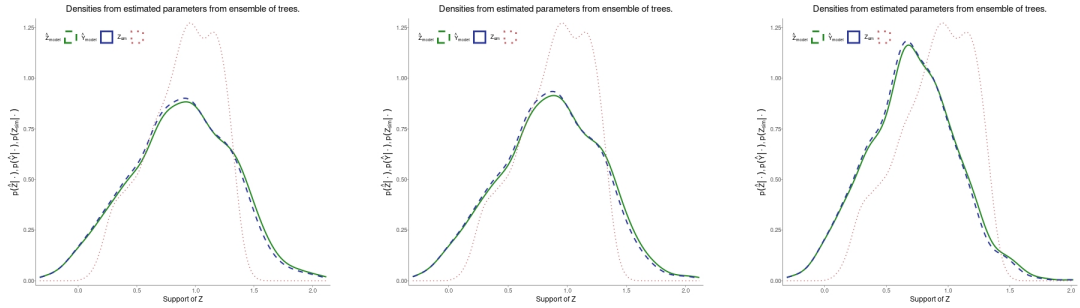
3. Parameters 2 and 5 moves per iteration

Note that for all these experiments only 3 move types, change, grow and prune were used.

Figures 6.3.2a to 6.3.2c show a sample stream from the Mackey-Glass generator. Clearly the Experiment 3, where the signal-to-noise ratio is smallest has the largest effect on the predictions.

Figures 6.3.3a to 6.3.3c show the cumulative square error of the experiments. The best performing is Experiment 2 where the state estimate is stable, unlike in Experiment 1 where it is is slightly increasing.

Figures 6.3.4 to 6.3.6 each show the MCMC component output for each experiment. Of particular notice is Experiment 3 where the signal-to-noise ratio is

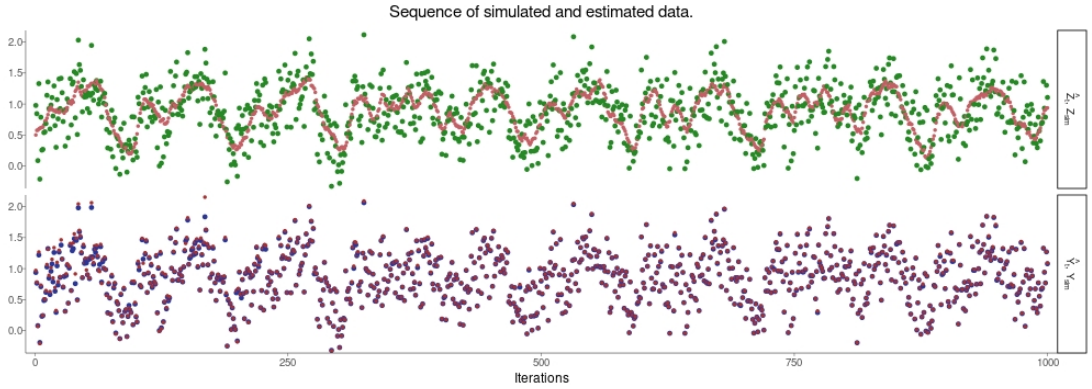


(a) Experiment 1. (b) Experiment 2. (c) Experiment 3.

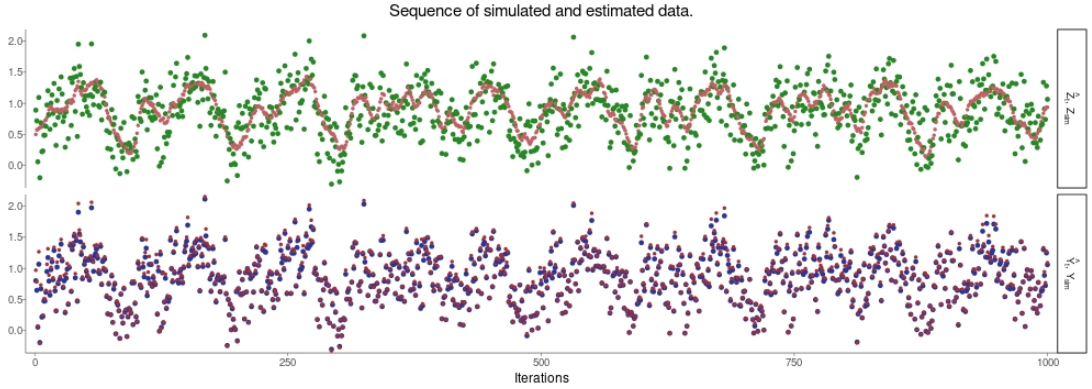
Figure 6.3.1: Estimated densities for three Mackey-Glass experiments

lowest. Here, tree 1 "escapes" and becomes very large. This can cause the algorithm to fail in extreme cases because the complexity of the algorithm increases exponentially with the number of leaves.

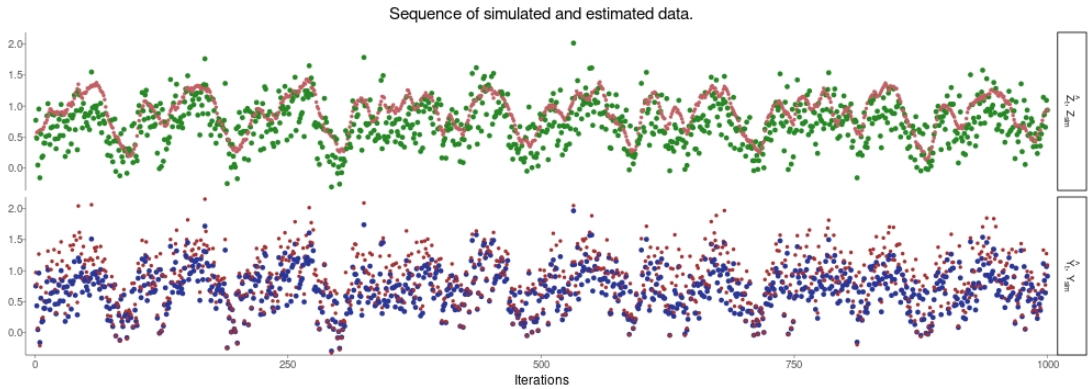
Figures 6.3.7a to 6.3.7c show the number of leaves and some additional information about the MCMC process. Note especially the number of leaves in Experiment 3.



(a) Experiment 1.

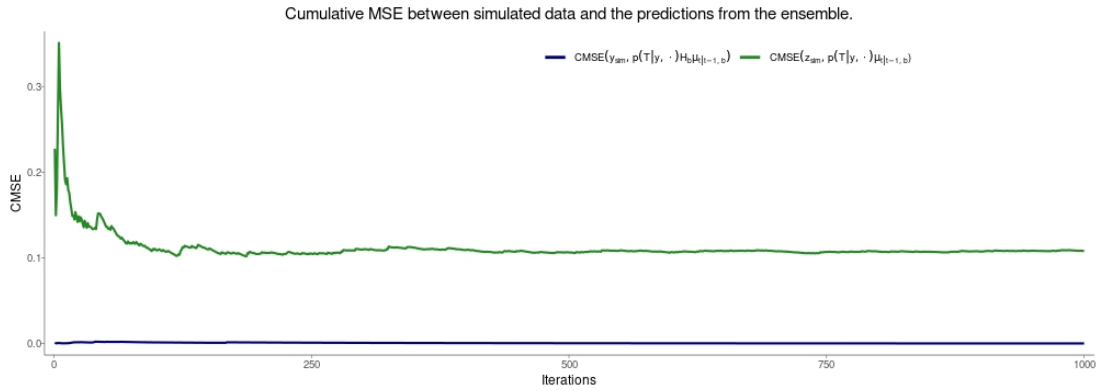


(b) Experiment 2.

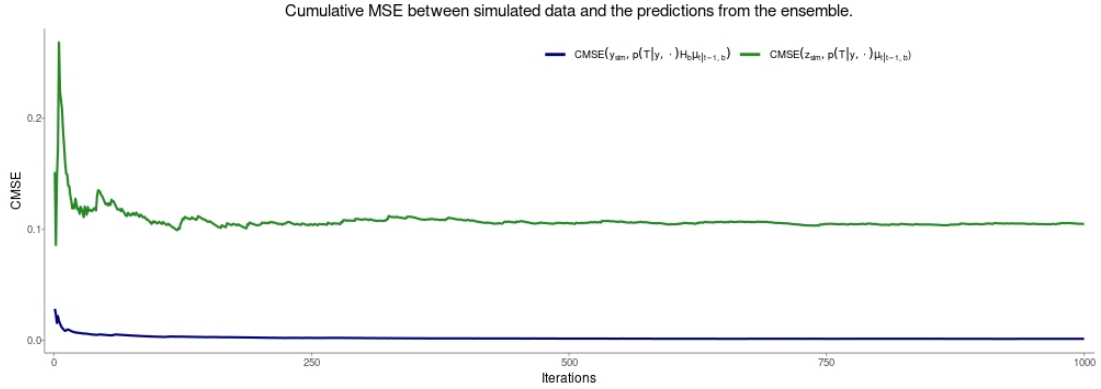


(c) Experiment 3.

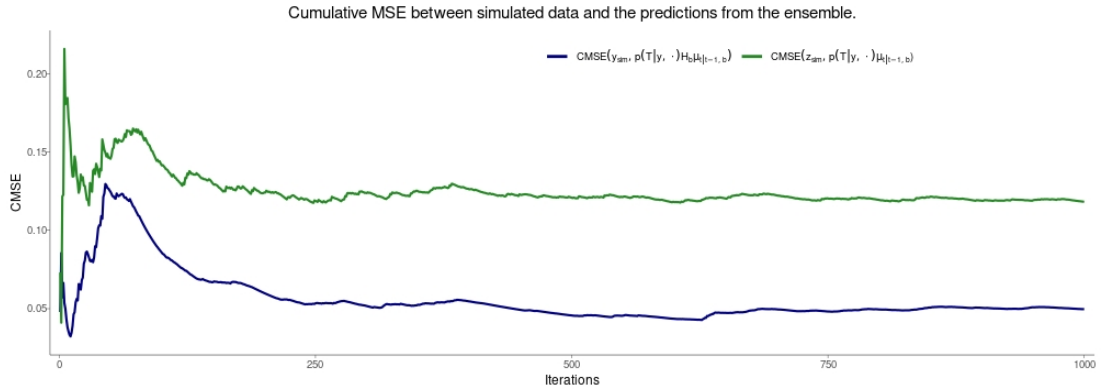
Figure 6.3.2: A sample stream from the Mackey-Glass data generator



(a) Experiment 1.



(b) Experiment 2.



(c) Experiment 3.

Figure 6.3.3: The cumulative means square error for three experiments with the Mackey-Glass sample stream.

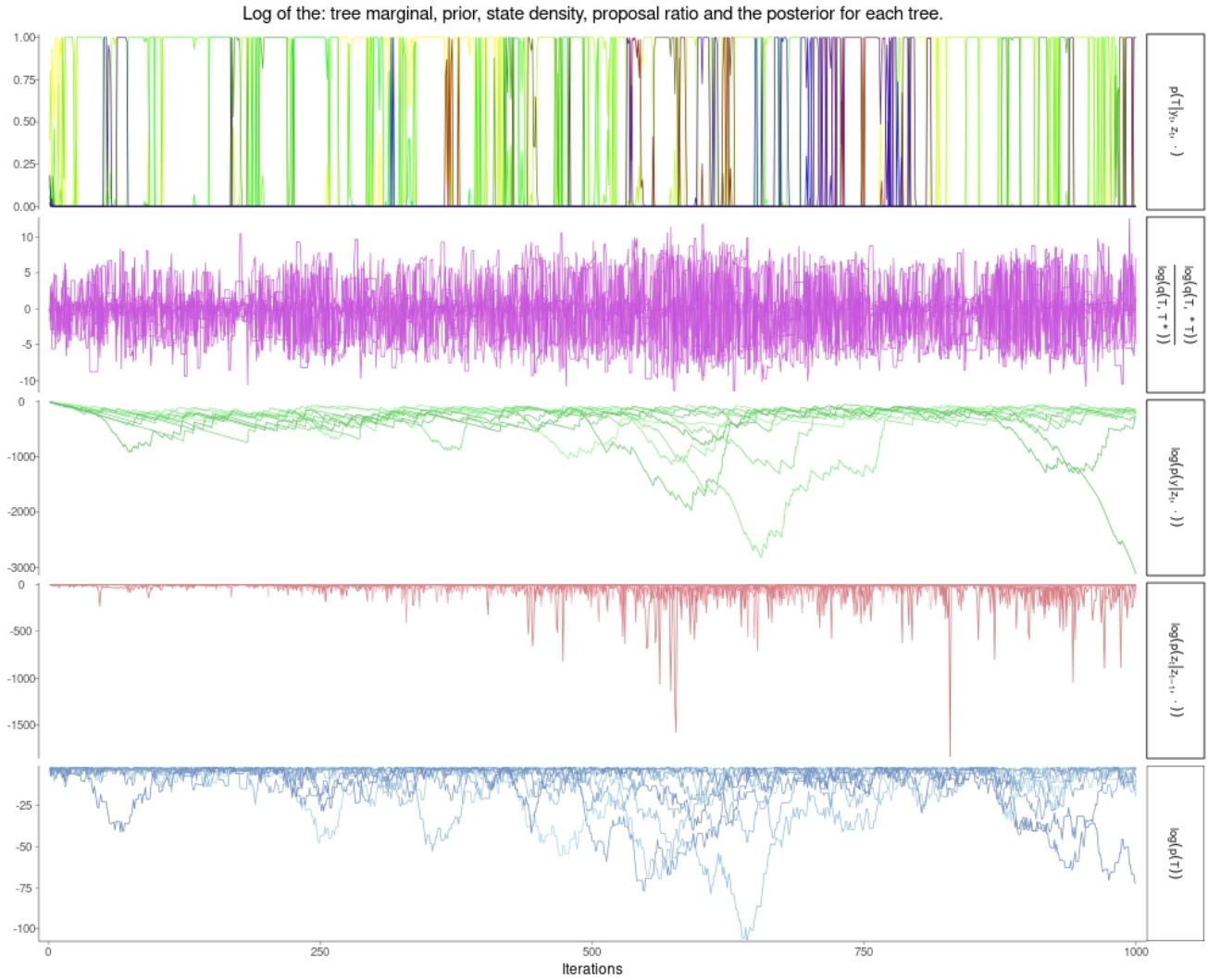


Figure 6.3.4: MCMC component output for Experiment 1.

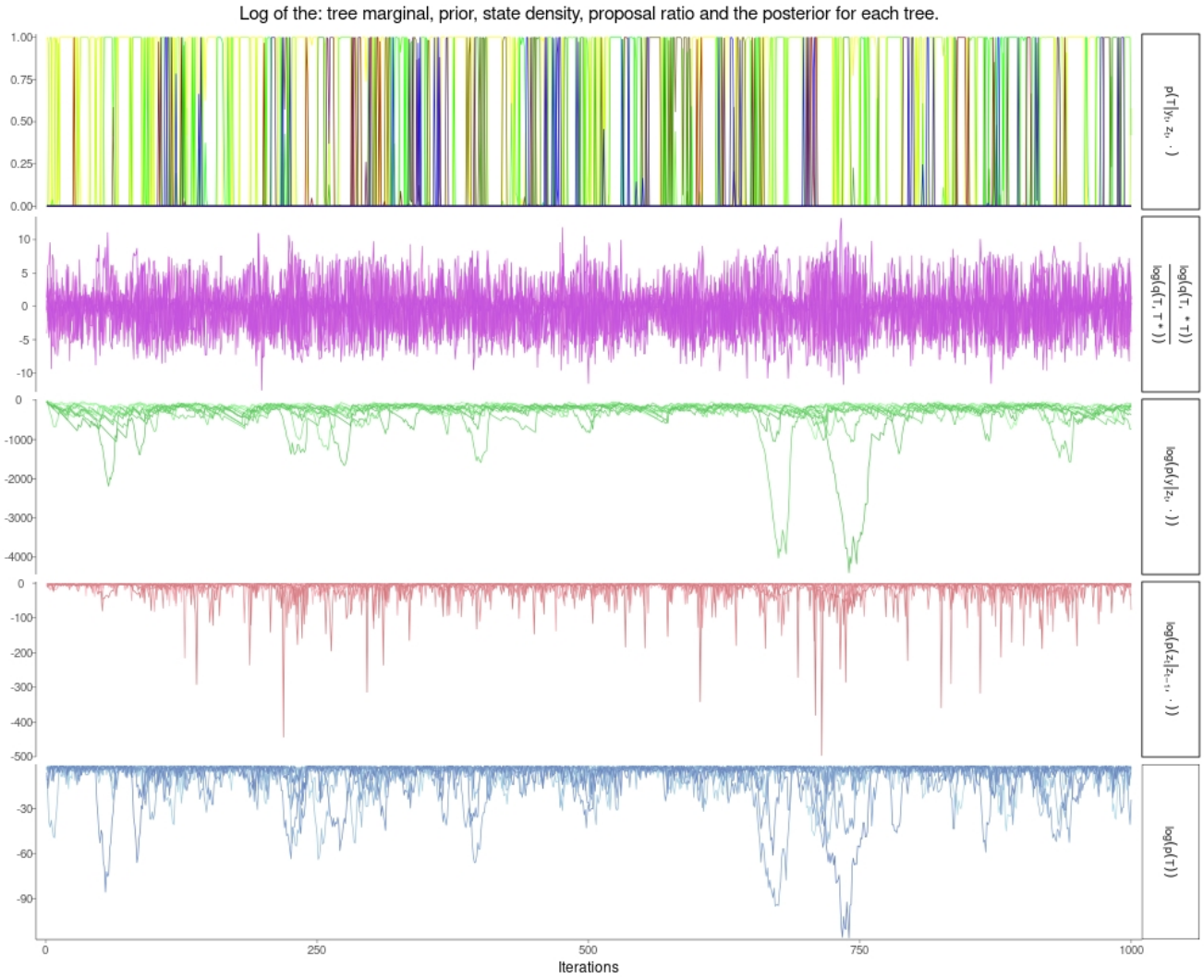


Figure 6.3.5: MCMC component output for Experiment 2.

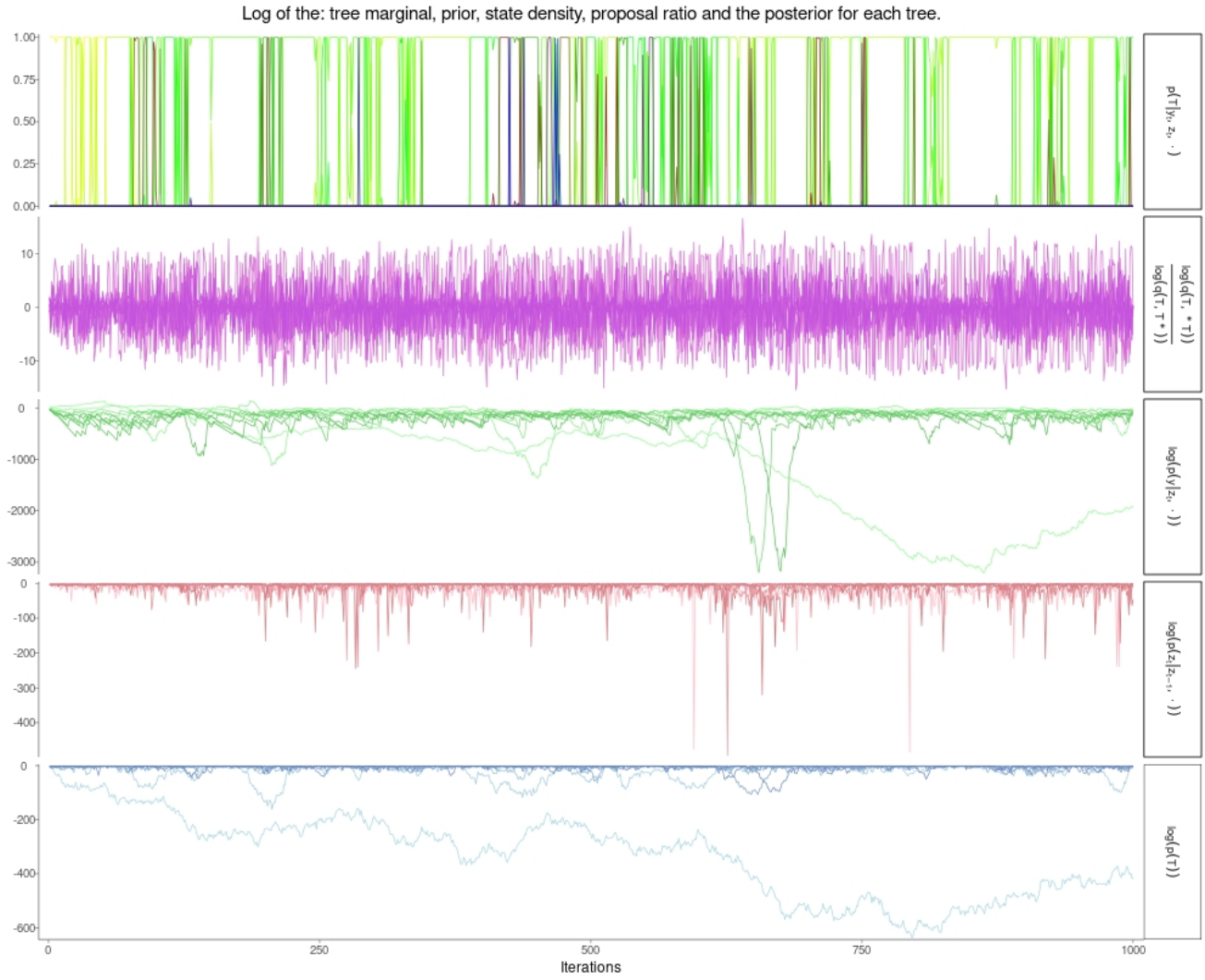
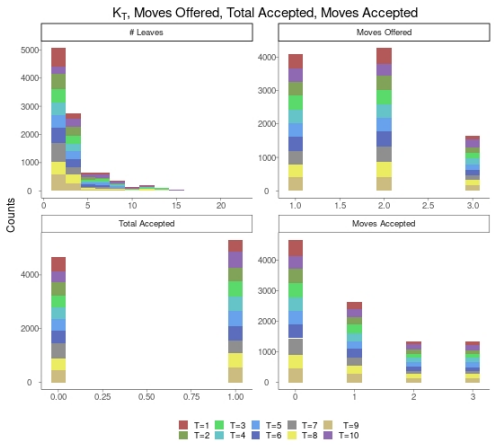
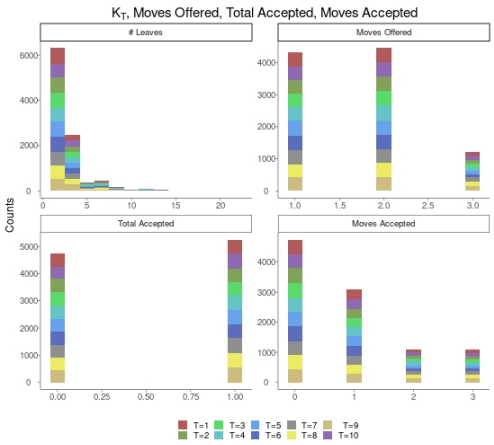


Figure 6.3.6: MCMC component output for Experiment 3.



(a) Experiment 1.



(b) Experiment 2.

(c) Experiment 3.

Figure 6.3.7: MCMC analysis for three experiments of Mackey-Glass stream

7 Review and Conclusion

7.1 Critical Review

The aim of this research was to propose a method for on-the-fly regression using tree models in a streaming data setting. The Bayesian statistical framework was chosen to provide estimates and predictions with uncertainty bounds. Unlike existing approaches in the Bayesian paradigm which use Sequential Monte Carlo (SMC) (Taddy et al. 2011; Anagnostopoulos and Gramacy 2013) or Variational Bayes (Broderick et al. 2013; Campbell et al. 2015) the proposed approach used Markov chain Monte Carlo (MCMC) as an iterative sampler for the space of tree models. This is not without controversy because a fundamental assumption behind MCMC is a stationary target distribution, an assumption invalidated by definition in the streaming data setting. A key argument in this paper was the approximation of the target posterior by the ensemble of tree models that was "close enough" to the true target posterior distribution. The loss in accuracy could be balanced by a gain in speed of inference if that loss was small enough and the gain was necessary and large enough in the current temporal context. This document showed the complexities and difficulties of designing a regression modelling system that is adaptive in a nonstationary streaming data (on-the-fly) setting. Using some simulated data, the proposed approach demonstrated that it can outperform some of the state-of-the-art methods. However the lack of: complete simulation studies; a real-world data comparison; comparisons outside of the Bayesian setting and an exploration of automated parameter sampling creates some doubt regarding the applicability of the modelling approach in its current form. A further lack of quantified measures and measurements between the benefits of speed and the loss of accuracy makes supporting the conclusions drawn from the model difficult.

A fixed sized ensemble of tree models with intermittent Kalman filters (Sinopoli et al. 2004) at all the leaves of each of the trees was used to provide estimates, predictions and their associated uncertainty bounds at each discrete index t. Every

tree in the mixture of tree models were weighted with their posterior probabilities. The prior for a tree was a sample from the space of possible trees and the generator for these samples was a stochastic tree sampling process (Chipman et al. 1998). The likelihood of a tree model was formed by marginalising over the leaf models that existed at index t with respect to the state space of the filter at those leaves. At each t a new observation was assigned, by the rules at the internal nodes of the tree, to update the state estimate of the filter level. Thus the likelihood of the tree increments by a single observation at each t and was relative to the state space of all the leaf models at that t. Sampling tree models from the prior involves proposing alternate tree model shapes and rules and these alternate models are compared to the existing models using the MCMC acceptance ratio. Every tree, either the accepted proposal or the current model, is passed to the next index, $t + 1$, and the above procedure for choosing leaf models to update, and propose new trees, is repeated. No covariate data is stored in the tree thus removing the need for data reservoirs and the associated memory overheads. The response values are used only to update the state level and, by the assumption of conditional exchangeability given the covariates, are not necessary for subsequent iterations. The recursive intermittent Kalman filter and the forward-only passing of the tree models means that this proposed modelling approach has constant complexity in: the order of the size of the ensemble; the upper bound in tree size (number of leaves hence filters) and the order of the minimum number of matrix inversions necessary in the chosen intermittent Kalman filter. At every t, the evidence for the tree models is approximated by summing over the tree ensemble. The posterior density for each tree is calculated using Baye's rule and these weights, once normalised, are applied to their respective estimates, predictions and bound values so that a single set of values is produced by the ensemble at each t.

The first contribution of this thesis is the fixed tree filter. A tree model (Chipman et al. 1998) is adapted to provide an intermittent Kalman filter at each leaf with the intention of coping with local nonstationarity around the main effect at the root node. This model shows how conditioning a filter can improve uncertainty quantification. A significant aspect of this contribution is the combined hidden Markov structure of the tree model and the intermittent filters at the leaves so that storing new covariate and response data in memory is not required. However, this complex and over-parametrised model requires significant effort in specification. There is a lack of asymptotic bounds as shown in related models (Kalman 1960; Harrison and West 1999; Sinopoli et al. 2004). The calibration study (Section 4.4) was an attempt to provide some guidance for choosing a set

of parameters. This attempt was partially successful in finding that the tree marginal could be bounded even if updates were not as frequent as that required by the critical probability of Sinopoli et al. (2004). Some effort was made to examine the tree filter residual and its variance but no firm relationships were provided.

The second contribution of this thesis was to extend the tree filter by incorporating dynamic model search into the on-the-fly estimation process. Unlike the usual MCMC process, inference on the state of the observed process was performed at every iteration of the model search. Including the bounds of each chosen covariate into the tree node rules was used to ensure that all leaf measures were coherent. Additional MCMC moves were proposed to improve chain mixing. An additional aspect to resolve parameter selection for randomly chosen leaf models was to assign a deterministic function to new parameters based on the parameters of ancestral leaf models, or where appropriate, to the initial leaf parameters. Partially complete simulation studies of prior and tree model behaviour were provided but the results of these did not provide enough strength to support the arguments therein.

The third contribution of this thesis was to extend the random tree model search to an ensemble of random tree models. This extension provides an approximate cover for the tree model space so that at each t the tree model evidence can be calculated and used to normalise the density of each tree model. These posterior weights are combined with model outputs (estimates, predictions and bounds) to form linear combinations that are the ensemble outputs. A considerable difficulty in designing experiments with this model is setting initial values of parameters for the filters of the ensemble. Nonstationarity renders prior values of limited use. While a deterministic method for parameter adaptation was presented, sampling of parameters was not explored. A reason for using an ensemble of Markov chains, much like using particles (Carvalho et al. 2010), is to explore the model space. Increasing ensemble size and setting a range of hyperparameters for sampling distributions would be more in keeping within the aims of the MCMC approach. The weakness of results and quantified boundaries from the preceding experiments did not help to limit the scope of sampling to a subset of parameters. These weaknesses further limited the design of subsequent experiments that would have provided more concrete results for the assessment of the proposed method.

In the final contribution two adaptions of the initial approach were proposed, the first adaption was to use only three simpler moves and the second was to allow for multiple model proposals per observation. These modifications showed that performance and accuracy could be improved but a lack of firm benchmarks in

earlier chapters made quantification of the improvements from these adaptations difficult.

One aim of this thesis was to show that Markov chain Monte Carlo could be used to sample tree models for on-the-fly regression. This was achieved to some extent in showing that it was possible and, at least with limited simulations and comparisons, accurate "enough". However, a lack of quantification and measurement of results makes it difficult to assess the extent to which this thesis contributes to the large body of knowledge that surrounds tree methods. At the end of Section 5.4.2.3 some attempt was made towards measuring the performance of the three simulations of that section but this method was not used for comparing against the state-of-the-art Bayesian models, Dynamic Trees (Taddy et al. 2011) and Bayesian additive regression trees (Chipman et al. 2010) let alone other tree-based models in the wider field (Hulten and Domingos 2003; Freund and Robert E Schapire 1997; Bifet and Gavaldà 2009). The lack of real-world examples prevented empirical comparison.

The chosen Bayesian methodology with its recursive structure is intuitively appropriate to this thesis. While there was acknowledgement of the frequentist setting there were no direct comparisons to this methodology. Direct comparison via a real-world example would have been best but even a comparison between speeds/complexity on simulated data would have helped to motivate the benefits of this approach. Markov chain Monte Carlo (MCMC) sampling was the method chosen to gather evidence to support the data models. However the MCMC method is fundamentally supported by the assumption of a stationary target distribution. This thesis does not challenge the assumption of stationarity but seeks to explore how this assumption can be stretched in a sampling approach with the aim of enabling wider use of MCMC in the streaming setting. A direct comparison with statistical mechanics and a canonical ensemble, which does allow for gradual changes the stationary distribution, would have provided greater structure to the approach. Further, including simulated annealing (Geyer and Thompson 1995) would have provided a direct link to the mechanical basis behind MCMC and strengthened this argument. The chosen MCMC approach was compared to sequential Monte Carlo (SMC) (Gordon et al. 1993; Doucet et al. 2001; Carvalho et al. 2010) but the variational approach of Broderick et al. (2013) and Campbell et al. (2015) should have been further explored as this too has a basis in statistical mechanics.

There is a clear problem statement and hypothesis. However, the development of this hypothesis would have been better if the background section had been more targeted towards supporting the direct interventions proposed rather than linking

all of regression to dynamic modelling. Separating the development of the model helped to introduce ideas in a logical way but these chapters lacked a firm goal and outcomes that prevented the accumulation of results for a final judgement on the model. The graphs and results that were presented were informative and demonstrated the performance of the model. It is acknowledged that the timing of results is pertinent to streaming analysis and the comparison of results in this setting is difficult. However, as already mentioned, there exist methods for quantifying the differences between distributions and these were not fully utilised to support claims in the thesis. While the simulations showed that the proposed model could get "very close" to a fixed target distribution, the simulations did not show how the proposed (or any other method) would adapt as the target distribution itself changed. Finally, despite the late addition of the Mackey-Glass simulation, no other real-world examples were provided. Empirical evidence and comparison would have supported the claims, that could have been stated rather than shown, better than further simulations.

In summary, it is clear that this was a difficult problem to be solved. There were many interactive and changing parts which required complicated coding and preliminary analysis before experiments could be run let alone real-world examples developed for comparison. None-the-less, as evidenced by the document and the argument, a more considered, reductionist approach to the thesis would have provided more way-points for benchmarks. These benchmarks, and their quantification, would have helped to shape the development of the model which, in turn, would have helped limit the scope and argument of the thesis. Limiting the scope of the thesis would have pinpointed simulations and examples to be pursued for comparison. The results of these experiments could then have been used to simplify the specification of the model so that sampling of fewer parameters and greater automation of the approach could have been demonstrated and tested. The lack of experiments on at least one real-world example prevents one from assessing the applicability of the proposed model. However, with complicated problems some clearing of the "under brush" is necessary before the wood can be seen so, to this extent, the exploratory analysis and testing that went into this thesis has provided a meaningful contribution and some guidance for future work on this interesting subject.

7.2 Future Work

The proposed model has only been initially explored. Showing that it is possible to use MCMC in the streaming and *Big Data* settings for on-the-fly inference

could open several avenues for further exploration in this field. Future work should focus on simplifying the above model so that benchmarks and comparisons can be made. The first step in this process would be to explore the use of sampling for controlling the signal-to-noise ratio. Automating the modelling process and reducing the number of input filter parameters would likely lead to better comparisons with existing approaches. The next step would be to assess the mixing of the ensemble. Exploring the modifications of Chapter 6 by considering different numbers of proposal moves per data point for each tree one might be able to have "slow" and "fast" trees that adapt to changes in the data stream. The third initial step, which would part of exploring mixing and accuracy, would be to use simulated annealing (Geyer and Thompson 1995) to extend the range of "temperatures" at which the ensemble could adapt to changes in the streaming data environment. The final initial step would be to assess the performance of the model with respect to speed of inference and complexity. This could lead to quantitative assessments of the trade-off between accuracy and performance.

Once these initial steps have been completed and quantified the task would be to run the algorithm in some real-world data settings. The first of these tasks would be a bake-off between this model, other Bayesian models and some approaches in the frequentist setting using benchmark data with accepted standard results. The second task would be use the proposed model with some novel data and compare this against some existing methods. These tasks could show whether it would be worth developing a stand-alone package or programme (in the manner of Bifet, G. Holmes, et al. (2010) and *streamDM: Data Mining for Spark Streaming* (2021)) that connects to existing streaming architecture such as Kafka (*Apache Kafka - Streaming* 2021). Experimentation on real-world data would also indicate whether this proposed model filled a particular niche or was a better alternative to existing methods.

Once the validity and applicability of the model using base settings was established, additional extensions of the model could be considered. These might include nonlinear filters at the leaves, dependent leaves, duration based parameter estimation and variable selection or model interpretation.

A1 Appendices

A1.1 Derivation of the Multivariate Marginal Posterior

The posterior probability in Equation (4.14) is proportional to two terms, the likelihood of the data from $t = 1, \ldots, t$ conditional on the latent process Z^t from $t = 0, \ldots, t$ and both conditional on the prior probability for the tree. The posterior probability can be factored as:

$$\begin{aligned}\mathsf{p}\left(T, Z^t \mid Y^t, \theta_T, x^t\right) &= \mathsf{p}\left(T \mid Y^t, \theta_T, x^t\right) \mathsf{p}\left(Z^t \mid T, Y^t, \theta_T, x^t\right) \\ &= \underbrace{\mathsf{p}\left(T \mid Y^t, \theta_T, x^t\right)}_{\text{Ⓐ Posterior for tree model}} \times \underbrace{\prod_{b=1}^{K_T} \mathsf{p}\left(Z_b^t \mid T, Y^t, \theta_T, x^t\right)}_{\text{Ⓑ Product of Gaussians from the IKF}}\end{aligned} \tag{A1.1}$$

where $\theta_T = \{\xi_T, \psi_T\}$. The Gaussian posterior distributions in Ⓑ are the result of either an updated or not-updated density as described in Section 4.3 and will not be further described here. The focus of this derivation is on Ⓐ, the posterior for the tree model conditional on the latent processes at each of the leaves.

By Bayes' rule:

$$\begin{aligned}\mathsf{p}\left(T \mid Y^t, \theta_T, x^t\right) &\propto \mathsf{p}\left(T \mid \xi_T\right) \mathsf{p}\left(Y^t \mid T, \psi_T, x^t\right) \\ &= \mathsf{p}\left(T \mid \xi_T\right) \underbrace{\int \mathsf{p}\left(Y^t \mid T, Z^t, \psi_T, x^t\right) \mathsf{p}\left(Z^t \mid T, \psi_T, x^t\right) \, dZ^t}_{\text{Ⓒ Marginal for } T \text{ over latent process } Z^t}\end{aligned} \tag{A1.2}$$

so that Ⓒ, is the marginal likelihood of the tree over all latent processes Z_i, from $i = 0, \ldots, t$. At each i, the tree prior, $\mathsf{p}\left(T \mid \xi_T\right)$ is sampled according the method specified in Chapter 4 so while Equation (A1.2) incorporates the joint densities of all the observed and latent data from $i = 0, \ldots, t$, the posterior for the tree,

Ⓐ, only reflects the density conditional on each tree sample at each t, hence the lack of a i or t sub-or-superscript on T.

By using the Markov assumption for the Z^t processes; the independence of each of the $Z_{t,b} \mid T$ and also by the filter model assumption $Y_s \perp\!\!\!\perp Y_t \mid Z_t, x_t$ Equation (A1.2) can be written as:

$$
\begin{aligned}
&= \mathsf{p}\left(T \mid \xi_T\right) \int \prod_{b=1}^{K_T} \mathsf{p}\left(Y_b^t \mid Z^t, T, \psi_b, x^t\right) \mathsf{p}\left(Z_b^t \mid T, \psi_b, x^t\right) \, dZ^t \\
&= \mathsf{p}\left(T \mid \xi_T\right) \int \prod_{i=0}^{t} \prod_{b=1}^{K_T} \mathsf{p}\left(y_{i,b} | z_{i,b}, T, \psi_b, x_i\right) \mathsf{p}\left(z_{i,b} | T, \psi_T, x_i\right) \mathsf{p}\left(z_{i-1,b} | T, \psi_b, x_i\right) dz^t
\end{aligned}
\tag{A1.3}
$$

where the change to lower case for the integrands is for the sake of visual clarity. Note that the densities under the integral are random functions of the parameter Z_t at each t and the differential $dZ^t = dz^t = dz_0 dz_1 \ldots dz_t$ is with respect to the random measure represented by the density $\mathsf{p}\left(z_{i-1,b} | T, \psi_T, x_i\right)$.

At each i from $i = 1, \ldots, t$ only one leaf is selected from the tree by x_i and to indicate this define:

$$
\mathrm{I_{i,b}} = \begin{cases} 1, & \text{if } \eta(x_i, T) = b \\ 0, & \text{otherwise} \end{cases}
$$

where $\eta(x_i, T) = b$ is a function $\eta(x_i, T) : T \to b$, $b \in \{1, \ldots, K_T\}$ that chooses a leaf from a tree based on covariate vector x_i. This simplifies the conditional likelihood of the data over all $1, \ldots, t$ to:

$$
\begin{aligned}
\mathsf{p}\left(y_{i,b} \mid z_{i.b}, T, x_i, \psi_b\right) = \mathsf{p}\left(y_{i,b} \mid T, z_{i,\eta(x_i,T)}, \psi_b\right) &= \prod_{i=1}^{t} \prod_{b=1}^{K_T} \mathsf{p}\left(y_{i,b} \mid T, z_{i,b}, \psi_b\right) \\
&= \prod_{i=1}^{t} \mathsf{p}\left(y_{i,b} \mid T, z_{i,b}, \psi_b\right)^{\mathrm{I_{i,b}}}
\end{aligned}
\tag{A1.4}
$$

and, as there is no observation at $t = 0$,

$$
\begin{aligned}
&\mathsf{p}\left(T \mid y^t, x^t, \theta_T\right) \propto \\
&\qquad \mathsf{p}\left(T \mid \xi_T\right) \int \prod_{b=1}^{K_T} \mathsf{p}\left(z_{0,b} \mid T, \psi_b\right) \prod_{i=1}^{t} \mathsf{p}\left(y_{i,b} \mid T, z_{i,b}, \psi_b\right)^{\mathrm{I_{i,b}}} \mathsf{p}\left(z_{i,b} \mid z_{i-1,b}, T, \psi_b\right) dz^t.
\end{aligned}
\tag{A1.5}
$$

The covariate at each $i \in 1, \ldots, t$ allocates the data $y_{i,b}$ to one of the $b \in$

$1, \ldots, K_T$ leaves. But at every i the latent state z_i is predicted at every b so the densities, $\mathsf{p}\,(\,z_{i,b} \mid T, \psi_b\,)$ (where ψ_b are the leaf model parameters), are independent of the x_i for all t. Further, in this derivation the leaf model parameters, $\psi_b = (H_b, F_b, V_b, W_b, \mu_{0,b}, W_{0,b})$, may differ at each leaf but are known up to a deterministic functional form for the duration of the streaming analysis. The tree model parameters, $\xi_T = (\alpha, \beta, B, p)$, are also assumed fixed and known. Thus θ_T and x_i are dropped from the equations that follow, and where necessary, the ψ_b are made explicit.

K_T, the number of leaves of T, is a function of the tree. Thus at each t the number of leaves to be marginalised over may be different because the structure of the tree model may change. However, by marginalising over all the leaves, as is it done in Chipman et al. 1998 and others, this change in the dimension of the tree can be ignored. Further, this marginalization means that, conditional on the tree model and covariate vector x_t, all possible latent states that complete Y_t have been accounted for at each t. Thus irrespective of the number of possible models (leaves), the tree model (dubbed tree filter) provides a complete representation of the support for Y^t.

Combining the above marginalization with the aforementioned sample nature of the tree prior means that $\mathsf{p}\,(\,T\,)$ is not a sample of a sequence of trees from $\mathsf{p}\,(\,T^t\,)$, a tree evolution process, hence there is no need to consider $K_{T_t} \mid K_{T_{t-1}}, \ldots, K_{T_0}$, the sequence of tree dimensions nor their joint probability. Thus $K_{T_i} \perp\!\!\!\perp K_{T_j}$ for all $i, j \in 1, \ldots, t$.

Bearing the above in mind Equation (A1.5) can be written as:

$$\mathsf{p}\left(\,T \mid y^t\,\right) \propto \mathsf{p}\,(\,T\,) \prod_{b=1}^{K_T} \underbrace{\int \mathsf{p}\,(\,z_{0,b} \mid T\,) \prod_{i=1}^{t} \mathsf{p}\,(\,y_{i,b} \mid T, z_{i,b}, \psi_b\,)^{\mathrm{I_{i,b}}}\, \mathsf{p}\,(\,z_{i,b} \mid T, z_{i-1,b}, \psi_b\,)\, dz^t}_{\text{(D).The leaf likelihood}} \tag{A1.6}$$

Equation (A1.6) shows that the likelihood of the tree is the product of a discrete number of leaf likelihoods. Each leaf likelihood is a stochastic process where the joint distribution of the latent variables are modelled by the intermittent Kalman filter.

It is clear from Equation (A1.6) that when new leaves are generated the initial values for the latent processes at these leaves must be sampled from the latent process of the parent leaf else these values will not be incorporated into leaf marginal which, for each leaf runs from the initialization of the leaf up to index t. Thus in Equation (A1.6), $\mathsf{p}\,(\,z_{0,b}\,)$ represents the initial distribution of the state

and the parameters, $\mu_{0,b}$ and $W_{0,b}$ for $\mathsf{p}\,(\,z_{0,b}\,) \sim N(\mu_{0,b}, W_{0,b})$ for each new filter, are typically only specified by the analyst for the initial two leaf models $b = 2, 3$ for each T.

Further, although the index t in Equation (A1.6) starts at 1 this is clearly not true for every b. Each refinement of b into two children generates two new leaf likelihoods but removes the parent leaf likelihood and similarly removing child leaves removes two trajectories and replaces them with one trajectory. Thus the leaf likelihood in Ⓓ represents the joint marginal of all the sequences of latent processes Z^t that have occurred from $0, \ldots, t$. Bearing in mind that every tree is initialized with a minimum of two leaves this means that Equation (A1.6) represents, at minimum, the joint likelihood of the two alternative models each conditional on the main effect at the root.

Equation (A1.6) and the above description means that up to b every integral is the same so that for most of the rest of the derivation Ⓓ, will be considered an arbitrary leaf likelihood, will be the calculation of interest.

Using the assumptions of Gaussian distributions as mentioned in Section 2.2.2 and in Chapter 4, the densities for the initial states, $z_{0,2}$ and $z_{0,3}$, are:

$$\mathsf{p}\,(\,z_{0,b}\,) = ((2\pi)^m |W_{0,b}|)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(z_{0,b} - \mu_{0,b})^T W_{0,b}^{-1}(z_{0,b} - \mu_{0,b})\right] \tag{A1.7}$$

and for each subsequent index $i = 1, \ldots, t$ the conditional densities of the state are

$$\mathsf{p}\,(\,z_{i,b} \mid z_{i-1,b}\,) = ((2\pi)^m |W_{t,b}|)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(z_{i,b} - F_{t,b} z_{i-1,b})^T W_{t,b}^{-1}(z_{i,b} - F_{t,b} z_{i-1,b})\right]. \tag{A1.8}$$

The joint density for the conditionally exchangeable observations, following Equation (A1.4), is:

$$\prod_{i=1}^{t} \mathsf{p}\,(\,y_i \mid z_{0,b}\,)^{\mathrm{I_{i,b}}} = ((2\pi)^n |V_{t,b}|)^{\frac{t \cdot \mathrm{I_{i,b}}}{2}} \exp\left[-\frac{1}{2}\sum_{i=1}^{t} \mathrm{I_{i,b}}(y_{i,b} - H_{t,b} z_{i,b})^T V_{t,b}^{-1}(y_{i,b} - H_{t,b} z_{i,b})\right]. \tag{A1.9}$$

Thus:

$$\begin{aligned}
\mathsf{p}\left(T \mid y^t\right) \propto \mathsf{p}\left(T\right) \prod_{b=1}^{K_T} \int \mathsf{p}\left(z_{0,b} \mid T\right) \cdot \\
\prod_{i=1}^{t} \mathsf{p}\left(y_{i,b} \mid T, z_{i,b}, \psi_T\right)^{\mathrm{I_{i,b}}} \mathsf{p}\left(z_{i,b} \mid T, z_{i-1,b}, \psi_T\right) dz^t \\
= \mathsf{p}\left(T\right) \prod_{b=1}^{K_T} \int ((2\pi)^m |W_{0,b}|)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(z_{0,b} - \mu_{0,b})^T W_{0,b}^{-1}(z_{0,b} - \mu_{0,b})\right] \cdot \\
\prod_{i=1}^{t} ((2\pi)^n |V_{t,b}|)^{\frac{\mathrm{I_{i,b}}}{2}} \exp\left[-\frac{1}{2}\,\mathrm{I_{i,b}}(y_{i,b} - H_{t,b} z_{i,b})^T V_{t,b}^{-1}(y_{i,b} - H_{t,b} z_{i,b})\right] \cdot \\
((2\pi)^m |W_{t,b}|)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(z_{i,b} - F_{t,b} z_{i-1,b})^T W_{t,b}^{-1}(z_{i,b} - F_{t,b} z_{i-1,b})\right] dz^t
\end{aligned} \tag{A1.10}$$

Now let $((2\pi)^m |W_{t,b}|)^{-\frac{K_T}{2}}$ be the Gaussian constant for the state variable density such that this constant is raised to the power of all the current leaves at each t and $((2\pi)^n |V_{t,b}|)^{\frac{t \cdot \mathrm{I_{i,b}}}{2}}$ the Gaussian constant for the observations for up to t then

$$k_0 = ((2\pi)^m |W_{0,b}|)^{-\frac{K_T}{2}} \cdot ((2\pi)^m |W_{t,b}|)^{-\frac{K_T}{2}} \cdot ((2\pi)^n |V_{t,b}|)^{\frac{t \cdot \mathrm{I_{i,b}}}{2}}$$

is a constant that can be removed from the derivation. The focus now on:

$$\begin{aligned}
k_0 \cdot \int \exp\left[-\frac{1}{2}(z_{0,b} - \mu_{0,b})^T W_{0,b}^{-1}(z_{0,b} - \mu_{0,b})\right] \\
\prod_{i=1}^{t} \exp\left[-\frac{1}{2}\,\mathrm{I_{i,b}}(y_{i,b} - H_{t,b} z_{i,b})^T V_{t,b}^{-1}(y_{i,b} - H_{t,b} z_{i,b})\right] \cdot \\
\exp\left[-\frac{1}{2}(z_{i,b} - F_{t,b} z_{i-1,b})^T W_{t,b}^{-1}(z_{i,b} - F_{t,b} z_{i-1,b})\right] dz^t
\end{aligned} \tag{A1.11}$$

which is an arbitrary leaf marginal density from 0 to t. Thus is it clear that the stochastic process that defines the tree filter is a product of leaf filter trajectories.

Writing out each term of the integral of Equation (A1.11) and bringing all terms

under the exponential provides:

$$
\begin{aligned}
k_0 \cdot \int \exp\Bigg[-\frac{1}{2}\Bigg(& (z_{0,b}-\mu_{0,b})^T W_{b,0}^{-1}(z_{0,b}-\mu_{0,b}) + \\
& \mathrm{I}_{1,\mathrm{b}} \left(y_{1,b}-H_{1,b}z_{1,b}\right)^T V_{1,b}^{-1} \left(y_{1,b}-H_{1,b}z_{1,b}\right) + \\
& \quad (z_{1,b}-F_{1,b}z_{0,b})^T W_{1,b}^{-1}(z_{1,b}-F_{1,b}z_{0,b}) + \\
& \ldots\ldots\ldots\ldots + \\
& \mathrm{I}_{\mathrm{t}\text{-}1,\mathrm{b}} \left(y_{t-1,b}-H_{t-1,b}z_{t-1,b}\right)^T V_{t-1,b}^{-1} \left(y_{t-1,b}-H_{t-1,b}z_{t-1,b}\right) + \\
& \qquad\qquad (z_{t-1,b}-F_{t-1,b}z_{t-2,b})^T W_{t-1,b}^{-1}(z_{t-1,b}-F_{t-1,b}z_{t-2,b}) + \\
& \mathrm{I}_{\mathrm{t},\mathrm{b}} \left(y_{t,b}-H_{t,b}z_{t,b}\right)^T V_{t,b}^{-1}(y_{t,b}-H_{t,b}z_{t,b}) + \\
& \quad (z_{t,b}-F_{t,b}z_{t-1,b})^T W_{t,b}^{-1}(z_{t,b}-F_{t,b}z_{t-1,b}) dz^t.
\end{aligned}
$$

Further expanding each of the quadratic forms and bearing mind that because all covariance matrices are assumed symmetric, positive semi-definite, their inverses are too and hence:

$$
\begin{aligned}
k_0 \cdot \int \exp\Bigg[-\frac{1}{2}\Bigg(& z_{0,b}^T W_{0,b}^{-1} z_{0,b} - 2\mu_{0,b}^T W_{0,b}^{-1} z_{0,b} + \mu_{0,b}^T W_{0,b}^{-1}\mu_{0,b} + \\
& \mathrm{I}_{1,\mathrm{b}} \left(y_{1,b}^T V_{1,b}^{-1} y_{1,b} - 2z_{1,b}^T V_{1,b}^{-1} H_{1,b} y_{1,b} + z_{1,b}^T H_{1,b}^T V_{1,b}^{-1} H_{1,b} z_{1,b} \right) + \\
& \quad z_{1,b}^T W_{1,b}^{-1} z_{1,b} - 2z_{1,b}^T W_{1,b}^{-1} F_{1,b} z_{0,b} + z_{0,b}^T F_{1,b}^T W_{1,b}^{-1} F_{1,b} z_{0,b} - \\
& \ldots\ldots\ldots\ldots + \\
& \mathrm{I}_{\mathrm{t}\text{-}1,\mathrm{b}} \left(y_{t-1,b}^T V_{t-1,b}^{-1} y_{t-1,b} - 2z_{t-1,b}^T V_{t-1,b}^{-1} H_b y_{t-1,b} + z_{t-1,b}^T H_{t-1,b}^T V_{t-1,b}^{-1} H_{t-1,b} z_{t-1,b} \right) + \\
& \quad z_{t-1,b}^T W_{t-1,b}^{-1} z_{t-1,b} - 2z_{t-1,b}^T W_{t-1,b}^{-1} F_{t-1,b} z_{t-2,b} + z_{t-2,b}^T F_{t-1,b}^T W_{t-1,b}^{-1} F_{t-1,b} z_{t-2,b} - \\
& \mathrm{I}_{\mathrm{t},\mathrm{b}} \left(y_{t,b}^T V_{t,b}^{-1} y_{t,b} - 2z_{t,b}^T V_{t,b}^{-1} H_{t,b} y_{t,b} + z_{t,b}^T H_{t,b}^T V_{t,b}^{-1} H_{t,b} z_{t,b} \right) + \\
& \quad z_{t,b}^T W_{t,b}^{-1} z_{t,b} - 2z_{t,b}^T W_{t,b}^{-1} F_{t,b} z_{t-1,b} + z_{t-1,b}^T F_{t,b}^T W_{t,b}^{-1} F_{t,b} z_{t-1,b}\, dz^t
\end{aligned}
$$

Taking out terms that are not marginalised over for any z_t gives a new constant term independent of the latent variable for all t:

$$
k_1 = k_0 \cdot \exp\left[-\frac{1}{2}\left(\mu_{0,b}^T W_{0,b}^{-1} \mu_{0,b} + \sum_{i=1}^{t} \mathrm{I}_{\mathrm{i},\mathrm{b}}\, y_{i,b}^T V_{i,b}^{-1} y_{i,b} \right) \right]
$$

Then rearranging by lower t index order in the z terms gives:

$$
\begin{aligned}
k_1 \cdot \int \exp\Bigg[-\frac{1}{2}\Bigg(& z_{0,b}^T W_{0,b}^{-1} z_{0,b} + z_{0,b}^T F_{1,b}^T W_{1,b}^{-1} F_{1,b} z_{0,b} - 2\mu_{0,b}^T W_{0,b}^{-1} z_{0,b} - 2 z_{1,b}^T W_{1,b}^{-1} F_{1,b} z_{0,b} + \\
& \mathrm{I}_{1,\mathrm{b}} \left(-2 z_{1,b}^T V_{1,b}^{-1} H_{1,b} y_{1,b} + z_{1,b}^T H_{1,b}^T V_{1,b}^{-1} H_{1,b} z_{1,b} \right) + \\
& \quad z_{1,b}^T W_{1,b}^{-1} z_{1,b} + z_{1,b}^T F_{2,b}^T W_{2,b}^{-1} F_{2,b} z_{1,b} - 2 z_{2,b}^T W_{2,b}^{-1} F_{2,b} z_{1,b} - \\
& \ldots\ldots\ldots\ldots + \\
& \mathrm{I}_{\text{t-1,b}} \left(-2 z_{t-1,b}^T V_{t-1,b}^{-1} H_{t-1,b} y_{t-1,b} + z_{t-1,b}^T H_{t-1,b}^T V_{t-1,b}^{-1} H_{t-1,b} z_{t-1,b} \right) + \\
& \quad z_{t-1,b}^T W_{t-1,b}^{-1} z_{t-1,b} + z_{t-1,b}^T F_{t,b}^T W_{t,b}^{-1} F_{t,b} z_{t-1,b} - 2 z_{t,b}^T W_{t,b}^{-1} F_{t,b} z_{t-1,b} - \\
& \mathrm{I}_{\mathrm{t,b}} \left(-2 z_{t,b}^T V_{t,b}^{-1} H_{t,b} y_{t,b} + z_{t,b}^T H_{t,b}^T V_{t,b}^{-1} H_{t,b} z_{t,b} \right) + z_{t,b}^T W_{t,b}^{-1} z_{t,b} \Bigg] dz^t
\end{aligned}
$$

and after a bit more rearranging and collecting like z terms with lower index orders taking precedence gives:

$$
\begin{aligned}
k_1 \cdot \int \exp\Bigg[-\frac{1}{2}\Bigg(& z_{0,b}^T \Bigg(\underbrace{W_{0,b}^{-1} + F_{1,b}^T W_{1,b}^{-1} F_{1,b}}_{\left(A_{0,b}\right)} \Bigg) z_{0,b} - 2 \Bigg(\underbrace{W_{0,b}^{-1} \mu_{0,b} + z_{1,b}^T W_{1,b}^{-1} F_{1,b}}_{\left(b_{0,b}^T\right)} \Bigg) z_{0,b} + \\
& z_{1,b}^T \Bigg(\underbrace{W_{1,b}^{-1} + F_{2,b}^{-1} W_{2,b}^{-1} F_{2,b} + \mathrm{I}_{1,\mathrm{b}}\, H_{1,b}^{-1} V_{1,b}^{-1} H_{1,b}}_{\left(A_{1,b}\right)} \Bigg) z_{1,b} - \\
& \qquad 2 \Bigg(\underbrace{\mathrm{I}_{1,\mathrm{b}}\, V_{1,b}^{-1} H_{1,b} y_{1,b} + z_{2,b}^T W_{2,b}^{-1} F_{2,b}}_{\left(b_{1,b}^T\right)} \Bigg) z_{1,b} + \\
& \ldots\ldots\ldots\ldots + \\
& z_{t-1,b}^T \Bigg(\underbrace{W_{t-1,b}^{-1} + F_{t,b}^{-1} W_{t,b}^{-1} F_{t,b} + \mathrm{I}_{\text{t-1,b}}\, H_{t-1,b}^{-1} V_{t-1,b}^{-1} H_{t-1,b}}_{\left(A_{t-1,b}\right)} \Bigg) z_{t-1,b} - \\
& \qquad 2 \Bigg(\underbrace{\mathrm{I}_{\text{t-1,b}}\, V_{t-1,b}^{-1} H_{t-1,b} y_{t-1,b} + z_{t,b}^T W_{t,b}^{-1} F_{t,b}}_{\left(b_{t-1,b}^T\right)} \Bigg) z_{t-1,b} + \\
& z_{t,b}^T \Bigg(\underbrace{W_{t,b}^{-1} + \mathrm{I}_{\mathrm{t,b}}\, H_{t,b}^{-1} V_b^{-1} H_{t,b}}_{\left(A_{t,b}\right)} \Bigg) z_{t,b} - 2 \Bigg(\underbrace{\mathrm{I}_{\mathrm{t,b}}\, V_{t,b}^{-1} H_{t,b} y_{t-1,b}}_{\left(b_{t,b}^T\right)} \Bigg) z_{t,b} \Bigg) \Bigg] dz^t
\end{aligned}
\qquad \text{(A1.12)}
$$

Notice that every term involving the products $W_{t,b}^{-1} F_{t,b}$ or $F_{t,b}^{-1} W_{t,b}^{-1} F_{t,b}$ have been

moved up an index. That is, they are the cross terms that result from the quadratic product of the observation equation and the prediction equation. At each index t the square of the current state A_t is only a function of the random update and the current state variance-covariance matrix.

Now for each $j = 0, \ldots, t$ in Equation (A1.12) there is an equation of the form $z_j^T A_j z_j - 2b_j^T z_j$ which means we can complete the square and integrate for each j as follows:

$$\int_{-\infty}^{\infty} \exp\left[-\frac{1}{2}\left(z_{j,b}^T A_{j,b} z_{j,b} - 2b_{j,b}^T z_{j,b} \right) \right] dz_{j,b}$$
$$= ((2\pi)^m |A_{j,b}|)^{\frac{1}{2}} \exp\left[\frac{1}{2} b_{j,b}^T A_{j,b}^{-1} b_{j,b} \right]. \tag{A1.13}$$

So let $A_{0,b} = \left(W_{0,b}^{-1} + F_{1,b}^T W_{1,b}^{-1} F_{1,b} \right)$ and $b_{0,b}^T = \left(W_{1,b}^{-1} \mu_{0,b} + z_{1,b}^T W_{1,b}^{-1} F_{1,b} \right)$ then we integrate over $z_{0,b}$. This is possible because the integral function is a linear operator and, although b_0 contains a $z_{1,b}$ term, from the point of view of the integral at $z_{0,b}$, this term is constant. However this $z_{0,b}$ term needs to be reintroduced into the integral at the next iteration so that its influence on the marginal can be included in that step. The above reasoning is used for every iteration of the integral in Equation (A1.12) from $j = 0$ to $j = t - 1$. At $j = t$ all terms have the same index order.

Also notice that for all j, $A_{j,b}$ is dependent on whether a particular leaf is chosen via $\mathrm{I_{t,b}}\, H_{t,b}^{-1} V_{t,b}^{-1} H_{t,b}$. Thus if $\psi_b = (V_{t,b}, H_{t,b}, W_{t,b}, F_{t,b})$ are known (but possibly different) for each b then A_j is random function of the choice of leaf.

The constant term $((2\pi)^m |A_{j,b}|)^{\frac{1}{2}}$ will occur at every j so

$$c_0 = ((2\pi)^m |W_{0,b}|)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}\left(\mu_{0,b}^T W_{0,b}^{-1} \mu_{0,b} \right) \right] ((2\pi)^m |A_{0,b}|)^{\frac{1}{2}} \text{ then}$$
$$c_1 = c_0 \cdot ((2\pi)^m |W_{1,b}|)^{-\frac{1}{2}} ((2\pi)^n |V_{1,b}|)^{\frac{\cdot \mathrm{I_{1,b}}}{2}} \exp\left[-\frac{1}{2}\left(\mathrm{I_{1,b}}\, y_{1,b}^T V_{1,b}^{-1} y_{1,b} \right) \right] ((2\pi)^m |A_{1,b}|)^{\frac{1}{2}}$$

and

$$c_j = c_{j-1} \cdot ((2\pi)^m |W_{j,b}|)^{-\frac{1}{2}} ((2\pi)^n |V_{j,b}|)^{\frac{\cdot \mathrm{I_{j,b}}}{2}} \exp\left[-\frac{1}{2}\left(\mathrm{I_{j,b}}\, y_{j,b}^T V_{j,b}^{-1} y_{j,b} \right) \right] ((2\pi)^m |A_{j,b}|)^{\frac{1}{2}}$$
$$j = 0, \ldots, t \tag{A1.14}$$

is a recursive constant term that is independent of Z^t and for each t is randomly updated by some observation if a leaf is selected at that t. Note that so far c_j is a random variable that is dependent on the chance that a particular leaf in a

tree is updated and the random size of the update provided by the data $y_{j,b}$. In other words, if ψ_b is known for all b then $c_j(T)$ is a random function of the tree and size of the measurement [1].

Focusing on the expansion of $\exp\left[\frac{1}{2}b_{0,b}^T A_{0,b}^{-1} b_{0,b}\right]$ and also letting $d_{0,b} = W_{0,b}^{-1}\mu_{0,b}$:

$$\begin{aligned}
&\exp\left[\frac{1}{2}\left(d_{0,b} + z_{1,b}W_{1,b}^{-1}F_{1,b}\right)^T A_{0,b}^{-1}\left(d_{0,b} + z_{1,b}W_{1,b}^{-1}F_{1,b}\right)\right] \\
&\quad = \exp\left[\frac{1}{2}\left(d_{0,b}^T A_{0,b}^{-1} d_{0,b} + 2d_{0,b}^T A_{0,b}^{-1} W_{1,b}^{-1} F_{1,b} z_{1,b} + z_{1,b}^T F_{1,b}^T W_{1,b}^{-1} A_{0,b}^{-1} W_{1,b}^{-1} F_{1,b} z_{1,b}\right)\right] \\
&\quad = \exp\left[\frac{1}{2}\left(d_{0,b}^T A_{0,b}^{-1} d_{0,b}\right)\right]\cdot \\
&\qquad \exp\left[\frac{1}{2}\left(2\underbrace{d_{0,b}^T A_{0,b}^{-1} W_{1,b}^{-1} F_{1,b}}_{w_0} z_{1,b} + z_{1,b}^T \underbrace{F_{1,b}^T W_{1,b}^{-1}}_{v_0^T} A_{0,b}^{-1} \underbrace{W_{1,b}^{-1} F_{1,b}}_{v_0} z_{1,b}\right)\right] \\
&\quad = \exp\left[\frac{1}{2}\left(d_{0,b}^T A_{0,b}^{-1} d_{0,b}\right)\right] \exp\left[\frac{1}{2}\left(2w_0 z_{1,b} + z_{1,b}^T v_0^T A_{0,b}^{-1} v_0 z_{1,b}\right)\right] \qquad \text{(A1.15)}
\end{aligned}$$

The terms v_0 and w_0 are still coefficients of the variable $z_{1,b}$ and these must be returned to the integral for next stage of the recursive calculation. Doing so presents:

$$\begin{aligned}
&c_0 \exp\left[\frac{1}{2}\left(d_{0,b}^T A_{0,b}^{-1} d_{0,b}\right)\right] \int \exp\Bigg[-\frac{1}{2}\Bigg(-2w_0 z_{1,b} - z_{1,b}^T v_0^T A_{0,b} v_0 z_{1,b} + \\
&\quad z_{1,b}^T\left(W_{1,b}^{-1} + F_{2,b}^{-1} W_{2,b}^{-1} F_{2,b} + \mathrm{I}_{1,\mathrm{b}}\, H_{1,b}^{-1} V_{1,b}^{-1} H_{1,b}\right) z_{1,b} - \\
&\qquad\qquad 2\left(\mathrm{I}_{1,\mathrm{b}}\, V_{1,b}^{-1} H_{1,b} y_{1,b} + z_{2,b}^T W_{2,b}^{-1} F_{2,b}\right) z_{1,b} + \\
&\quad \ldots\ldots\ldots\ldots\Bigg)\Bigg] dz^{1:t}
\end{aligned}$$

which after expansion and rearrangement gives:

$$\begin{aligned}
&c_0 \exp\left[\frac{1}{2}\left(d_{0,b}^T A_{0,b}^{-1} d_{0,b}\right)\right] \\
&\qquad \int \exp\Bigg[-\frac{1}{2}\Bigg(z_{1,b}^T\left(W_{1,b}^{-1} + F_{2,b}^{-1} W_{2,b}^{-1} F_{2,b} + \mathrm{I}_{1,\mathrm{b}}\, H_{1,b}^{-1} V_{1,b}^{-1} H_{1,b} - v_0^T A_{0,b}^{-1} v_0\right) z_{1,b} - \\
&\qquad 2\left(\mathrm{I}_{1,\mathrm{b}}\, V_{1,b}^{-1} H_{1,b} y_{1,b} + z_{2,b}^T W_{2,b}^{-1} F_{2,b} - w_0\right) z_{1,b} + \ldots\ldots\ldots\ldots\Bigg)\Bigg] dz^{1:t}
\end{aligned} \qquad \text{(A1.16)}$$

[1] To be clear here in this appendix c_j refers to a constant value and is not to be confused with c_j in the main body of the text which refers to a threshold value.

Letting:

$$
\begin{aligned}
A_{1,b}^{-1} &= \left(W_{1,b}^{-1} + F_{2,b}^{-1} W_{2,b}^{-1} F_{2,b} + \mathrm{I}_{1,\mathrm{b}}\, H_{1,b}^{-1} V_{1,b}^{-1} H_{1,b} - v_0^T A_{0,b}^{-1} v_0\right) \\
b_{1,b}^T &= \left(\underbrace{\mathrm{I}_{1,\mathrm{b}}\, V_{1,b}^{-1} H_{1,b} y_{1,b}}_{u_1} + z_{2,b}^T \underbrace{W_{2,b}^{-1} F_{2,b}}_{v_1} - w_0\right) \\
&= \left(u_1 - w_0 + z_{2,b}^T v_1\right) \qquad \text{(A1.17)}
\end{aligned}
$$

then completing the square as in Equation (A1.13) and following the same procedure as in Equation (A1.15) provides:

$$
\begin{aligned}
&\exp\left[\frac{1}{2}\left(\left(u_1 - w_0 + z_{2,b}^T v_1\right)^T A_{1,b}^{-1}\left(u_1 - w_0 + z_{2,b}^T v_1\right)\right)\right] \\
&= \exp\left[\frac{1}{2}\left(u_1^T A_{1,b}^{-1} u_1 - 2u_1^T A_{1,b}^{-1} w_0 + w_0^T A_{1,b}^{-1} w_0 + \right.\right. \\
&\qquad\qquad \left.\left. z_{2,b}^T v_1^T A_{1,b}^{-1} v_1 z_{2,b} + 2z_{2,b}(v_1^T A_{1,b}^{-1} u_1 - v_1^T A_{1,b}^{-1} w_1)\right)\right] \\
&= \exp\left[\frac{1}{2}\left(u_1 - w_0)^T A_{1,b}^{-1}(u_1 - w_0) + z_{2,b}^T v_1^T A_{1,b}^{-1} v_1 z_{2,b} + 2z_{2,b} v_1^T A_{1,b}^{-1}(u_1 - w_0)\right)\right] \\
&= \exp\left[\frac{1}{2}\left(u_1 - w_0)^T A_{1,b}^{-1}(u_1 - w_0)\right)\right] \cdot \\
&\qquad \exp\left[\frac{1}{2}\left(z_{2,b}^T v_1^T A_{1,b}^{-1} v_1 z_{2,b} + 2z_{2,b} v_1^T A_{1,b}^{-1}(u_1 - w_0)\right)\right] \text{ and so} \\
d_{1,b} &= (u_1 - w_0) = (\mathrm{I}_{1,\mathrm{b}}\, V_{1,b}^{-1} H_{1,b} y_{1,b} - d_{0,b}^T A_{0,b} W_{1,b}^{-1} F_{1,b}) \text{ and } w_1 = v_1^T A_{1,b}^{-1} d_1
\end{aligned}
\qquad \text{(A1.18)}
$$

Reinserting terms that are coefficients of $z_{2,b}$ and starting from Equation (A1.16) gives:

$$
\begin{aligned}
&c_1 \exp\left[\frac{1}{2}\left(\sum_{j=0}^{1} d_{j,b}^T A_{j,b}^{-1} d_{j,b}\right)\right] \cdot \\
&\int \exp\left[-\frac{1}{2}\left(z_{2,b}^T\left(W_{2,b}^{-1} + F_{3,b}^T W_{3,b}^{-1} F_{3,b} + \mathrm{I}_{2,\mathrm{b}}\, H_{2,b}^T V_{2,b}^{-1} H_{2,b} - v_1^T A_{1,b}^{-1} v_1\right) z_{2,b} - \right.\right. \\
&\qquad \left.\left. 2\left(\mathrm{I}_{2,\mathrm{b}}\, V_{2,b}^{-1} H_{2,b} y_{2,b} + z_{2,b}^T W_{3,b}^{-1} F_{3,b} - w_1\right) z_{2,b} + \ldots\ldots\ldots\ldots\right)\right] dz^{2:t} \\
&= c_1 \exp\left[\frac{1}{2}\left(\sum_{j=0}^{1} d_{j,b}^T A_{j,b}^{-1} d_{j,b}\right)\right] \int \exp\left[-\frac{1}{2}\left(z_{2,b}^T A_{2,b} z_{2,b} - 2\left(u_2 + z_{2,b}^T v_2 - w_1\right) z_{2,b} + \ldots\right)\right] dz^{2:t}
\end{aligned}
\qquad \text{(A1.19)}
$$

and from this point forward up to $t-1$ the recursion is clear:

$$\begin{aligned}
A_{0,b} &= \left(W_{0,b}^{-1} + F_{1,b}^{T} W_{1,b}^{-1} F_{1,b}\right) \\
d_{0,b} &= W_{0,b}^{-1} \mu_{0,b} \\
A_{t-1,b} &= \left(W_{t-1,b}^{-1} + F_{t,b}^{T} W_{t,b}^{-1} F_{t,b} + \mathrm{I_{t\text{-}1,b}}\, H_{t-1,b}^{T} V_{t-1,b}^{-1} H_{t-1} - v_{t-2}^{T} A_{t-2,b}^{-1} v_{t-2}\right) \\
&= \left(W_{t-1,b}^{-1} + F_{t,b}^{T} W_{t,b}^{-1} F_{t,b} + \mathrm{I_{t\text{-}1,b}}\, H_{t-1,b}^{T} V_{t-1,b}^{-1} H_{t-1,b} - F_{t-1,b}^{T} W_{t-1,b}^{-1} A_{t-2,b}^{-1} W_{t-1,b}^{-1} F_{t-1,b}\right) \\
d_{t-1,b} &= (u_{t-1} - w_{t-2}) \\
&= (\mathrm{I_{t\text{-}1,b}}\, V_{t-1,b}^{-1} H_{t-1} y_{t-1,b} - d_{t-2,b}^{T} A_{t-2,b}^{-1} W_{t-1,b}^{-1} F_{t-1,b})
\end{aligned} \tag{A1.20}$$

For the t^{th} instance looking back at Equation (A1.12) one notices that both the $\textcircled{A_t}$ and $\textcircled{b_t^T}$ parts are missing terms. Thus

$$\begin{aligned}
& c_{t-1} \exp\left[\frac{1}{2}\left(\sum_{j=0}^{t-1} d_{j,b}^{T} A_{j,b}^{-1} d_{j,b}\right)\right] \cdot \\
& \int \exp\left[-\frac{1}{2}\left(z_{t,b}^{T} \left(W_{t,b}^{-1} + \mathrm{I_{t,b}}\, H_{t,b}^{T} V_{t,b}^{-1} H_{t,b} - v_{t-1}^{T} A_{t-1,b}^{-1} v_{t-1}\right) z_{t,b} - \right.\right. \\
& \qquad \left.\left. 2\left(\mathrm{I_{2,b}}\, V_{t,b}^{-1} H_{t,b} y_{2,b} - w_1\right) z_{2,b}\right)\right] dz_t \\
=& c_{t-1} \exp\left[\frac{1}{2}\left(\sum_{j=0}^{t-1} d_{j,b}^{T} A_{j,b}^{-1} d_{j,b}\right)\right] \int \exp\left[-\frac{1}{2}\left(z_{t,b}^{T} A_{t,b}^{-1} z_{t,b} - 2\left(u_t - w_{t-1}\right) z_{t,b}\right)\right] dz_t
\end{aligned} \tag{A1.21}$$

which after completing the square for the final time yields:

$$\begin{aligned}
& \int \mathsf{p}\left(\, z_{0,b}\,\right) \prod_{i=1}^{t} \mathsf{p}\left(\, y_{i,b} \mid z_{i,b}, \psi_T\,\right)^{\mathrm{I_{i,b}}} \mathsf{p}\left(\, z_{i,b} \mid z_{i-1,b}, T, \psi_T\,\right) dz^t \\
&\qquad = c_t \exp\left[\frac{1}{2}\left(\sum_{j=0}^{t} d_{j,b}^{T} A_{j,b}^{-1} d_{j,b}\right)\right] \\
&\qquad = c_{t-1} ((2\pi)^m |W_{t,b}|)^{-\frac{1}{2}} ((2\pi)^n |V_{t,b}|)^{\frac{\cdot \mathrm{I_{t,b}}}{2}} \exp\left[-\frac{1}{2}\left(\mathrm{I_{t,b}}\, y_{t,b}^{T} V_{t,b}^{-1} y_{t,b}\right)\right] \cdot \\
&\qquad\qquad ((2\pi)^m |A_{t,b}|)^{\frac{1}{2}} \exp\left[\frac{1}{2}\left(\sum_{j=0}^{t} d_{j,b}^{T} A_{j,b}^{-1} d_{j,b}\right)\right]
\end{aligned} \tag{A1.22}$$

where

$$\begin{aligned}
A_{t,b} &= \left(W_{t,b}^{-1} + \mathrm{I_{t,b}}\, H_{t,b}^{T} V_{t,b}^{-1} H_{t,b} - v_{t-1}^{T} A_{t-1,b}^{-1} v_{t-1}\right) \text{ and} \\
b_{t,b}^{T} &= d_{t,b}^{T} = (u_t - w_{t-1}) = (\mathrm{I_{t,b}}\, V_{t,b}^{-1} H_{t,b} y_{t,b} - d_{t-1,b}^{T} A_{t-1,b}^{-1} W_{t,b}^{-1} F_{t,b})
\end{aligned} \tag{A1.23}$$

Comparing Equation (A1.20) and Equation (A1.23) there is a single term difference between $A_{t-1,b}$ and $A_{t,b}$, specifically the term $F_{t,b}^T W_{t,b}^{-1} F_{t,b}$ which occurs in the $(t-1)^{st}$ term although this has t index. This makes sense because at every t there are two steps to the calculation, the prediction step, which occurs at every leaf, and the update step. The final two equations (A1.23) make it clear that there is no "borrowing from the future": every term in the integral is known at index t. Thus the recursive calculation for the parameters $A_{i,b}$ and $d_{i,b}$ are as follows:

$$
\begin{array}{l|l}
t=0 & \begin{aligned} A_{0,b} &= W_{0,b}^{-1} \\ d_{0,b} &= W_{0,b}^{-1}\mu_{0,b} \end{aligned} \\
\\
t=1 & \begin{aligned} A_{0,b}^{*} &= \left(A_{0,b} + F_{1,b}^T W_{1,b}^{-1} F_{1,b}\right) \\ A_{1,b} &= \left(W_{1,b}^{-1} + \mathrm{I}_{1,\mathrm{b}}\, H_{1,b}^T V_{1,b}^{-1} H_{1,b} - F_{1,b}^T W_{1,b}^{-1} A_{0,b}^{-1*} W_{1,b}^{-1} F_{1,b}\right) \\ b_{1,b}^T &= \left(\mathrm{I}_{1,\mathrm{b}}\, V_{1,b}^{-1} H_{1,b} y_{1,b} - d_{0,b}^T A_{0,b}^{-1*} W_{1,b}^{-1} F_{1,b}\right) \end{aligned} \\
\\
t=2 & \begin{aligned} A_{1,b}^{*} &= \left(A_{1,b} + F_{2,b}^T W_{2,b}^{-1} F_{2,b}\right) \\ A_{2,b} &= \left(W_{2,b}^{-1} + \mathrm{I}_{2,\mathrm{b}}\, H_{2,b}^T V_{2,b}^{-1} H_{2,b} - F_{2,b}^T W_{2,b}^{-1} A_{1,b}^{-1*} W_{2,b}^{-1} F_{2,b}\right) \\ b_{2,b}^T &= \left(\mathrm{I}_{2,\mathrm{b}}\, V_{2,b}^{-1} H_{2,b} y_{2,b} - d_{1,b}^T A_{1,b}^{-1*} W_{2,b}^{-1} F_{2,b}\right) \end{aligned} \\
\\
\vdots & \vdots
\end{array}
\tag{A1.24}
$$

which requires that the terms $A_{i,b}$ and $d_{i,b}$ are maintained for the entire run of the algorithm and that at every t two matrix inversions are required for the marginal calculation, one for $A_{i,b}$ and one for $A_{i,b}^{*}$.

Based on the recursive integral of Equation (A1.22) the calculation of the log marginal, ℓm_t, for the leaf from the index that the leaf is generated (initially or by a leaf move) to t is:

$$
\begin{array}{l|l}
t = 0 & lm_0 = -\frac{1}{2}\left(\ln((2\pi)^m \mid \mathrm{W}_{0,\mathrm{b}} \mid) + \mu_{0,b}^T W_{0,b}^{-1} \mu_{0,b}\right) + \\
 & \quad \frac{1}{2}\left(\ln((2\pi)^m \mid \mathrm{A}_{0,\mathrm{b}} \mid) + d_{0,b}^T A_{0,b}^{-1} d_{0,b}\right) \\
 & \\
t = 1 & \ell m_1 = \ell m_0 - \frac{1}{2}\left(\ln((2\pi)^m \mid \mathrm{W}_{1,\mathrm{b}} \mid)\right) - \\
 & \quad \frac{1}{2}\left(\mathrm{I}_{1,\mathrm{b}}(\ln((2\pi)^n \mid \mathrm{V}_{1,\mathrm{b}} \mid) + y_{1,b}^T V_{1,b}^{-1} y_{1,b})\right) + \\
 & \quad \frac{1}{2}\left(\ln((2\pi)^m \mid \mathrm{A}_{1,\mathrm{b}} \mid + d_{1,b}^T A_{1,b}^{-1} d_{1,b})\right) \\
 & \\
t = 2 & \ell m_2 = \ell m_2 - \frac{1}{2}\left(\ln((2\pi)^m \mid \mathrm{W}_{2,\mathrm{b}} \mid)\right) - \\
 & \quad \frac{1}{2}\left(\mathrm{I}_{2,\mathrm{b}}(\ln((2\pi)^n \mid \mathrm{V}_{2,\mathrm{b}} \mid) + y_{2,b}^T V_{2,b}^{-1} y_{2,b})\right) + \\
 & \quad \frac{1}{2}\left(\ln((2\pi)^m \mid \mathrm{A}_{2,\mathrm{b}} \mid + d_{2,b}^T A_{2,b}^{-1} d_{2,b})\right) \qquad \text{(A1.25)} \\
\vdots &
\end{array}
$$

To calculate the tree marginal likelihood at each t, return to equation Equation (A1.6) to see that the tree marginal is the product of extant leaf marginals at t conditional on the tree and its prior probability.

Let $\ell m_{t,b}$ be the log marginal of leaf b at index t. Then the log marginal of the tree at each t is:

$$
\begin{aligned}
\ell\, \mathsf{p}\left(T \mid y^t\right) &\propto \ell\, \mathsf{p}\left(T\right) + \\
&\quad \sum_{b=1}^{K_T(t)} \ell\left(\int \mathsf{p}\left(z_{0,b} \mid T\right) \prod_{i=1}^{t} \mathsf{p}\left(y_{i,b} \mid T, z_{i,b}, \psi_b\right)^{\mathrm{I}_{\mathrm{i,b}}} \mathsf{p}\left(z_{i,b} \mid T, z_{i-1,b}, \psi_b\right) dz^t\right) \\
&= \ell\, \mathsf{p}\left(T\right) + \sum_{b=1}^{K_T(t)} \ell m_{t,b} \qquad \text{(A1.26)}
\end{aligned}
$$

where $K_T(t)$ emphasizes that the number of leaves at each t is possibly different but this randomness has been accounted for by the marginalization over the latent space sample denoted by the leaves.

Thus the log posterior for a particular tree at index t conditional on the data from $t = 0$ to $t = t$ is the log of the prior tree model sampled at time t, based on its current structure and rules, added to the accumulative evidence provided by the sum of the log leaf latent sequences.

The smaller a tree is, the more concentrated the support for likelihood should be

because at each t there are fewer latent models to update. Similarly, the larger a tree is the greater the uncertainty in the tree model because a single data point can only update one of many possible leaf models. The tree prior provides a penalty on the tree model that should limit the growth of the tree.

A1.1.1 One dimensional marginal likelihood

Using the Equations (A1.20) and (A1.23) to (A1.25) a one dimensional version of the recursion of the marginal parameters, $A_{t,b} = a_{t,b}$ and $d_{t,b} = d_{t,b}$ are:

$$
\begin{aligned}
a_{0,b} &= \frac{1}{w_{0,b}^2} \\
d_{0,b} &= \frac{\mu_{0,b}}{w_{0,b}^2} \\
a_{t-1,b}^* &= a_{t-1,b} + \frac{f_{t,b}^2}{w_{t,b}^2} && \text{(A1.27)} \\
a_{t,b} &= \frac{1}{w_{t,b}^2} + \mathrm{I}_{\mathrm{t,b}} \frac{h_{t,b}^2}{v_{t,b}^2} - \frac{f_{t,b}^2}{a_{t-1,b}^* w_{t,b}^4} && \text{(A1.28)} \\
d_{t,b} &= \mathrm{I}_{\mathrm{t,b}} \frac{h_{t,b}}{v_{t,b}^2} y_{t,b} - \frac{f_{t,b}}{a_{t-1,b}^* w_{t,b}^2} d_{t-1,b} && \text{(A1.29)}
\end{aligned}
$$

and the log marginal for each leaf b up to instance t is:

$$
\begin{array}{l|l}
t = 0 & \ell m_0 = -\frac{1}{2}\left(\ln(2\pi w_{0,b}^2) + \frac{\mu_{0,b}^2}{w_{0,b}^2} - \ln(2\pi a_{0,b}) - \frac{d_{0,b}^2}{a_{0,b}}\right) \quad \text{(A1.30)} \\
t = 1 & \ell m_1 = \ell m_0 - \frac{1}{2}\left(\ln(2\pi w_{1,b}^2) + \mathrm{I}_{1,\mathrm{b}}(\ln(2\pi v_{1,b}^2) + \frac{y_{1,b}^2}{v_{1,b}^2}) - \ln(2\pi a_{1,b}) - \frac{d_{1,b}^2}{a_{1,b}}\right) \\
t = 2 & \ell m_2 = \ell m_1 - \frac{1}{2}\left(\ln(2\pi w_{2,b}^2) + \mathrm{I}_{2,\mathrm{b}}(\ln(2\pi v_{2,b}^2) + \frac{y_{1,b}^2}{v_{2,b}^2}) - \ln(2\pi a_{2,b}) - \frac{d_{2,b}^2}{a_{2,b}}\right) \\
\vdots & \vdots
\end{array}
$$

Notice that in Equation (A1.28), $a_{t,b}$ is a random variable that is independent of the data y_t. The randomness of $a_{t,b}$ is derived from the indicator $\mathrm{I}_{\mathrm{t,b}}$ which is a random function of the tree. Of particular interest in this derivation are the extreme cases where $\mathrm{I}_{\mathrm{t,b}} = 1$ or 0 for all t. In each of these cases it is possible to calculate bounds for the these random process $\{a_{t,b}\}$ conditional on some specific ranges of the parameters $\psi_b = (H_b, V_b, F_b, W_b)$. The case that $\mathrm{I}_{\mathrm{t,b}} = 1$ is the upper bound of the random process and $\mathrm{I}_{\mathrm{t,b}} = 0$ is the lower bound of this process and

so reconfiguring Equation (A1.28) gives:

$$\boldsymbol{a}_{t,b} = \begin{cases} \dfrac{1}{w_{t,b}^2} + \dfrac{h_{t,b}^2}{v_{t,b}^2} - \dfrac{f_{t,b}^2}{\boldsymbol{a}_{t-1,b}^* w_{t,b}^4} \text{ if } \mathrm{I_{t,b}} = 1 \; \forall \; t & \text{(A1.31)} \\ \dfrac{1}{w_{t,b}^2} - \dfrac{f_{t,b}^2}{\boldsymbol{a}_{t-1,b}^* w_{t,b}^4} \text{ if } \mathrm{I_{t,b}} = 0 \; \forall \; t & \text{(A1.32)} \end{cases}$$

Assuming that $v^2 = v$ and $w^2 = w$ from Equations (A1.27) and (A1.28) are variances rather than standard deviations then $\boldsymbol{a}_{t,b}$ in Equation (A1.32) can be rewritten as:

$$\begin{aligned} \boldsymbol{a}_{t,b} &= \frac{1}{w_{t,b}} - \frac{f_{t,b}^2}{w_{t,b}^2 \boldsymbol{a}_{t-1,b}^*} = \frac{1}{w_{t,b}} - \frac{f_{t,b}^2}{\boldsymbol{a}_{t-1,b} + \frac{f^2}{w_{t,b}}} = \frac{w \boldsymbol{a}_{t-1,b}}{w_{t,b}^2 \boldsymbol{a}_{t-1,b} + f_{t,b}^2 w_{t,b}^2} \\ &= \frac{\boldsymbol{a}_{t-1,b}}{w_{t,b} \boldsymbol{a}_{t-1,b} + f_{t,b}^2} \end{aligned} \tag{A1.33}$$

and the recursion for this equation is:

$$\begin{aligned} &\boldsymbol{a}_{0,b} = \frac{1}{w_{0,b}}; \;\; \boldsymbol{a}_{1,b} = \frac{\boldsymbol{a}_{0,b}}{w_{1,b}\boldsymbol{a}_{0,b} + f_{t,b}^2}; \;\; \boldsymbol{a}_{2,b} = \frac{\boldsymbol{a}_{0,b}}{w_{2,b}(1 + f_{t,b}^2)\boldsymbol{a}_{0,b} + f_{t,b}^4}; \\ &\boldsymbol{a}_{3,b} = \frac{\boldsymbol{a}_{0,b}}{w_{3,b}(1 + f_{t,b}^2 + f_{t,b}^4)\boldsymbol{a}_{0,b} + f_{t,b}^6}; \;\; \ldots\ldots; \;\; \boldsymbol{a}_{t,b} = \frac{\boldsymbol{a}_{0,b}}{w_{t,b}(\sum_{i=0}^{t-1} f^{2i})\boldsymbol{a}_{0,b} + f^{2t}}; \end{aligned} \tag{A1.34}$$

If one assumes that the streaming event will go on for long enough then $t \to \infty$ and $\sum_{i=0}^{\infty} f^{2i} = 1/(1 - f^2)$.

The recursion for Equation (A1.31) follows that of Equation (A1.33) and can be written as:

$$\begin{aligned} \boldsymbol{a}_{t,b} &= \frac{h_{t,b}^2}{v_{t,b}^2} + \frac{1}{w_{t,b}} - \frac{f_{t,b}^2}{w_{t,b}^2 \boldsymbol{a}_{t-1,b}^*} = \frac{h_{t,b}^2}{v_{t,b}} + \frac{\boldsymbol{a}_{t-1,b}}{w_{t,b}\boldsymbol{a}_{t-1,b} + f_{t,b}^2} = \frac{h_{t,b}^2 \boldsymbol{a}_{t-1,b}(w_{t,b} + \frac{v}{h_{t,b}^2}) + h_{t,b}^2 f_{t,b}^2}{v_{t,b}(w_{t,b}\boldsymbol{a}_{t-1,b} + f_{t,b}^2)} \\ &= \underbrace{\frac{h_{t,b}^2}{v_{t,b}}}_{c_1} \cdot \frac{\boldsymbol{a}_{t-1,b} \overbrace{(w + \frac{v_{t,b}}{h_{t,b}^2})}^{c_2} + f_{t,b}^2}{w_{t,b}\boldsymbol{a}_{t-1,b} + f_{t,b}^2} = \frac{c_1 c_2 \boldsymbol{a}_{t-1,b} + c_1 f_{t,b}^2}{w_{t,b}\boldsymbol{a}_{t-1,b} + f_{t,b}^2} \end{aligned} \tag{A1.35}$$

Key to controlling this upper bound are the values $c_1 c_2 = h^2/v \cdot (w + v/h^2) = h^2 w + v$ and $c_1 = h^2/v$.

A similar process can be followed for $d_{t,b}$ in Equation (A1.29) where:

$$d_{t,b} = \begin{cases} \frac{h_{t,b}}{v_{t,b}} y_{t,b} - \frac{f_{t,b}}{a^*_{t-1,b} w_{t,b}} d_{t-1,b} \text{ if } \mathrm{I_{t,b}} = 1 \; \forall \; t & \text{(A1.36)} \\ - \frac{f_{t,b}}{a^*_{t-1,b} w_{t,b}} d_{t-1,b} \text{ if } \mathrm{I_{t,b}} = 0 \; \forall \; t & \text{(A1.37)} \end{cases}$$

In this case the randomness in the recursion in Equation (A1.29) is based on both $\mathrm{I_{t,b}}$ and $y_t \mid \mathrm{I_{t,b}}$. In the cases in Equations (A1.36) and (A1.37) the randomness induced by the tree is removed so that the only random component of the recursion is the data, y_t, which occurs at every t assuming $\mathrm{I_{t,b}} = 1 \; \forall t$. The objective is to calibrate the effect of the parameters ψ_b given some sample sequence $\{y_t\}$. As a result of the independence of $a_{t,b}$ from the data, which implies that its extreme values or bounds ($\mathrm{I_{t,b}} = 1$ and 0) are known before the streaming exercise begins, it is assumed that the values $a_{t-1,b}$ are constants in both cases, where in the former the value of $a_{t-1,b}$ comes from the recursion of Equation (A1.35) and in the latter from the recursion of Equation (A1.33).

For Equation (A1.37), using the notation u_t and w_t from Equations (A1.15) and (A1.17), the recursion for the lower "marginal residual" is:

$$d_{t,b} = -\frac{f_{t,b}}{a^*_{t-1,b} w_{t,b}} d_{t-1,b} = -\underbrace{\frac{f_{t,b}}{w_{t,b} a_{t-1,b} + f^2_{t,b}}}_{c^0_{t-1,b}} d_{t-1,b} = -d_{t-1,b} c^0_{t-1,b}$$

$$d_{0,b} = \frac{\mu_{0,b}}{W_{0,b}}; \qquad c_{0,b} = f_{0,b}$$

$$d_{1,b} = -w_0 = -d_{0,b} c^0_{0,b}$$

$$d_{2,b} = -w_1 = -(-d_{1,b} c^0_{0,b}) c^0_{1,b} = d_{1,b} c^0_{0,b} c^0_{1,b}$$

$$d_{3,b} = -w_2 = -(d_{2,b} c^0_{2,b}) c^0_{1,b} = -d_{2,b} c^0_{0,b} c^0_{1,b} c^0_{2,b}$$

$$\cdots\cdots\cdots$$

$$d_{t,b} = d_{0,b} \prod_{i=0}^{t} (-1)^i c^0_{i,b} \qquad \text{(A1.38)}$$

For Equation (A1.36) and using the same notation and assumptions used in the

equations in A1.38 the recursion for the upper "marginal residual" is:

$$d_{t,b} = \frac{h_{t,b}}{v_{t,b}} y_t - \underbrace{\frac{f_{t,b}}{w_{t,b} a_{t-1,b} + f_{t,b}^2}}_{c^0_{t-1,b}} d_{t-1,b} = \frac{h_{t,b}}{v_{t,b}} y_t - d_{t-1,b} c^1_{t-1,b}$$

$$d_{0,b} = \frac{\mu_{0,b}}{W_{0,b}}; \qquad c_{0,b} = f_{0,b}$$

$$d_{1,b} = u_1 - w_0 = \frac{h_{1,b}}{v_{1,b}} y_1 - d_{0,b} c^1_{0,b}$$

$$d_{2,b} = u_2 - w_1 = \frac{h_{2,b}}{v_{2,b}} y_2 - d_{1,b} c^1_{1,b}$$

$$= \frac{h_{2,b}}{v_{2,b}} y_2 - \frac{h_{1,b}}{v_{1,b}} y_1 c^1_{1,b} + d_{0,b} c^1_{0,b} c^1_{1,b}$$

$$d_{3,b} = u_3 - w_2 = \frac{h_{3,b}}{v_{3,b}} y_3 - d_{2,b} c^1_{2,b}$$

$$= \frac{h_{3,b}}{v_{3,b}} y_3 - \frac{h_{2,b}}{v_{2,b}} y_2 c^1_{2,b} + \frac{h_{1,b}}{v_{1,b}} y_1 c^1_{1,b} c^1_{2,b} - d_{0,b} c^1_{0,b} c^1_{1,b} c^1_{2,b}$$

$$\cdots\cdots\cdots$$

$$d_{t,b} = \frac{h_{t,b}}{v_{t,b}} \left(\sum_{i=1}^{t} y_i + \sum_{i=1}^{t-1} (-1)^i y_i \prod_{j=1}^{i} c^1_{j,b}\right) + d_{0,b} \prod_{i=0}^{t} (-1)^i c^1_{i,b} \tag{A1.39}$$

To compare the Kalman filter marginal and tree filter marginal it is necessary to get them into a similar form. Let s_t be the innovation variance of the 1-dimensional Kalman filter and $\tilde{y} = y_t - h\hat{\mu}_{t|t-1} = y_t - hf\hat{\mu}_{t-1|t-1}$ be the innovation or pre-fit residual. Then the likelihood of the Kalman filter from 0 to t can be written as:

$$\mathsf{p}\,(\,y \mid \psi\,) = \prod_{i=0}^{t} \mathsf{p}\,(\,y_t \mid y_{t-1}, \ldots, y_0, \psi\,) \tag{A1.40}$$

and the marginal likelihood based on the latent state and using the Markov assumption is:

$$\mathsf{p}\,(\,y \mid \psi\,) = \prod_{i=0}^{t} \int \mathsf{p}\,(\,y_t \mid z_t\,)\,\mathsf{p}\,(\,z_t \mid z_{t-1}, y_{t-1}.\psi\,)\,dz_t$$

$$\mathsf{p}\,(\,y \mid \psi\,) = \prod_{i=0}^{t} N(\tilde{y},\, s_t \mid \psi) \tag{A1.41}$$

where the Equation (A1.41) comes from the derivation of the Kalman filter provided in Meinhold and N. D. Singpurwalla 1983. This equation can be written in

recursive form as:

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(\frac{\tilde{y}^2}{s_t} + \log(s_t) + \log(2\pi)\right)$$

and expanding $\tilde{y}^2$ slightly gives:

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(\frac{\left(y_t - hf\hat{\mu}_{t-1|t-1}\right)^2}{s_t} + \log(s_t) + \log(2\pi)\right) \tag{A1.42}$$

The recursive form of the leaf marginal in Equation (A1.30) can be rewritten as

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(\frac{\mathrm{I_{t,b}}\, y_{1,b}^2}{v_{t,b}} - \frac{d_{t,b}^2}{a_{t,b}} + \log\left(\frac{(2\pi)^2 w_{t,b}\, \mathrm{I_{t,b}}\, v_{t,b}}{2\pi\, a_{t,b}}\right)\right)$$

which takes the forms

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(\frac{y_{1,b}^2 a_{t,b} - d_{t,b}^2 v_{t,b}}{a_{t,b} v_{t,b}} + \log\left(\frac{(2\pi)^2 w_{t,b}\, \mathrm{I_{t,b}}\, v_{t,b}}{2\pi\, a_{t,b}}\right)\right) \text{ if } \mathrm{I_{t,b}} = 1 \tag{A1.43}$$

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(-\frac{d_{t,b}^2}{a_{t,b}} + \log\left(\frac{(2\pi)^2 w_{t,b}}{2\pi\, a_{t,b}}\right)\right) \text{ if } \mathrm{I_{t,b}} = 0. \tag{A1.44}$$

As $\mathrm{I_{t,b}}$ introduces tree dependent randomness to the tree marginal likelihood it is necessary for comparison with the marginal in Equation (A1.42) to only consider the case where $\mathrm{I_{t,b}} = 1$ for all t. Note that $d_{t,b}$ is also a function of y_t at every t so that it is necessary to expand Equation (A1.43) to

$$\ell m_t = \ell m_{t-1} - \frac{1}{2}\left(\frac{(y_{1,b}\sqrt{a_{t,b}} - d_{t,b}\sqrt{v_{t,b}})(y_{1,b}\sqrt{a_{t,b}} + d_{t,b}\sqrt{v_{t,b}})}{a_{t,b} v_{t,b}} + \dots\right)$$

which leads to

$$\begin{aligned}
y_{1,b}\sqrt{a_{t,b}} - d_{t,b}\sqrt{v_{t,b}} &= y_{t,b}\sqrt{a_{t,b}} - \left(\frac{hyw a_{t-1,b}^* - fv d_{t-1,b}}{\sqrt{v} w a_{t-1,b}^*}\right) \\
&= \frac{a_{t-1,b}^* w y_{t,b}(\sqrt{v a_{t-1,b}^*} - h) + fv d_{t-1,b}}{\sqrt{v} w a_{t-1,b}^*} \\
y_{1,b}\sqrt{a_{t,b}} - d_{t,b}\sqrt{v_{t,b}} &= y_{t,b}\sqrt{a_{t,b}} + \left(\frac{hyw a_{t-1,b}^* - fv d_{t-1,b}}{\sqrt{v} w a_{t-1,b}^*}\right) \\
&= \frac{a_{t-1,b}^* w y_{t,b}(\sqrt{v a_{t-1,b}^*} + h) - fv d_{t-1,b}}{\sqrt{v} w a_{t-1,b}^*}
\end{aligned} \tag{A1.45}$$

Let $a = a_{t-1,b}^* w y_{t,b}$, $b = \sqrt{v a_{t-1,b}^*}$, $c = fv d_{t-1,b}$ and $d = \sqrt{v} w a_{t-1,b}^*$ and then after

some algebra:

$$\begin{aligned}\frac{(a(b-h)+c)(a(b+h)-c)}{d} &= \frac{a^2(b^2-h^2)+2cah-c^2}{d} \\ &= \frac{(b^2-h^2)(a-\frac{ch}{(b^2-h^2)})^2+c^2(1-\frac{h^2}{(b^2-h^2)})}{d} \\ &= \frac{(v\mathfrak{a}^*_{t-1,b}-h^2)(\mathfrak{a}^*_{t-1,b}wy_{t,b}-hf\frac{v\mathfrak{d}_{t-1,b}}{(v\mathfrak{a}^*_{t-1,b}-h^2)})^2+fv\mathfrak{d}^2_{t-1,b}(1-\frac{h^2}{(v\mathfrak{a}^*_{t-1,b}-h^2)})}{\sqrt{v}w\mathfrak{a}^*_{t-1,b}}\end{aligned} \tag{A1.46}$$

A1.2 Simulation Study 1.1

A1.2.1 Aim

This simulation study compares the effect of known (as opposed to randomly chosen) wrong covariates on the filter estimates and tree marginal against both the true model and the Kalman filter. The study averages over 100 data sets simulated from a 2 leaf tree model with known leaf parameters. The results will show that, on average, the 2 leaf filter accuracy of estimating the latent state will depend on the specification of the parameters at both the leaves that are and not updated. The tree model is not shown to be more accurate than the Kalman filter in every case but the error bounds around the tree filter estimates are narrower thus confirming that tree filter concentrates the likelihood of the latent state by more accurately describing the support of the sequence of data than just using a Kalman filter.

A1.2.2 Method

A1.2.2.1 Data generation

Data for the experiment are generated from 100 repeated simulations from a 2-leaf tree model, T_{sim}, where there is a single covariate, x_1 with bounds $(0, 1)$ and threshold value 0.5, at the root node. The leaf model parameters for $b = 2, 3$ are shown in Table A1.2.1:

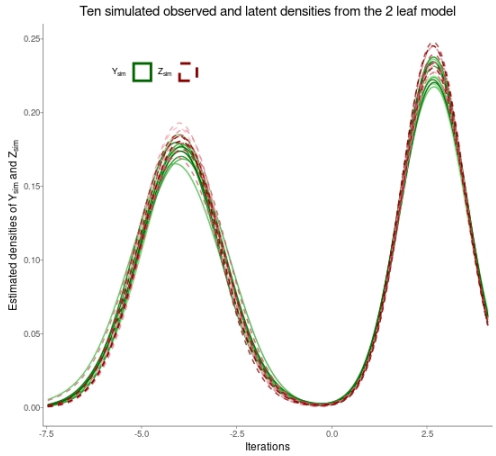
Leaf	Parameters for 2 leaf tree simulation					
	V_b	W_b	F_b	H_b	G_b	u_b
2	0.1	0.25	0.75	1	1	-1
3	0.05	0.1	-0.5	1	1	4

Table A1.2.1: Parameters for the tree simulation. Note the additional parameters G_b and u_b used to generate a constant mean at a fixed value for each of the leaves.

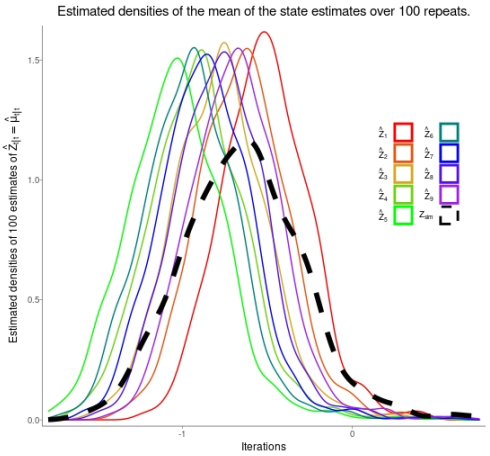
The model that is used for data generation, shown in, Equation (A1.47), is different from that used for modelling because it is necessary to generate a constant value from each of the models at the leaves.

$$\begin{aligned} z_{t,b} &= F_{t,b} z_{t-1,b} + G_{t,b} u_{t,b} + \mathsf{N}(0, W_{t,b}) \\ y_{t,b} &= H_{t,b} z_{t,b} + \mathsf{N}(0, V_{t,b}) \end{aligned} \tag{A1.47}$$

Figure A1.2.1a shows ten examples of the observed data Y_{sim} and the latent data, Z_{sim} drawn from the 2 leaf model. The simulation is performed by drawing



(a) Ten example densities of the observed and latent distributions simulated from the 2-leaf tree model.



(b) The densities of the latent state mean after being averaged over 100 sample data sets.

Figure A1.2.1

a threshold value $c_{t,1}$ uniformly from $(0, 1)$ and using this to choose one of the two leaf models at nodes $b = 2, 3 \in K_{T_{sim}}$. A latent value, $z_{t,b}$, is generated from the chosen leaf and from this latent value an observed value, y_t, is generated. The data point $d_t = (x_{t1}, y_t)$ is then written to a file to be used for simulation modelling. In total $N = 1000$ data points are generated for each of 100 trees.

A1.2.2.2 Simulation modelling

Nine weak learners are initialised with covariate values shown in Table A1.2.2a. Each tree model is set with the same leaf parameters, shown in Table A1.2.2b, which also match those of the simulation model.

Covariate Values for Trees 1 to 9								
1	2	3	4	5	6	7	8	9
0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9

(a) Covariate values for Simulation 1.1. Tree 5 is the 'true' model.

Leaf	Model params for 2 leaf tree					
	V_b	W_b	F_b	H_b	$Z_{0,b}$	$W_{0,b}$
2	0.1	0.25	0.75	1	1	1
3	0.05	0.1	-0.5	1	4	1

(b) Leaf parameters for the model trees.

Table A1.2.2

The simulation modelling is performed for each of the generated data sets by reading in a data point $d_i, i \in 1, \ldots, N = 1000$, passing the data point to each

of the tree models $T = 1, \ldots, 9$, and choosing a leaf from each of the trees. At each T a prediction and update are made via the Kalman filter model of Equations (4.11) and (4.12). That is, a prediction is made at both leaves and an update of the model, using y_i, is made only at the the leaf that is chosen by x_i. Once the one-step ahead prediction and estimated level update are completed, the leaf marginal is recursively calculated using Equation (4.33).The log of the tree marginal is the sum of these leaf marginals. For the purposes of analysis, the values the values a_t and d_t from Equations (4.27) to (4.30) are also produced.

An alternative method with a different but similar objective as this study would be to uniformly sample the covariate values for a single tree and then to repeat this sampling process for each of the sample data sets. In that case the objective would be to find the covariate values that provide the best estimates given the leaf model parameters and fixed model structure. The method chosen for this study however is to explore the effect of known, but wrong threshold parameters on both the average of the filter estimates and the average of their bounds. The aim is also to explore the effect of using a tree model which marginalises over two filters, one that is updated and one that is not.

Also included in this study is a comparison between the average performance of the tree model against the average performance of the Kalman filter. The parameters used for the Kalman filter are shown in Table A1.2.3. The Kalman filter was run over each of the data sets once and then the average of these filter estimates are what is used in the analysis that follows.

Kalman Filter Parameters					
V	W	F	H	Z_0	W_0
1	1.5	0.5	1	0	1

Table A1.2.3: The parameters for the Kalman filter estimation.

The metric used to compare the average of the state level estimates is the cumulative mean square error, CMSE, calculated using Algorithm A1.1.

There are two cases to consider:

1. CMSE$(z_{t|t,b,b} - \hat{\mu}_{t|t,b,b})$ which is the cumulative mean square error between the generated value $z_{t|t,b,b}$ and the estimated level $\hat{\mu}_{t|t,b,b}$ which is the level chosen from the tree at each t by covariate x_t.

2. CMSE$(z_{t|t,b,b} - \hat{\mu}_{t|t,kal,b})$ which is the cumulative mean square error between the generated value $z_{t|t,b,b}$ and the Kalman filter.

Algorithm A1.1: Calculate CMSE, the cumulative mean square error.

Result: CMSE: A vector of length t of the cumulative means of the pointwise errors between two values, x_k and y_k.

Initialise x_0 and y_0
Initialise output vector, out, to length $t + 1$
Calculate $\mathsf{sqdiff}_0 = (x_0 - y_0)^2$ and assign to $\mathsf{out}[0]$
for $i \in 1 : t$ **do**
    Calculate: $\mathsf{sqdiff}_i = (x_i - y_i)^2$
    Calculate: $\mathsf{CMSE}_i = (i-1)/i \times \mathsf{out}_{i-1} + (\mathsf{sqdiff}_i)/i$
    Assign CMSE_i to $\mathsf{out}[i]$
end
return out

A1.2.3 Results

The results of the simulation study are presented mostly in graphical form in the main body of the document. The reason for this, as described in Section 4.5.1 is that the streaming setting does not have a defined end point so that it is necessary to choose a point or points at which one can make reasonable assumptions about the performance of the model if more data were to be received. The end point for this simulation is $N = 1000$. Thus the results that follow are averages that hold only up to 1000 iterations of the algorithm. The key idea with the streaming setting is that N is arbitrary.

A1.3 Simulation Study 3.1

A1.3.1 Aim

The aims of this simulation study are to explore the effect of different tree parameters $\xi_T = (\alpha, \beta)$ on a collection of different leaf model parameters $\psi_T = (V_b, H_b, F_b, W_b, \mu_{0,b}, \Sigma_{0,b})$. There are 8 pairs of ξ_T and 3 groups of ψ_T. The is not a fully comprehensive factorial design so undoubtedly there are interaction effects that may well be missed. This simulation study should be considered exploratory. The results are mostly presented graphically interspersed with commentary where necessary.

A1.3.2 Method

A1.3.2.1 Data generation

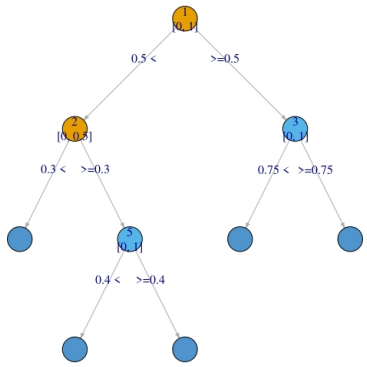
Data are generated from a 5 leaf tree, T_{cgm}, called the CGM model. A single run of 1000 iterations are used. The random effects being marginalised over by 100 experiments are the random shapes of the trees and the filter model output that results from choosing the filter predictions and updates based on the random tree structure.

The data generator is shown in Figure A1.3.1a. This graph shows the internal node threshold values. There are two covariates, x_1 and x_2. The leaf model parameters for $b = 4, 10, 11, 6, 7$ are shown in Figure A1.3.1b: The parameters for the leaves were also provided to 5 independent Kalman filters. These filters are used to show that conditional filtering on an adaptive tree is better than using multiple independent filters.

The model that is used for data generation is shown in Equation (A1.48). The additional input parameters, $G_{t,b}$, and $u_{t,b}$, are necessary to maintain a constant value from each of the models at the leaves.

$$\begin{aligned} z_{t,b} &= F_{t,b} z_{t-1,b} + G_{t,b} u_{t,b} + \mathsf{N}(0, W_{t,b}) \\ y_{t,b} &= H_{t,b} z_{t,b} + \mathsf{N}(0, V_{t,b}) \end{aligned} \tag{A1.48}$$

The simulation is performed from $t \in 1, \ldots N = 1000$ by sampling $x_t = (x_{t,1}, x_{t,2})$ uniformly from $(0, 1)$ for each of the covariate values. The data point is passed through the tree until a terminal node is reached and then a latent value is drawn and used to simulate an observation. These data points are written to a file and subsequently used for modelling.



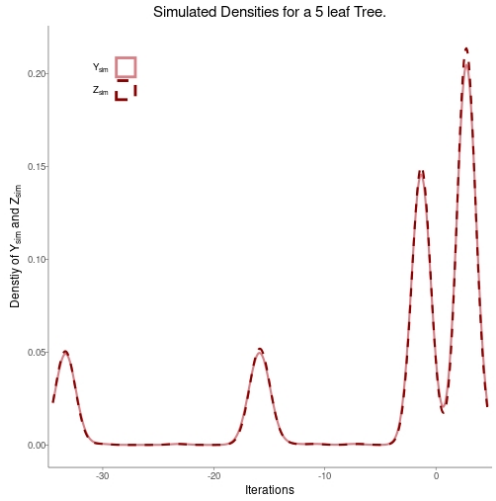
(a) The adapted tree model from (Chipman et al. 1998)

Simulation leaf parameters.

Leaf	V_b	W_b	F_b	H_b	$Z_{0,b}$	$W_{0,b}$	u_b	G_b
4	0.02	0.1	0.25	1	0	1	1	-1
10	0.03	0.15	-0.5	1	5	1	1	5
11	0.05	0.15	0.5	1	8	1	-1	8
6	0.05	0.01	-0.5	1	4	1	1	4
7	0.05	0.15	0.25	1	2	1	-1	25

(b) The leaf parameters for the 5 leaf tree. The additional parameters G_b and u_b used to generate a constant mean at a fixed value for each of the leaves.

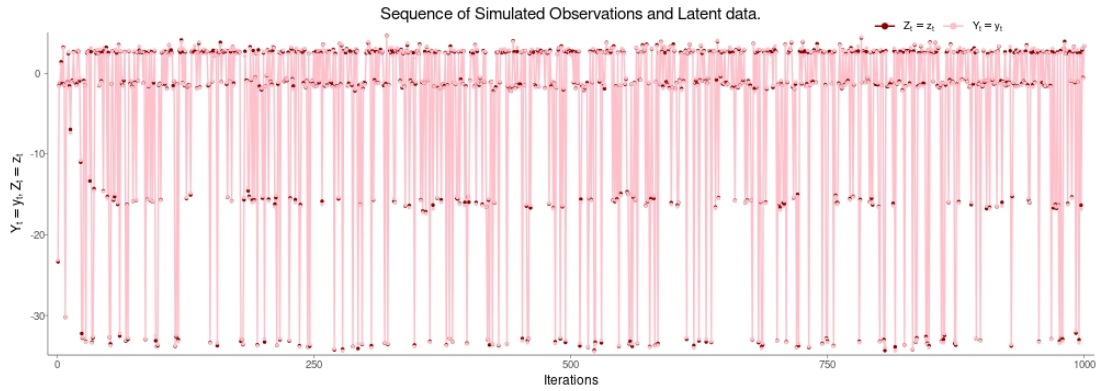
Figure A1.3.1



(a)

Figure A1.3.2: The density generated by the CGM tree model.

Figures A1.3.2 and A1.3.3a show the estimated density of the observed data Y_{sim} and the latent data, Z_{sim} and the generated sequence of data drawn from the 5 leaf model that is used for the simulation modelling that follows.



(a)

Figure A1.3.3: Sequence of data simulated from the CGM model.

A1.3.2.2 Simulation modelling

The main objective of this study is to examine the effect of the tree parameters $\xi_T = (\alpha, \beta)$ on both the output of the model and the random tree generation. However, there are interaction effects between the ξ_T and the leaf model parameters $\psi_T = (V_b, H_b, F_b, W_b, \mu_{0,b}, \Sigma_{0,b})$. This is not only clear from Section 4.5.1 where it was shown that the leaf models have a sizeable, possible detrimental effect on the performance on the model but also because in the random tree model these parameters are being randomly selected from a deterministic set of possible parameters. Further, these parameters are being modified as the tree changes shape. This study will focus on the change in tree output accuracy and the change in the posterior target density that results from different fixed treatments by ξ_T and a fixed, limited selection of ψ_T.

Table A1.3.1 show the different sets of ξ_T that will be used in the experiments. The labels for the experiments the numbers 1 to 8 and will be referred to by the tag EXP or as experiments. The entries in the table are the values of ξ_T that have used.

Tree parameter values for experiments 1 to 8								
ξ_T	1	2	3	4	5	6	7	8
α	0.5	0.95	0.95	0.95	0.75	0.95	0.75	0.95
β	0.5	0.5	1	1.5	2	3	3	2

Table A1.3.1: Tree parameters ξ_T for each of 8 experiments.

Figure A1.3.1b shows the sets of ψ_T that are used for the treatments of the leaf model parameters. The trees are divided up further into groups of experiments:

- Tree 1 is the base tree. Other experiments, with group names W_0, H and W for the parameters that change are deviated from these base parameters

by either increasing or decreasing them substantially.

- Trees 2 and 3 are group W_0. Relative to Tree 1, W_0 has been either decreased substantially in treatment 2 and increased substantially in treatment 3.
- Trees 4 to 7 form the H group. The H_b parameter increases slightly from treatment 4 to treatment 7. Preliminary investigations show that the algorithm is very sensitive to this parameter so small increments are necessary.
- Trees 8 and 9 make up the W group. Like the W_0 group these parameters increase and decrease substantially relative to the control in Tree 1.

All parameters must be seen as not only relative to the control but relative to the parameter V_b. This has been fixed at 2 for all experiments. Preliminary studies and experiments have shown the maintaining a relatively high signal to noise ratio is essential for modelling performance. Theory and the calibration study of Section 4.4 have shown that it is necessary to keep F_b in $(-1, 1)$ and ideally substantially within this interval so 0.7 was fixed on. Section 4.5.1 showed that a negative sign concentrates the scale parameter of the marginal so this has been alternated with a positive sign to increase variation in possible models.

Model Params for all Random Trees							
Tree	Leaf	V_b	W_b	F_b	H_b	$Z_{0,b}$	$W_{0,b}$
1	2	2	12	0.7	1	10	20
	3	2	16	-0.7	1	20	40
2	2	2	12	0.7	1	10	4
	3	2	16	-0.7	1	20	40
3	2	2	12	0.7	1	10	4
	3	2	16	-0.7	1	20	80
4	2	2	12	0.7	0.95	10	20
	3	2	16	-0.7	0.95	20	40
5	2	2	12	0.7	0.99	10	20
	3	2	16	-0.7	0.99	20	40
6	2	2	12	0.7	1.01	10	20
	3	2	16	-0.7	1.01	20	40
7	2	2	12	0.7	1.05	10	20
	3	2	16	-0.7	1.05	20	40
8	2	2	2	0.7	1	10	20
	3	2	4	-0.7	1	20	40
9	2	2	24	0.7	1	10	20
	3	2	32	-0.7	1	20	40

Table A1.3.2: Leaf model parameters ψ_T for the 9 trees.

The final set of functions used are those used to modify the leaf parameters as the tree model changes shape.

Functions parameters for filter modification.										
Move	Parameter	Function	Args 1	Args 1	Args 2	Args 2	Args 3	Args 3	Args 4	Args 4
grow	Z0	runif	n	1	min	-40.00	max	10.0		
	W0	runif	n	1	min	1.00	max	30.0		
	V	Vfunc	ln	2	scale	0.01	min	0.1		
	W	IVfunc	ln	2	scale	0.50	max	30.0		
	F	Ffunc	ln	2	scale	0.05	sn	1.0	min	0.10
	H	constF								
prune	Z0	runif	n	1	min	-40.00	max	10.0		
	W0	runif	n	1	min	1.00	max	30.0		
	V	IVfunc	ln	2	scale	0.01	max	5.0		
	W	Vfunc	ln	2	scale	0.50	min	5.0		
	F	IFfunc	ln	2	scale	0.05	sn	1.0	max	0.95
	H	constF								

Table A1.3.3: Leaf parameter modification functions. These change the parameters as a function of the changes in tree shape.

Functions 'Ffunc' ($f(\cdot)$) and 'IFfunc' ($f^{-1}(\cdot)$) are,

$$f(F_{\mathrm{P}(b)}, b, s, \mathrm{sgn}, \min_F) = \max\{\mathrm{sgn} \cdot F_{\mathrm{P}(b)} \lfloor \log_2(b) \rfloor^{-s}, \min_F\} \tag{A1.49}$$

$$f^{-1}(F_{\mathrm{C_{l,r}}(b)}, b, s, \mathrm{sgn}, \max_F) = \min\{\mathrm{sgn} \cdot F_{\mathrm{C_{l,r}}(b)} \lfloor \log_2(b) \rfloor^{s}, \max_F\} \tag{A1.50}$$

and functions 'Vfunc' ($g(\cdot)$) and 'IVfunc' ($g^{-1}(\cdot)$) are,

$$g(V_{\mathrm{P}(b)}, b, s, \min_V) = \max\{V_{\mathrm{P}(b)} \lfloor \log_2(b) \rfloor^{-s}, \min_V\} \tag{A1.51}$$

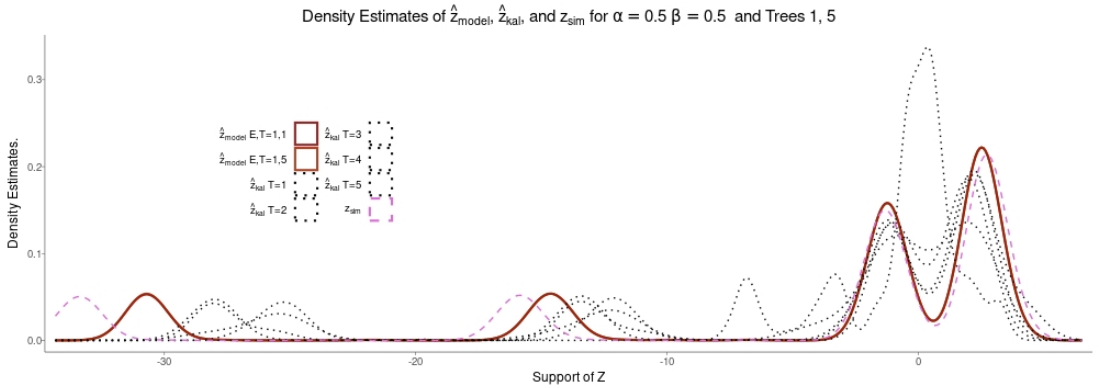
$$g^{-1}(V_{\mathrm{C_{l,r}}(b)}, b, s, \max_F) = \min\{V_{\mathrm{C_{l,r}}(b)} \lfloor \log_2(b) \rfloor^{s}, \max_V\} \tag{A1.52}$$

while, 'runif' is a continuous uniform sample and 'constF' is the constant function.

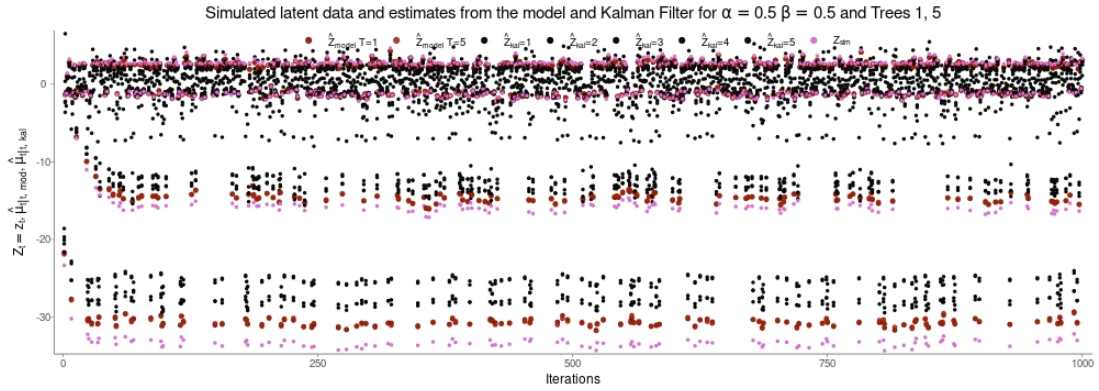
A1.3.3 Results

Figure A1.3.4a shows the fundamental validity of this approach. There are 5 independent Kalman filters each with one of the sets of $\psi_{sim,b=4,10,11,6,7}$. None of the independent filters can accurately converge on the conditional process generated by the tree model (Figure A1.3.2). This result is supported by Figure A1.3.4b which shows that rate of convergence of conditional and independent filters is roughly the same but that the tree (conditional) filters approach the simulated processes more accurately, albeit not fully precisely. Bearing in mind that the conditional filters all start from two leaves it is clear that additional information about the relationship between the processes is learned by the trees although specifying exactly what has been learnt is not clear.

Figure A1.3.5 shows the cumulative mean square error (detailed in Algorithm A1.1) between the Kalman filters and the generated data as well as between the gen-



(a) Density estimates for of the latent variable for two trees, one experiment and 5 independent Kalman filters. The independent KFs have the same filter parameters as those that were used to generate the data.



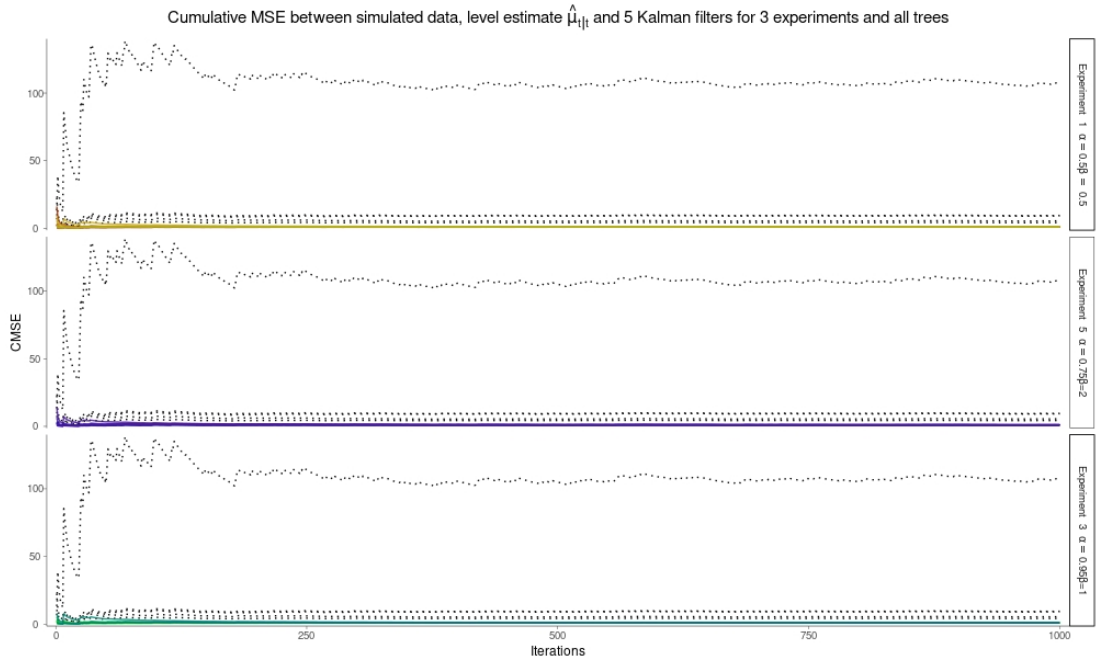
(b) The sequence of estimates from the two trees and the 5 Kalman filters.

Figure A1.3.4

erated data and the simulation models. This graphic shows all of the trees over experiments 1,5,3 each of which have been repeated 100 times. Thus, using the specifications of the previous section it is clear that under these experimental conditions the tree model outperforms the Kalman filter irrespective of the parameters ξ_T.

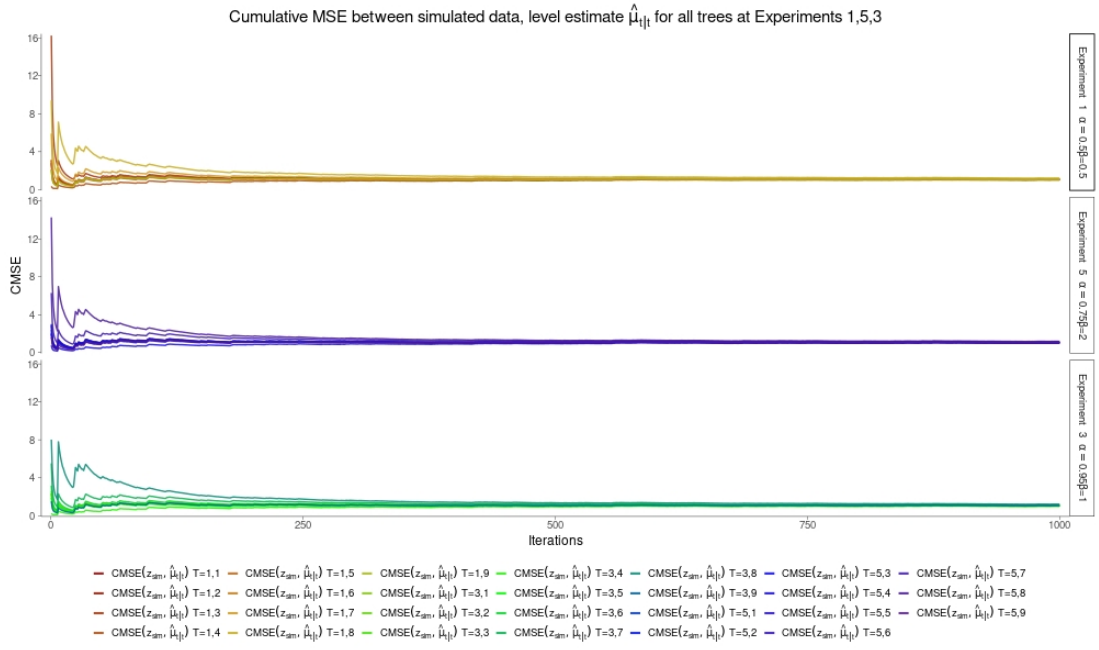
Figure A1.3.6 leaves out the independent Kalman filters and takes a look at the same experiments as Figure A1.3.5 but here the focus is on the effect of the initial different parameters at the leaves of the trees. The range of CMSEs after 1000 iterations is from 0.995 for tree 4 in Experiment 5 to 1.235 for Tree 8 in Experiment 7. Experiment 3 has the lowest initial CMSE but does not converge below 4 as rapidly as Experiments 1 and 5. This is important because the rate at which the accuracy of the model attains its optimal error helps to suggest both the filter parameter relationships and the number of iterations necessary per data point (Section 6.2.2).

Figure A1.3.7 looks at the CMSE from the point of view of a selection of individual trees over all experiments. These graphs show that each of the trees converge for



(a)

Figure A1.3.5: The cumulative means square error (CMSE) between the simulated data, the 5 Kalman filters and all of the trees for 3 experiments.

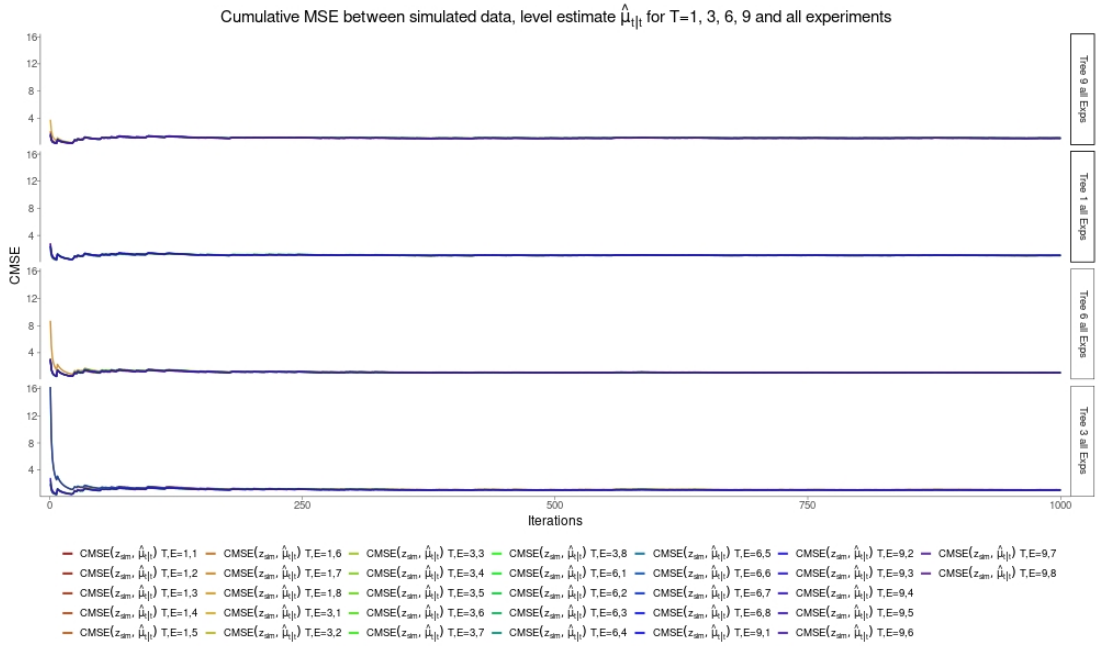


(a)

Figure A1.3.6: A more detailed view of the cumulative means square error (CMSE) between the simulated data showing all of the trees for Experiments 1,5,3.

all settings of ξ_T. Tree 3 which has the largest range of initial parameter, $W_{0,b}$, has the largest difference in initial convergence accuracy with Tree 1, the base tree having the least. Tree 9 which has largest initial signal-to-noise ratio is the second most accurate and second fastest to converge. This result helps to confirm

that preliminary investigations that showed the importance of the signal-to-noise ratio.



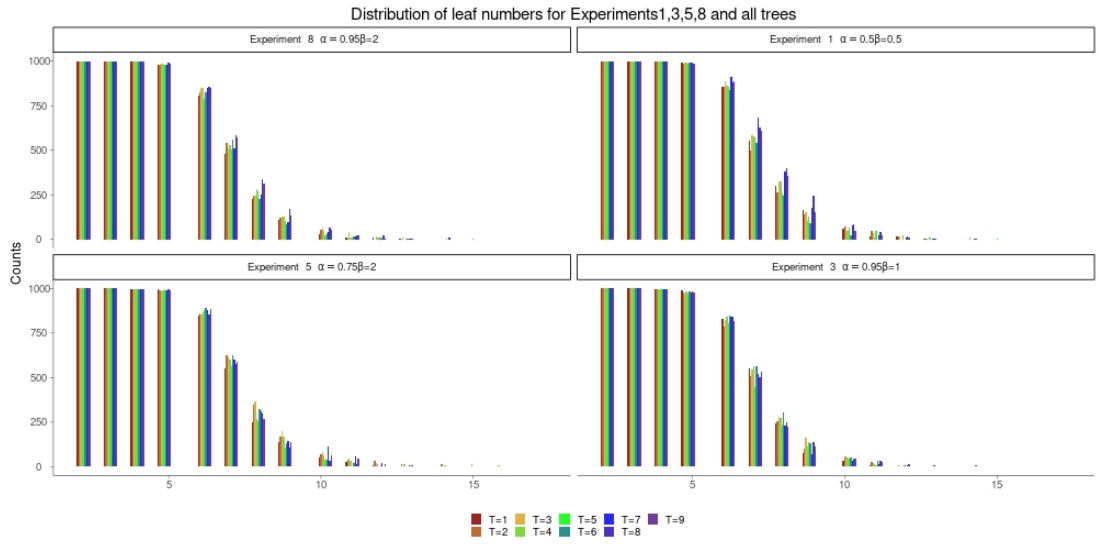
(a)

Figure A1.3.7: The cumulative means square error (CMSE) between the simulated data and Trees 9,1,6,3 for all experiments.

Figure A1.3.8 presents 4 experiments, 8,1,5,3, that show the distribution of the largest leaf numbers (as opposed to tree size). Conditional on the parameter settings, every tree was restricted to a leaf number that was less than 20 ($d_\eta < 5$, $|K_T| + |I_T| < 21$). This is good because it means that, over 1000 iterations repeated 100 times (100000 iterations), the combination of prior settings and parameters were able to control the tree performance (as opposed to preliminary work where the tree size was expanding chaotically). On average, all trees obtained a maximum of 5 leaves over 1000 iterations, the same as the simulated model, but in some of the 100 repeats of experiments trees of up to 18 nodes ($|K_T| + |I_T|$) were found. Experiments with different tree shapes would be necessary to establish if the combination of prior and parameters is able to restrict and learn tree structure.

Another interesting item to notice are Trees 8 and 9. In Experiment 1, these trees which have alternate s-n-r ratios to the base tree (Tree 1) also have the largest difference in tree size when compared to Tree 1 and reach this difference more often. This occurs to a slightly lesser extent in Experiment 5 and 8. The frequency with with larger trees are reached, conditional on parameter settings helps to support the case for the effect of signal-to-noise ratio (s-n-r) on tree

control but is currently not conclusive. Experiment 3 and Tree 7 also show the effect of the parameter H_b on the frequency of tree leaf number conditional on the parameters ξ_T. That is, the tree size seems to get larger more often if H_b exaggerates the effect of the data update on the leaf.



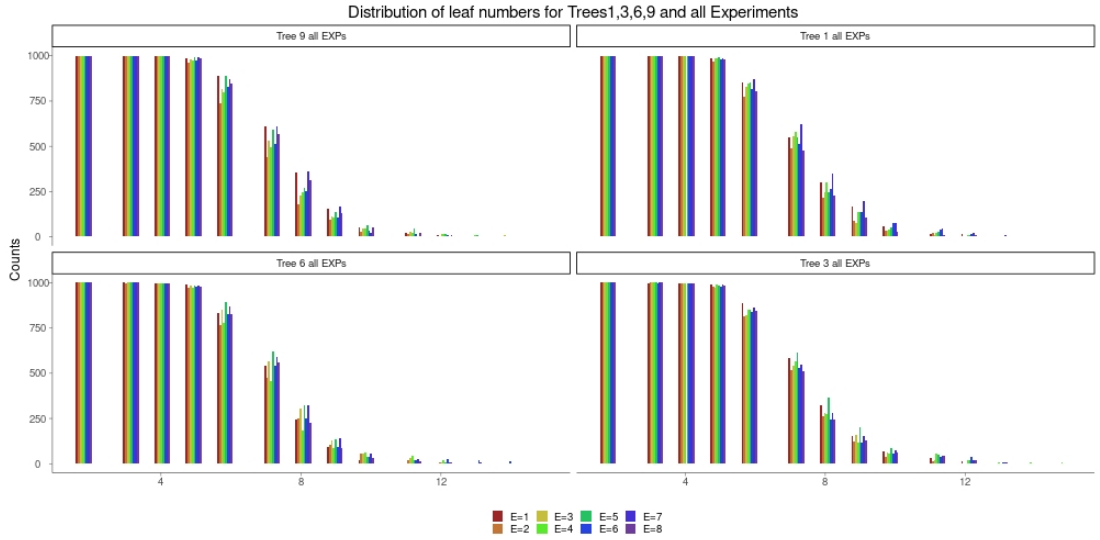
(a)

Figure A1.3.8: The distribution of the largest leaf number attained by the trees for Experiments 1,3,5,8 over all trees 1 to 9.

Figure A1.3.9 considers Trees 1,3,6,9 over all eight experiments (different settings of ξ_T). For the different tree parameter settings shown, Experiment 1 seems to all allow for larger trees more often. For Tree 9, the largest trees are attained more often for Experiments 8 and 9. Trees 6 and 3 obtain there most frequent largest, size during Experiment 4.

These next tables, Tables A1.3.4 and A1.3.5, show the values of the CMSE at set iterations $5, 10, 15, 20, 50, 250$ as well as the maximum leaf number attained over all experiments up to that iteration. The intention is to compare the accuracy of the state estimates under the different experimental conditions and the stages at which these estimates converge. Figures A1.3.6 and A1.3.7 have shown that in all cases the CMSE converges to a small enough value. However, in the streaming setting it is important to minimise the number of iterations necessary to get a small enough CMSE. Thus only iterations up to 250 are shown and of particular interest are the iterations 5,10,11 because this provide insight into the number of iterations necessary for filter convergence if there are multiple tree proposals per update observation.

At iteration 5, Tree 4 has the smallest CMSE for all experiments except for Experiment 4 where Tree 9 has the smallest CMSE. The maximum leaf number



(a)

Figure A1.3.9: The distribution of the largest leaf number attained by the trees for Tree 1,3,6,9 over all experiments 1 to 8.

is 6 for Tree 4 in Experiment 2 but is 5 for all other trees. This a negligible difference in tree size. This ranking of trees persists until iteration 20 when all Tree 4 is then the most accurate tree in all experiments. Considering Table A1.3.2, Tree 4 is when H_b is the smallest: the effect of the update on the tree value is reduced when compared to the base Tree 1. At iteration 10 Tree 4 has largest leaf numbers 5 in Experiments 2,3,5; 6 in Experiment 7 and 7 in all the other experiments. Tree number 9 has largest leaf number 6 in Experiment 4. The largest leaf node does not increase hugely up to iteration 20 but by iteration 50 tree 4 in experiment 3 has grown to be the largest with leaf number 10 with only a marginal gain in accuracy.

A1.3.4 Conclusion

The results from the above table show that for all experiments, that is, for all different tree parameters ξ_T the effect of the parameter H_b appears to be the one that contributes largely to the performance of the model. Overall, after 1000 iterations, tree 4 in experiment 6 is the largest and most accurate tree. The smallest trees are Tree 4 in Experiments 4 and 5 and the most accurate of these is Tree4 in Experiment 5. Bearing in mind that a tree with largest leaf 9 is one depth greater than a tree with largest leaf 6 and the number of parameters involved per filter is 6 then some loss of accuracy per tree might be acceptable especially when averaging over an ensemble of trees.

Accuracy and Max Leaf node for each Experiment and Tree.															
Exp	Tree	CMSE 5	LN 5	CMSE 10	LN 10	CMSE 15	LN 15	CMSE 20	LN 20	CMSE 50	LN 50	CMSE 100	LN 100	CMSE 250	LN 250
	1	0.633	5	0.971	7	0.651	9	0.498	9	0.965	7	1.382	7	1.143	7
	2	0.704	5	0.991	6	0.664	7	0.508	6	0.987	8	1.400	8	1.173	7
	3	3.529	5	2.442	6	1.632	6	1.234	5	1.263	8	1.559	10	1.236	9
	4	0.126	5	0.385	7	0.259	6	0.202	6	0.484	8	0.895	9	0.941	7
1	5	0.500	5	0.847	6	0.568	6	0.435	6	0.855	7	1.258	8	1.121	14
	6	0.717	5	1.144	6	0.767	6	0.585	7	0.990	9	1.372	7	1.169	8
	7	1.262	4	1.865	5	1.255	5	0.955	6	1.644	8	1.852	7	1.370	9
	8	2.183	5	5.699	6	3.890	8	2.950	9	3.300	8	2.628	11	1.694	6
	9	0.469	5	0.696	6	0.467	6	0.360	9	0.934	7	1.324	9	1.127	9
	1	0.453	5	0.975	6	0.654	6	0.500	7	0.986	7	1.338	7	1.144	6
	2	0.616	5	1.057	5	0.709	5	0.542	5	1.014	6	1.329	6	1.125	9
	3	0.585	5	1.003	8	0.673	7	0.515	6	1.095	7	1.485	7	1.222	9
	4	0.140	6	0.397	5	0.267	5	0.208	8	0.496	8	0.864	6	0.922	8
2	5	1.777	5	1.619	6	1.083	6	0.822	7	1.005	7	1.296	6	1.114	6
	6	2.041	5	1.831	5	1.227	6	0.931	7	1.277	6	1.497	10	1.213	6
	7	1.069	5	1.882	5	1.268	7	0.965	7	1.759	7	2.004	7	1.461	7
	8	2.565	5	5.705	6	3.884	5	2.943	5	3.241	7	2.516	6	1.602	7
	9	0.911	5	0.883	6	0.592	6	0.452	7	0.901	9	1.287	7	1.108	6
	1	0.529	6	0.966	7	0.649	7	0.496	8	1.003	7	1.340	8	1.133	8
	2	0.550	5	0.968	5	0.649	6	0.496	6	0.977	11	1.350	8	1.153	7
	3	0.602	5	1.003	6	0.673	6	0.514	7	0.986	9	1.403	8	1.234	7
	4	0.108	5	0.386	5	0.260	6	0.203	5	0.441	10	0.864	9	0.903	7
3	5	0.453	4	0.831	6	0.556	6	0.427	8	0.849	7	1.208	6	1.059	7
	6	0.723	6	1.228	6	0.823	6	0.629	8	1.144	8	1.558	7	1.247	6
	7	1.196	7	1.836	6	1.238	6	0.942	6	1.706	7	2.031	7	1.457	8
	8	2.145	5	6.233	6	4.280	6	3.250	5	3.927	7	2.856	7	1.744	9
	9	0.361	5	0.654	7	0.438	5	0.338	6	0.903	9	1.340	12	1.161	7
	1	0.492	5	0.931	5	0.626	6	0.479	5	0.982	6	1.324	7	1.120	9
	2	0.780	6	1.106	6	0.741	6	0.566	6	1.016	6	1.398	7	1.153	10
	3	0.465	5	0.956	6	0.642	6	0.492	6	1.027	6	1.372	8	1.153	10
	4	1.264	5	0.878	5	0.587	6	0.448	6	0.531	7	0.907	9	0.948	6
4	5	0.401	5	0.869	6	0.582	8	0.446	5	0.877	9	1.278	6	1.151	7
	6	0.635	5	1.158	6	0.779	6	0.595	6	1.221	5	1.491	9	1.209	7
	7	1.196	5	1.905	6	1.283	6	0.976	7	1.651	10	2.028	7	1.455	6
	8	2.727	5	6.306	6	4.291	5	3.249	7	3.519	5	2.715	8	1.680	9
	9	0.363	5	0.708	6	0.474	6	0.365	8	0.929	14	1.345	8	1.130	7

Table A1.3.4: Number of iterations, CMSE and maximum tree leaf number for all trees and Experiments 1 to 4.

Accuracy and Max Leaf node for each Experiment and Tree.															
Exp	Tree	CMSE 5	LN 5	CMSE 10	LN 10	CMSE 15	LN 15	CMSE 20	LN 20	CMSE 50	LN 50	CMSE 100	LN 100	CMSE 250	LN 250
	1	0.555	6	1.047	7	0.702	6	0.536	6	1.037	9	1.416	8	1.207	7
	2	0.681	5	1.048	7	0.703	6	0.537	5	1.039	6	1.437	7	1.208	8
	3	0.509	5	0.944	5	0.633	7	0.485	7	0.945	7	1.356	9	1.164	6
	4	0.204	5	0.406	5	0.273	5	0.213	6	0.447	8	0.845	7	0.911	8
5	5	0.471	5	0.805	7	0.539	6	0.413	7	0.846	7	1.240	8	1.105	8
	6	0.658	6	1.117	6	0.750	6	0.573	5	1.063	10	1.448	8	1.178	7
	7	1.350	5	1.913	6	1.288	7	0.980	8	1.743	7	2.023	11	1.443	7
	8	3.043	6	5.577	6	3.807	6	2.888	6	3.298	8	2.557	7	1.624	10
	9	0.387	5	0.651	6	0.436	7	0.336	6	0.866	6	1.344	11	1.139	7
	1	0.551	5	0.984	6	0.660	8	0.505	7	0.922	10	1.325	8	1.133	9
	2	0.572	6	0.945	6	0.634	6	0.485	8	0.995	7	1.428	6	1.181	7
	3	3.580	4	2.492	6	1.667	7	1.260	6	1.314	7	1.537	8	1.203	6
	4	0.246	5	0.429	7	0.289	6	0.224	5	0.446	7	0.809	6	0.856	7
6	5	0.550	5	0.843	6	0.565	7	0.433	8	0.855	8	1.237	8	1.080	9
	6	0.685	5	1.129	7	0.758	7	0.580	6	1.107	5	1.414	7	1.151	8
	7	1.200	4	1.851	5	1.246	5	0.948	6	1.736	8	2.009	8	1.431	7
	8	2.091	5	5.459	5	3.723	8	2.824	9	3.397	6	2.579	9	1.618	6
	9	0.332	4	0.596	6	0.400	5	0.309	6	0.883	7	1.313	7	1.121	6
	1	0.565	5	0.965	7	0.647	7	0.495	8	1.025	7	1.396	8	1.187	6
	2	0.581	5	1.031	5	0.691	7	0.528	7	0.956	8	1.359	11	1.121	9
	3	0.641	4	0.991	5	0.664	5	0.508	6	0.949	8	1.389	10	1.168	7
	4	0.143	5	0.355	6	0.239	8	0.187	7	0.475	8	0.861	7	0.920	7
7	5	0.510	5	0.870	6	0.582	6	0.446	8	0.835	8	1.225	8	1.095	11
	6	0.693	6	1.151	6	0.772	7	0.590	7	1.092	13	1.481	8	1.205	9
	7	1.093	5	1.864	7	1.256	6	0.956	8	1.721	6	1.982	6	1.436	8
	8	8.300	4	9.458	6	6.404	6	4.838	5	4.249	6	2.994	10	1.823	6
	9	0.404	5	0.693	6	0.465	6	0.357	8	0.919	9	1.372	9	1.147	9
	1	0.636	5	1.004	6	0.674	7	0.516	6	1.028	7	1.371	8	1.131	6
	2	0.681	5	1.168	7	0.783	7	0.597	7	1.048	6	1.373	8	1.140	8
	3	0.478	5	0.882	7	0.592	7	0.454	7	1.008	6	1.326	8	1.155	7
	4	0.161	5	0.356	7	0.239	7	0.187	7	0.481	6	0.911	8	0.961	9
8	5	0.502	5	0.866	6	0.580	5	0.444	7	0.871	7	1.216	8	1.075	6
	6	0.691	5	1.132	8	0.760	6	0.580	6	1.136	6	1.466	8	1.161	8
	7	1.211	6	1.887	6	1.271	6	0.968	6	1.657	6	1.943	6	1.423	8
	8	2.047	5	6.616	8	4.539	6	3.444	9	3.705	8	2.795	13	1.728	8
	9	0.358	5	0.589	7	0.395	8	0.305	7	0.870	6	1.283	7	1.105	6

Table A1.3.5: Number of iterations, CMSE and maximum tree leaf number for all trees and Experiments 5 to 8.

Bibliography

Amazon Kinesis - Data Firehose (2021). Data Firehose - Streaming Data Pipeline - Amazon Web Services. URL: https://aws.amazon.com/kinesis/data-firehose/ (visited on 2021).

Amazon Kinesis - Developer Guide (2017). Streaming Data Developer Guide - Amazon Web Services. URL: https://docs.aws.amazon.com/streams/latest/dev/.

Amazon Kinesis - Streaming (2021). Process & Analyze Streaming Data - Amazon Web Services. URL: https://aws.amazon.com/kinesis/ (visited on 2021).

Amazon MSK (2021). Managed Streaming For Apache Kafka - Amazon Web Services. URL: https://aws.amazon.com/msk/ (visited on 2021).

Anagnostopoulos, Christoforos and Robert B. Gramacy (2013). "Information-Theoretic Data Discarding for Dynamic Trees on Data Streams". In: *Entropy* 15.12, pp. 5510–5535.

Andrieu, Christophe et al. (2010). "Particle Markov chain Monte Carlo methods: Particle Markov Chain Monte Carlo Methods". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72.3, pp. 269–342.

Apache Kafka (2021). Apache Kafka. URL: https://kafka.apache.org/documentation/ (visited on 2021).

Apache Kafka - Documentation (2021). Apache Kafka Documentation. URL: https://kafka.apache.org/documentation/#adminapi (visited on 2021).

Apache Kafka - Monitoring (2021). Apache Kafka Monitoring. URL: https://kafka.apache.org/documentation/#monitoring (visited on 2021).

Apache Kafka - Streaming (2021). Apache Kafka Streams. URL: https://kafka.apache.org/28/documentation/streams/core-concepts (visited on 2021).

Apache Spark 3.1.2 (2021). Overview - Spark 3.1.2 Documentation. URL: https://spark.apache.org/docs/latest/index.html (visited on 2021).

Apache Storm (2021). URL: https://storm.apache.org/about/integrates.html (visited on 2021).

Battin, Richard H. (1964). *Astronomical Guidance*. New York: McGraw-Hill.

Baum, Leonard E and Ted Petrie (1966). "Statistical inference for probabilistic functions of finite state Markov chains". In: *The annals of mathematical statistics* 37.6, pp. 1554–1563.

Bernardo, J.M. and A.F.M. Smith (2007). *Bayesian Theory*. Wiley.

Bifet, Albert, Eibe Frank, et al. (2012). "Ensembles of Restricted Hoeffding Trees". In: 3.2.

Bifet, Albert, Ricard Gavald, et al. (2018). *Machine Learning for Data Streams: With Practical Examples in MOA*. The MIT Press.

Bifet, Albert and Ricard Gavaldà (2009). "Adaptive Learning from Evolving Data Streams". In: *Advances in Intelligent Data Analysis VIII*. Ed. by Niall M. Adams et al. Springer Berlin Heidelberg, pp. 249–260.

Bifet, Albert, Geoff Holmes, et al. (2010). "MOA: Massive Online Analysis". In: *Journal of Machine Learning Research* 11, pp. 1601–1604.

Birnbaum, Allan (1962). "On the Foundations of Statistical Inference". In: *Journal of the American Statistical Association* 57.298, pp. 269–306.

Bleich, Justin et al. (2014). "Variable selection for BART: An application to gene regulation". In: *The Annals of Applied Statistics* 8.3, pp. 1750–1781.

Box, G.E.P. and A. Luceño (1997). *Statistical control by monitoring and feedback adjustment*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley.

Box, George E. P. (1976). "Science and Statistics". In: *Journal of the American Statistical Association* 71.356, pp. 791–799.

Box, George E. P. et al. (2008). *Time series analysis: forecasting and control*. 4th. Wiley series in probability and statistics. John Wiley.

Breiman, Leo (1969). *Probability and stochastic processes: with a view toward applications*. Houghton Mifflin.

– (1996). "Bagging Predictors". In: *Machine Learning* 24.2, pp. 123–140.

– (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Breiman, Leo et al. (1984). *Classification and regression trees*. CRC press.

Broderick, Tamara et al. (2013). "Streaming Variational Bayes". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'13. Curran Associates Inc., pp. 1727–1735.

Campbell, Trevor et al. (2015). "Streaming, Distributed Variational Inference for Bayesian Nonparametrics". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc.

Carvalho, Carlos M. et al. (2010). "Particle learning for general mixtures". In: *Bayesian Analysis* 5.4, pp. 709–740.

Chipman, Hugh A. et al. (1998). "Bayesian CART model search". In: *Journal of the American Statistical Association* 93.443, pp. 935–948.

Chipman, Hugh A. et al. (2010). "BART: Bayesian additive regression trees". In: *The Annals of Applied Statistics* 4.1, pp. 266–298.

Confluent (2021). Confluent: Data in motion. URL: `https://www.confluent.io/` (visited on 2021).

De Finetti, B. et al. (1974). *Theory of Probability: A Critical Introductory Treatment*. Vol. 1. Probability and Statistics Series. Wiley.

Denison, David G. T., Christopher C. Holmes, et al. (2002). *Bayesian Methods for Nonlinear Classification and Regression*. Wiley Series in Probability and Statistics. Wiley.

Denison, David G. T., Bani K. Mallick, et al. (1998a). "A Bayesian CART Algorithm". In: *Biometrika* 85.2, pp. 363–377.

– (1998b). "Bayesian MARS". In: *Statistics and Computing* 8, pp. 337–346.

Domingos, Pedro and Geoff Hulten (2000). "Mining high-speed data streams". In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*. The sixth ACM SIGKDD International Conference. ACM Press, pp. 71–80.

Doucet, Arnaud et al., eds. (2001). *Sequential Monte Carlo Methods in Practice*. Information Science and Statistics. Springer-Verlag.

Draper, Norman R and Harry Smith (1981). *Applied regression analysis*. Vol. 326. John Wiley & Sons.

Evensen, Geir and Peter Jan van Leeuwen (2000). "An Ensemble Kalman Smoother for Nonlinear Dynamics". In: *Monthly Weather Review* 128.6, pp. 1852–1867.

Freund, Yoav and Robert E Schapire (1997). "A decision-theoretic generalization of on-line learning and an application to boosting". In: *Journal of Computer and System Sciences* 55.1, pp. 119–139.

Friedman, Jerome H. (1991). "Multivariate Adaptive Regression Splines". In: *Annals of Statistics* 19.1, pp. 1–67.

– (2001). "Greedy function approximation: A gradient boosting machine." In: *The Annals of Statistics* 29.5, pp. 1189–1232.

Gama, João (2010). *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC.

Gelman, A. et al. (2013). *Bayesian Data Analysis*. 3rd. Taylor & Francis.

Gelman, Andrew and Cosma Rohilla Shalizi (2013). "Philosophy and the practice of Bayesian statistics". In: *British Journal of Mathematical and Statistical Psychology* 66, pp. 8–38.

Geyer, Charles J. and Elizabeth A. Thompson (1995). "Annealing Markov Chain Monte Carlo with Applications to Ancestral Inference". In: *Journal of the American Statistical Association* 90.431, pp. 909–920.

Gilks, W.R. et al. (1995). *Markov Chain Monte Carlo in Practice*. CRC Press.

Gilks, Walter R. and Carlo Berzuini (2001). "Following a moving target-Monte Carlo inference for dynamic Bayesian models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.1, pp. 127–146.

Goldstein, Michael and David Wooff (2007). *Bayes linear statistics: Theory and methods*. John Wiley & Sons.

Gomes, Heitor Murilo et al. (2018). "Adaptive random forests for data stream regression." In: *ESANN*.

Gordon, Neil J et al. (1993). "Novel approach to nonlinear/non-Gaussian Bayesian state estimation". In: *IEE Proceedings F (Radar and Signal Processing)*. Vol. 140. 2. IET, pp. 107–113.

Gramacy, Robert B. et al. (2023). *dynaTree: Dynamic Trees for Learning and Design*. R package version 1.2-15. URL: `https://CRAN.R-project.org/package=dynaTree`.

Green, Peter J. (1995). "Reversible jump Markov chain Monte Carlo computation and Bayesian model determination". In: *Biometrika* 82.4, pp. 711–732.

Grimmett, Geoffrey et al. (2001). *Probability and random processes*. 3rd. Oxford university press.

Hacking, I. and J.W. Romeijn (2016). *Logic of Statistical Inference*. Cambridge Philosophy Classics. Cambridge University Press.

Harrison, Jeff and Mike West (1999). *Bayesian forecasting & dynamic models*. Vol. 1030. Springer New York City.

Hastie, Trevor and Robert Tibshirani (2000). "Bayesian Backfitting". In: *Statistical Science* 15.3, pp. 196–213.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2001). *The Elements of Statistical Learning*. 2nd. Springer Series in Statistics. Springer New York Inc.

Haykin, Simon (2004). *Kalman filtering and neural networks*. Vol. 47. John Wiley & Sons.

Hernández, Belinda et al. (2018). "Bayesian additive regression trees using Bayesian model averaging". In: *Statistics and computing* 28.4, pp. 869–890.

Hulten, Geoff and Pedro Domingos (2003). "VFML – A toolkit for mining high-speed time-changing data streams". URL: `http://www.cs.washington.edu/dm/vfml/`.

Humpherys, Jeffrey et al. (2012). "A Fresh Look at the Kalman Filter". In: *SIAM Review* 54.4, pp. 801–823.

Ikonomovska, Elena, João Gama, and Sašo Džeroski (2011). "Learning model trees from evolving data streams". In: *Data Mining and Knowledge Discovery* 23.1, pp. 128–168.

Ikonomovska, Elena, João Gama, Bernard Ženko, et al. (2011). "Speeding-Up Hoeffding-Based Regression Trees With Options." In: *Proceedings of the 28th International Conference on Machine Learning*. International Machine Learning Society, pp. 537–544.

Jolicoeur-Martineau, Alexia (2016). "Étude dune loi a Priori pour les Arbres Binaires de Régression". https://ajolicoeur.wordpress.com/bayesiantrees/. Masters Thesis. Université du Québec á Montréal.

Jordan, Michael I (1997). *An Introduction to Graphical Models (tutorial)*. URL: `http://www.ai.mit.edu/projects/jordan.html`.

Kalman, R. E. (1960). "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of basic Engineering* 82.1, pp. 35–45.

Kalman, R. E. and R. S. Bucy (1961). "New Results in Linear Filtering and Prediction Theory". In: *Journal of Basic Engineering* 83.1, pp. 95–108.

Kar, Soummya et al. (2012). "Kalman Filtering With Intermittent Observations: Weak Convergence to a Stationary Distribution". In: *IEEE Transactions on Automatic Control* 57.2, pp. 405–420.

Kopetz, Hermann (2011). *Real-Time Systems*. 2nd. Real-Time Systems Series. Springer US.

Kozierok, Charles (2005). *The TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Reference*. No Starch Press.

Levin, David A and Yuval Peres (2017). *Markov Chains and Mixing Times*. 2nd. American Mathematical Society.

Lindley, D.V. (1991). *Making Decisions*. Wiley.

Linero, Antonio R. (2018). "Bayesian Regression Trees for High-Dimensional Prediction and Variable Selection". In: *Journal of the American Statistical Association* 113.522, pp. 626–636.

Maathuis, Marloes et al. (2018). *Handbook of Graphical Models*. CRC Press, Inc.

MacKay, David J. C. (2002). *Information Theory, Inference & Learning Algorithms*. Cambridge University Press.

Meinhold, Richard J and Nozer D Singpurwalla (1983). "Understanding the Kalman filter". In: *The American Statistician* 37.2, pp. 123–127.

Mo, Yilin and Bruno Sinopoli (2008). "A characterization of the critical value for Kalman filtering with intermittent observations". In: *2008 47th IEEE Conference on Decision and Control*. 2008 47th IEEE Conference on Decision and Control. IEEE, pp. 2692–2697.

Mohammadi, Reza et al. (2020). "Continuous-Time Birth-Death MCMC for Bayesian Regression Tree Models". In: *Journal of Machine Learning Research* 21.201, pp. 1–26.

Montiel, Jacob et al. (2018). "Scikit-Multiflow: A Multi-output Streaming Framework". In: *Journal of Machine Learning Research* 19.72, pp. 1–5.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pratola, Matthew T. (2016). "Efficient Metropolis–Hastings Proposal Mechanisms for Bayesian Regression Tree Models". In: *Bayesian Analysis* 11.3, pp. 885–911.

Quinlan, J. R. (1986). "Induction of Decision Trees". In: *Mach. Learn.* 1.1, pp. 81–106.

– (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann series in machine learning. Elsevier Science.

Rasmussen, Carl Edward and Christopher K. I. Williams (2006). *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press.

Särkkä, Simo and Arno Solin (2019). *Applied Stochastic Differential Equations*. Cambridge University Press.

Savage, L.J. (1972). *The Foundations of Statistics*. 2nd. Dover Books on Mathematics Series. Dover Publications.

Sax, Matthias J. et al. (2018). "Streams and Tables: Two Sides of the Same Coin". In: *Proceedings of the International Workshop on Real-Time Business Intelligence and Analytics*. BIRTE '18: International Workshop on Real-Time Business Intelligence and Analytics. ACM, pp. 1–10.

Schapire, Robert E. (1990). "The strength of weak learnability". In: *Machine Learning* 5, pp. 197–227.

Shumway, Robert H. and David S. Stoffer (2017). *Time series analysis and its applications: with R examples*. 4th. Springer.

Singpurwalla, N.D. (2006). *Reliability and Risk: A Bayesian Perspective*. Wiley Series in Probability and Statistics. Wiley.

Sinopoli, Bruno et al. (2004). "Kalman filtering with intermittent observations". In: *IEEE transactions on Automatic Control* 49.9, pp. 1453–1464.

Smallcombe, Mark (2021). *ETL vs ELT: 5 Critical Differences*. Xplenty. URL: `https://www.xplenty.com/blog/etl-vs-elt/` (visited on 06/18/2021).

Stigler, Stephen M. (1986). *The History of Statistics: The Measurement of Uncertainty Before 1900*. Harvard University Press: Cambridge.

streamDM: Data Mining for Spark Streaming (2021). streamDM. URL: `http://huawei-noah.github.io/streamDM/` (visited on 2021).

Taddy, Matthew A. et al. (2011). "Dynamic trees for learning and design". In: *Journal of the American Statistical Association* 106.493, pp. 109–123.

TensorFlow - Probability (2021). A Tour of TensorFlow Probability. URL: https://www.tensorflow.org/probability/examples/A_Tour_of_TensorFlow_Probability (visited on 2021).

Wan, Eric A and Rudolph Van Der Merwe (2000). "The unscented Kalman filter for nonlinear estimation". In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*. IEEE, pp. 153–158.

Wu, Yuhong et al. (2007). "Bayesian CART: Prior Specification and Posterior Simulation". In: *Journal of Computational and Graphical Statistics* 16.1, pp. 44–66.

Xiangheng Liu and A. Goldsmith (2004). "Kalman filtering with partial observation losses". In: *2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)*. 2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601). Vol. 4. IEEE, pp. 4180–4186.