Tree-based Methods for Learning Probability Distributions

by

Naoki Awaya

Department of Statistical Science
Duke University

Date: __________________________

Approved:

______________________________
Li Ma, Supervisor

______________________________
Mike West

______________________________
Surya T. Tokdar

______________________________
Cliburn Chan

Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Statistical Science in the Graduate School of Duke University 2022
Abstract

Tree-based Methods for Learning Probability Distributions

by

Naoki Awaya

Department of Statistical Science
Duke University

Date: ________________________

Approved:

________________________________
Li Ma, Supervisor

________________________________
Mike West

________________________________
Surya T. Tokdar

________________________________
Cliburn Chan

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Statistical Science in the Graduate School of Duke University 2022
Abstract

Learning probability distributions is a fundamental inferential task in statistics but challenging if a data distribution of our interest is complicated and high-dimensional. Addressing this challenging problem is the main topic of this thesis, and mainly discussed herein are two types of new tree-based methods: a single-tree method and an ensemble method. The new single tree method, the main topic of Chapter 2, is introduced by constructing a generalized Pólya tree process, that is, a new Bayesian non-parametric model equipped with a new flexible tree prior. With this new prior we can find trees that represent the distributional structures well, and the tree space is efficiently explored with a new sequential Monte Carlo algorithm. The new ensemble method discussed in Chapter 3 is proposed under a new addition rule defined for probability distributions. The new rule based on cumulative distribution functions and their generalizations enables us to smoothly introduce a new efficient boosting algorithm, inheriting the important notions such as “residuals” and “zeros”. The thesis is closed by Chapter 4 which provides concluding remarks.
To my family
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>iv</td>
</tr>
<tr>
<td>List of Tables</td>
<td>ix</td>
</tr>
<tr>
<td>List of Figures</td>
<td>x</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>xii</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2 Hidden Markov Pólya Trees for High-dimensional Distributions</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Method</td>
<td>9</td>
</tr>
<tr>
<td>2.2.1 Pólya Trees Defined on Recursive Dyadic Partitions</td>
<td>9</td>
</tr>
<tr>
<td>2.2.2 Incorporating Flexible Partition Points</td>
<td>11</td>
</tr>
<tr>
<td>2.2.3 Hidden Markov Pólya Tree Models</td>
<td>13</td>
</tr>
<tr>
<td>2.3 Bayesian Inference</td>
<td>16</td>
</tr>
<tr>
<td>2.3.1 SMC to Sample from Tree Posterior</td>
<td>18</td>
</tr>
<tr>
<td>2.3.2 Posterior Computation given Sampled Tree Structures</td>
<td>22</td>
</tr>
<tr>
<td>2.4 Theoretical Properties</td>
<td>24</td>
</tr>
<tr>
<td>2.5 Experiments</td>
<td>28</td>
</tr>
<tr>
<td>2.5.1 Density Estimation</td>
<td>28</td>
</tr>
<tr>
<td>2.5.2 Two-group Comparison</td>
<td>32</td>
</tr>
<tr>
<td>2.6 Application to A Mass Cytometry Data Set</td>
<td>35</td>
</tr>
</tbody>
</table>
2.7 Concluding Remarks ......................................................... 38

2.8 Technical Details ............................................................ 38

2.8.1 Details of Posterior Computation .............................. 38

2.8.2 Spike-and-slab Type Prior for $L(A)$ ......................... 42

2.8.3 Density Estimation with the APT Model ................... 47

2.8.4 Computing the Conditional Predictive Distribution given a Tree ................................................................. 47

2.8.5 Two-groups Comparison with the MRS Model .......... 48

2.8.6 Proofs ........................................................................... 49

2.8.7 Consistency for the MRS Model with the Flexible Partitioning ................................................................. 61

2.8.8 Numerical Evaluation of the Density Estimators under Different Complexity of Data Generating Processes ............ 63

2.8.9 Details of the Experiments ........................................... 64

3 Unsupervised Tree Boosting for Learning Probability Distributions ................................................. 70

3.1 Introduction ................................................................. 70

3.2 Method ............................................................................ 74

3.2.1 CDF-based Addition and Residualization for Univariate Distributions ................................................................. 74

3.2.2 Generalization to Multivariate Distributions ............ 76

3.2.3 An Unsupervised Boosting Algorithm based on Forward-stagewise (FS) Fitting ................................................................. 83

3.2.4 Decision-theoretic Considerations ............................ 86

3.2.5 Practical Considerations .............................................. 88

3.2.6 Expressive Power of Additive Tree Ensembles ........ 93

3.2.7 Connection to Normalizing Flows ............................. 95

3.3 Numerical Experiments .................................................. 96
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Key concepts in boosting</td>
<td>76</td>
</tr>
<tr>
<td>3.2</td>
<td>The comparison of the predictive scores for the real data sets</td>
<td>101</td>
</tr>
<tr>
<td>3.3</td>
<td>The average training time for the benchmark datasets</td>
<td>101</td>
</tr>
<tr>
<td>3.4</td>
<td>The values of $c_0$ and $\gamma$ (the parameters of the learning rate) selected for the real data analysis</td>
<td>130</td>
</tr>
</tbody>
</table>
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>A graphical representation of the PT models.</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>An example of realized finite trees in the particle filter.</td>
<td>19</td>
</tr>
<tr>
<td>2.3</td>
<td>The KL divergence of the estimated densities with the original APT and the proposed HMPT.</td>
<td>29</td>
</tr>
<tr>
<td>2.4</td>
<td>The estimated densities and the representative trees obtained with the original APT and the proposed HMPT.</td>
<td>30</td>
</tr>
<tr>
<td>2.5</td>
<td>The wall time of the posterior computation for the higher-dimensional cases.</td>
<td>31</td>
</tr>
<tr>
<td>2.6</td>
<td>A comparison of the predictive performance of the proposed HMPT and the existing methods.</td>
<td>32</td>
</tr>
<tr>
<td>2.7</td>
<td>A comparison of the predictive performance under small sample sizes.</td>
<td>33</td>
</tr>
<tr>
<td>2.8</td>
<td>The ROC curves obtained with the PT models.</td>
<td>34</td>
</tr>
<tr>
<td>2.9</td>
<td>Examples of the node with the highest PMAP under the three scenarios for two-group comparison, under the MRS with flexible partitioning.</td>
<td>34</td>
</tr>
<tr>
<td>2.10</td>
<td>The MAP tree for the mass cytometry data set visualized up to the 9th level.</td>
<td>36</td>
</tr>
<tr>
<td>2.11</td>
<td>The full MAP tree for the mass cytometry data set.</td>
<td>36</td>
</tr>
<tr>
<td>2.12</td>
<td>A sample of the nodes capturing the large difference between the two groups on the MAP tree for the mass cytometry data set.</td>
<td>37</td>
</tr>
<tr>
<td>2.13</td>
<td>The average $L_1$ distance obtained with the proposed HMPT under different complexity of the data generating process</td>
<td>68</td>
</tr>
<tr>
<td>Section Number</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>2.14</td>
<td>The average number of nodes in fitted trees obtained with the proposed HMPT under different complexity of the data generating process</td>
<td>69</td>
</tr>
<tr>
<td>3.1</td>
<td>Visualization of the tree-based decomposition of a univariate CDF into three local moves</td>
<td>78</td>
</tr>
<tr>
<td>3.2</td>
<td>Visualization of the local move functions in the tree-CDF</td>
<td>81</td>
</tr>
<tr>
<td>3.3</td>
<td>The KL divergence from the true distribution obtained under the different values of the tuning parameters</td>
<td>98</td>
</tr>
<tr>
<td>3.4</td>
<td>The two-dimensional density functions estimated with the boosting under the different values of $c_0$ (the parameter of the learning rate)</td>
<td>99</td>
</tr>
<tr>
<td>3.5</td>
<td>The comparison of the predictive scores for the real data sets</td>
<td>100</td>
</tr>
<tr>
<td>3.6</td>
<td>Visualization of the computed variable importance for the digit image data, and a sample of handwriting of 0</td>
<td>102</td>
</tr>
<tr>
<td>3.7</td>
<td>The training set of the POWER data and the simulated values generated from the learned probability measure</td>
<td>131</td>
</tr>
<tr>
<td>3.8</td>
<td>The training set of the GAS data and the simulated values generated from the learned probability measure</td>
<td>131</td>
</tr>
<tr>
<td>3.9</td>
<td>The training set of the AReM data and the simulated values generated from the learned probability measure</td>
<td>132</td>
</tr>
</tbody>
</table>
Acknowledgements

I would like to sincerely thank my advisor Dr. Li Ma, who kindly shared his ideas, feedback, enthusiasm, and also life advice with me. I was extremely fortunate to have an opportunity to frequently discuss and share thoughts with him, without which my completing my Ph.D. study would be definitely impossible. I also appreciate my pre-lim/defense committee members, Dr. Mike West, Dr. Surya Tokdar, and Dr. Cliburn Chan for their time and insightful feedback. I also thank the members of Dr. Ma’s research group for their sharing interesting comments and ideas at the meetings.

I am grateful to Dr. Yasuhiro Omori and Dr. Kaoru Irie for supporting my undergraduate and master research and introducing the Duke statistics department to me. Without their encouragement, I would not have even considered joining the great community and missed all of the priceless experiences I have had during the Ph.D. program. I also would like to thank the Nakajima Foundation for financially supporting me with the generous scholarship.
Introduction

This thesis is mainly built on the two papers that were written during the author’s Ph.D. program. They both concern learning probability distributions with tree-based models; The first paper proposes a new single-tree method by generalizing the Pólya tree process-based models, and the second paper introduces a new tree-ensemble model with a novel boosting algorithm. The main two chapters are summarized below.

• Chapter 2: “Hidden Markov Pólya Trees for High-dimensional Distributions” (This chapter is based on Awaya and Ma (2021).)

The Pólya tree (PT) process is a general-purpose Bayesian nonparametric model that has found wide application in a range of inference problems. It has a simple analytic form and the posterior computation boils down to beta-binomial conjugate updates along a partition tree over the sample space. Recent development in PT models shows that performance of these models can be substantially improved by (i) allowing the partition tree to adapt to the structure of the underlying distributions and (ii) incorporating latent state variables
that characterize local features of the underlying distributions. However, important limitations of the PT remain, including (i) the sensitivity in the posterior inference with respect to the choice of the partition tree, and (ii) the lack of scalability with respect to dimensionality of the sample space. We consider a modeling strategy for PT models that incorporates a flexible prior on the partition tree along with latent states with Markov dependency. We introduce a hybrid algorithm combining sequential Monte Carlo (SMC) and recursive message passing for posterior sampling that can scale up to 100 dimensions. While our description of the algorithm assumes a single computer environment, it has the potential to be implemented on distributed systems to further enhance the scalability. Moreover, we investigate the large sample properties of the tree structures and latent states under the posterior model. We carry out extensive numerical experiments in density estimation and two-group comparison, which show that flexible partitioning can substantially improve the performance of PT models in both inference tasks. We demonstrate an application to a mass cytometry data set with 19 dimensions and over 200,000 observations.

- **Chapter 3: “Unsupervised Tree Boosting for Learning Probability Distributions”**
  (This chapter is based on Awaya and Ma (2022).)

We propose an unsupervised tree boosting algorithm for inferring the underlying sampling distribution of an i.i.d. sample based on fitting additive tree ensembles in a fashion analogous to supervised tree boosting. Integral to the algorithm is a new notion of “addition” on probability distributions that leads to a coherent notion of “residualization”, i.e., subtracting a probability distribution from an observation to remove the distributional structure from the sampling distribution of the latter. We show that these notions arise naturally for univariate distributions through cumulative distribution function (CDF)
transforms and compositions due to several “group-like” properties of univariate CDFs. While the traditional multivariate CDF does not preserve these properties, a new definition of multivariate CDF can restore these properties, thereby allowing the notions of “addition” and “residualization” to be formulated for multivariate settings as well. This then gives rise to the unsupervised boosting algorithm based on forward-stagewise fitting of an additive tree ensemble, which sequentially reduces the Kullback-Leibler divergence from the truth. The algorithm allows analytic evaluation of the fitted density and outputs a generative model that can be readily sampled from. We enhance the algorithm with scale-dependent shrinkage and a two-stage strategy that separately fits the marginals and the copula. The algorithm then performs competitively to state-of-the-art deep-learning approaches in multivariate density estimation on multiple benchmark datasets.
2

Hidden Markov Pólya Trees for High-dimensional Distributions

2.1 Introduction

The Pólya tree (PT) (Freedman, 1963; Ferguson, 1974; Lavine, 1992) is a stochastic process that generates random probability measures and is introduced as a prior for Bayesian nonparametric inference. While the PT generalizes the Dirichlet process (DP) (Ferguson, 1973) as it yields the DP under certain hyperparameters (Ferguson, 1974), the statistical properties and practical applications of the PT are very different. While the DP is most frequently used as a mixing distribution that induces clustering structures, the PT is often adopted for directly modeling probability densities.

The PT is defined generatively on a recursive partition—or a partition tree—over the sample space through coarse-to-fine sequential probability assignment at each tree split. In a classical (univariate) PT, the tree is dyadic and the conditional probability assigned to the two children nodes at each tree split arises from independent beta priors, which leads to analytic simplicity and ease in computing the
posterior. Obtaining the posterior is straightforward from beta-binomial conjugacy and incurs a computational budget that scales only linearly with the sample size, making the PT one of the few nonparametric models applicable to data with massive sample size. Moreover, the posterior computation is embarrassingly parallelizable across the tree nodes.

The PT has been applied in various contexts beyond the original application of density estimation. A far-from-exhaustive list includes survival analysis (Muliere and Walker, 1997; Neath, 2003), imputing missing values (Paddock, 2002), goodness-of-fit tests (Berger and Guglielmi, 2001), two-group comparison (Ma and Wong, 2011; Holmes et al., 2015; Chen and Hanson, 2014; Soriano and Ma, 2017), density regression (Jara and Hanson, 2011), ANOVA (Ma and Soriano, 2018), testing independence (Filippi et al., 2017), and hierarchical modeling (Christensen and Ma, 2020). The PT has also been utilized in semi-parametric analyses such as in (generalized) linear models (Walker et al., 1999; Hanson and Johnson, 2002; Walker and Mallick, 1997).

Early developments of the PT are based on an a priori fixed partition tree on the sample space. The resulting inference can be sensitive to the choice of the partition points defining the tree. In particular, the resulting process, both a priori and a posteriori, can be jumpy at these points. In hypothesis testing and model choice, this sensitivity is also reflected in the sometimes substantial change in the marginal likelihood/Bayes factor when the partition points are slightly varied. To remedy the issue, Paddock et al. (2003) modified the PT model so that observations are generated from the PT model with slightly different partition points. Hanson and Johnson (2002) and Hanson (2006) proposed a mixture of PTs by defining partition points along quantiles of a parametric model endowed with a hyperprior to allow model averaging on the partition points. This strategy does not allow individual partition points to adapt to local features of the distribution but only the whole set of points to the global structure of the distribution, and is most effective when
the underlying density is close to the specified parametric model. \cite{Nieto-Barajas and Müller 2012} took a different approach by modeling the probability assignments within each level of the tree in a correlated manner to smooth out the random measure over the boundaries of partitioning. While these approaches alleviate the sensitivity to partition points in low-dimensional settings, they are not easily applicable (though in principle possible) to problems with even just a handful of dimensions. Moreover, Bayesian inference with these models generally require MCMC, whose effectiveness can (in fact often does) still suffer from the sensitivity with respect to the partition points.

Another related issue regarding the partitioning scheme of the PT is that in multivariate problems, traditionally the partition tree is constructed by dividing all dimensions of the sample space at each split. For example, for a $d$-dimensional sample space, each time a tree node is divided, it is split into $2^d$ children nodes, and probability assignment over these $2^d$ child nodes is modeled by independent Dirichlet priors. \cite{Wong and Ma 2010} noted that such a “symmetric” partition scheme is undesirable as the dimensionality increases, in which case due to the exponential growth of the partition blocks, the vast majority of the blocks are barely, if at all, populated by data. As such they propose to incorporate adaptivity into the partitioning strategy with respect to the structure of the underlying distribution through adopting a Bayesian CART-like prior \cite{Chipman et al. 1998} on the space of dyadic partition trees.

However, in order to maintain the analytic simplicity of the posterior and achieving MCMC-free exact Bayesian inference with a linear computational budget, the Bayesian CART-like prior has to be restricted to only divide at the middle point (or otherwise a pre-determined fixed point) on one of the dimensions at each tree split. Not only does this hamper the model’s ability to adapt to distributional structures, but it makes the model suffer from the same sensitivity with respect to the
partition points. Moreover, even with this restriction, the inference algorithm (based on recursive message passing) is only computationally practical for up to about 10 dimensions on continuous sample spaces.

In a different vein, recent developments have demonstrated that aside from enhancing the partitioning strategy, the PT can also be substantially improved by adopting more flexible priors (as opposed to independent betas) on the probability assignment at each tree split (Jara and Hanson, 2011; Nieto-Barajas and Müller, 2012; Ma, 2017). One strategy for enriching the PT in this regard is by introducing latent state variables at each tree split and adopt priors on the probability assignment given these states. When the latent states are discrete and modeled with Markov dependency, analytical simplicity is preserved and exact Bayesian inference can proceed through recursive message passing that maintains the linear computational budget (Ma, 2017).

Given these developments, we are motivated by the following questions: Is it possible to incorporate into the PT a very flexible partition tree prior, such as the general Bayesian CART (i.e., without the restriction to partition at middle points), that will (i) enhance its adaptivity to distributional structures in multivariate settings; (ii) resolve its sensitivity to the choice of partition points; and (iii) allow a tractable form of the joint posterior and a posterior inference algorithm that is scalable to moderately high-dimensional problems (e.g., up to 100 dimensions)? Moreover, should such a strategy exist, can the resulting model and inference algorithm be made compatible with incorporating (possibly Markov dependent) latent states on the tree nodes?

The goals of making the partition tree prior more flexible while enhancing the computational scalability appear at odds with each other. Large tree spaces are well known to be very hard to compute over. In moderate to high dimensional settings exact inference involving flexible tree structures is beyond reach and even the most advanced MCMC approaches tailor-made for trees encounter substantial difficulty
due to the pervasive multi-modality of distributions in such spaces. Recent advances in sequential Monte Carlo (SMC) for regression tree models (Lakshminarayanan et al., 2013; Lu et al., 2013), however, suggest that efficient inference is possible in moderately high-dimensional settings (up to about 100 dimensions). Moreover, once the partition tree is sampled, the conditional posterior for the rest of the model can be computed analytically through recursive message passing. We will therefore exploit a hybrid strategy that uses a new SMC sampler to efficiently sample from the marginal posterior of the partition tree structure, along with recursive message passing to compute the exact conditional posterior of the latent state variables given the tree.

Beyond the methodological development, we will also investigate the theoretical properties of the posterior on the partition tree and the latent states. Previous theoretical literature on the PT and related models have mostly focused on establishing the posterior consistency and the contraction rate of the random measure induced under these models (Castillo, 2017; Castillo and Randrianarisoa, 2021). In multivariate settings, however, the partition tree itself is highly informative about the underlying distribution. Moreover, in applications involving model choice and hypothesis testing, it is often the latent states, not the random measures, that are of direct interest. As such, we focus on studying the asymptotic behavior of the marginal posterior on the partition tree and latent states, establishing consistency results on their convergence toward the trees and states that most closely characterize the underlying truth.

The rest of the paper is organized as follows. In Section 3.2 we describe a flexible prior on the partition tree structure that relaxes the restriction of “dividing in the middle” on partition points and present a general form of PT models that adopt this prior along with latent states associated with the tree nodes with a Markov dependency structure. In Section 2.3 we present our hybrid computational strategy
that can work effectively up to 100 dimensions consisting of an SMC algorithm for sampling on the marginal posterior of the partition tree and a recursive message passing algorithm for obtaining the exact conditional posterior of the latent states and the random measure given the sampled trees. In Section A we investigate the asymptotic properties of the tree structures and latent states identified under the posterior model. In Section 2.5 we carry out extensive numerical experiments to examine the performance of our method in the context of two important applications of PTs—density estimation and the two-group comparison, followed by an application to a data set from mass cytometry in Section 2.6. In Section 2.7 we conclude with a brief discussion. All technical details are provided in Section 2.8.

2.2 Method

We first review the PT model (Ferguson, 1973; Lavine, 1992) on a dyadic recursive partition in Section 2.2.1. The model, while defined on a general multivariate sample space, differs from a traditional multivariate PT which adopts a multi-way symmetric recursive partitioning. Then we introduce a new class of PT models that incorporates both the flexible partition prior and latent states with Markov dependency.

2.2.1 Pólya Trees Defined on Recursive Dyadic Partitions

Without loss of generality, we consider a continuous sample space represented as a $d$-dimensional rectangle $\Omega = (0, 1]^d$. For unbounded sample spaces such as $\mathbb{R}^d$, one can transform each margin to $[0, 1]$ by applying, say, a cumulative distribution function transform or by standardizing the data based on its observed range of values. We use $\mu$ to denote the Lebesgue measure on $\Omega$. A (dyadic) recursive partitioning $T$ on $\Omega$ is a sequence of partitions of $\Omega$ such that the partition blocks at each level of the partitioning are obtained by dividing each block in the previous level into two children blocks. Formally, we can write $T = \bigcup_{k=0}^{\infty} A^k$, where $A^k$ is a partition of $\Omega$ in the $k$th
level. More specifically, $A^0 = \{\Omega\}$, and $A \in A^k$ ($k = 0, 1, 2, \ldots$) is divided into $A_l$ and $A_r$, which satisfy $A_l, A_r \in A^{k+1}$, $A_l \cup A_r = A$, and $A_l \cap A_r = \emptyset$. (Throughout the paper, a subscript “l” or “r” on a node indicates the left or right child node.) For example, when $d = 1$ and if the tree is recursively divided at the middle point of each node, then nodes in level $k$ are of the form $(l/2^k, (l+1)/2^k]$ for some $l \in \{0, \ldots, 2^k-1\}$. Another common strategy is to define the tree based on the quantiles of a probability measure $F$ so that $A \in A^k$ is of the form $A = (F^{-1}(l/2^k), F^{-1}((l+1)/2^k)])$ for $l \in \{0, \ldots, 2^k-1\}$.

Given a partition tree $T$, we can define a random measure $Q$ by putting a prior on the conditional probability $\theta(A) = Q(A_l|A) = 1 - Q(A_r|A)$ at each $A \in T$. Under the PT prior, the parameters $\theta(A)$ follow independent beta distributions $\text{Beta}(\alpha_l(A), \alpha_r(A))$, where $\alpha_l(A)$ and $\alpha_r(A)$ are hyperparameters. The corresponding posterior, given an i.i.d. sample $x_1, \ldots, x_n$ from $Q$, is again a PT with a simple conjugate update on the conditional probabilities:

$$\theta(A) \mid x_1, \ldots, x_n \sim \text{Beta}(\alpha_l(A) + n(A_l), \alpha_r(A) + n(A_r)),$$

where $n(A)$ represents the number of observations in a set $A \subset \Omega$. Though the tree needs to be infinitely deep to ensure full support of the PT, for practical purposes, one typically sets a sufficiently large maximum depth (or resolution) of $T$ and compute the posteriors of $\theta(A)$’s defined on this finite tree structure \cite{Hanson2002}. We shall refer to a node in the deepest level as a “leaf” or “terminal node”. On a leaf, the conditional distribution can be set to a baseline $F(\cdot|A)$, such as the uniform distribution $\mu(\cdot|A)$. In Section 2.3 when we present inference algorithms, we shall adopt this practical strategy and assume $T$ is finite and use $N(T)$ and $L(T)$ to denote the collection of the non-terminal nodes and the leaf nodes, respectively.
2.2.2 Incorporating Flexible Partition Points

We incorporate a Bayesian-CART like prior on $T$ by randomizing both the dimension in which to divide a node and the location to divide. Our prior relaxes the “always-divide-in-the-middle” restriction imposed in [Wong and Ma (2010)]. This prior on the partition tree $T$ differs from that in the mixture of PTs of [Hanson (2006)], which does not randomize over the dimension to divide, but generates the boundaries of the tree nodes jointly using quantiles of a parametric family.

Our prior can be described iteratively as a generative process that recursively divides the sample space. Specifically, suppose we have a node $A$ in the rectangular form, $A = (a_1, b_1] \times \cdots \times (a_d, b_d]$. We divide $A$ into two rectangular children by cutting along a randomly chosen dimension at a random location. The dimension to divide $D(A) \in \{1, 2, \ldots, d\}$, and the (relative) location to divide $L(A) \in (0, 1)$ are given independent priors of the following forms:

$$ D(A) \sim \text{Mult}(\lambda_1(A), \ldots, \lambda_d(A)) \quad \text{and} \quad L(A) \sim \sum_{l=1}^{N_L-1} \beta_l(A) \delta_{l/N_L} (\cdot), \quad (2.1) $$

where $\delta_x (\cdot)$ represents the unit point mass at $x$, and $N_L - 1$ is the total number of grid points along $(0, 1)$. Both $\{\lambda_i(A)\}_{i=1,\ldots,d}$ and $\{\beta_l(A)\}_{l=1,\ldots,N_L-1}$ sum to 1. In the above, we have adopted a uniform grid over $(0, 1)$ for notational simplicity, but it does not have to be as such. With $D(A) = j$ and $L(A) = l/N_L$, the two children nodes $A_l$ and $A_r$ are

$$ A_l = (a_1, b_1] \times \cdots \times (a_j, a_j + l/N_L \cdot (b_j - a_j)) \times \cdots \times (a_d, b_d], $$

$$ A_r = (a_1, b_1] \times \cdots \times (a_j + l/N_L \cdot (b_j - a_j), b_j] \times \cdots \times (a_d, b_d]. $$

In principle one could adopt a continuous prior on the partition location $L(A)$. A discretized prior is helpful, however, because it will substantially simplify posterior computation. In practice, as long as the grid is dense enough, the discrete prior will be practically just as flexible. Indeed we have verified in extensive numerical
experiments that when \( N_L \) is large enough (more than 30 to 50) over a uniform grid, posterior inference no longer improves in any noticeable way.

For the prior on \( D(A) \), we set \( \lambda_j(A) = 1/d \) for all nodes \( A \) as a default choice. When \( L(A) \) is given a weak prior widely spread over \((0, 1)\), the resulting inference can be sensitive to the “tail” behavior of the distribution in the node, resulting in high posteriors of \( L(A) \) near the extreme values 0 and 1. A detailed discussion on this phenomenon will be provided in Section 2.5.1. This issue can be effectively addressed by making the prior of \( L(A) \) depend on the sample size \( n(A) \) so that it encourages more balanced divisions at large sample sizes. More specifically, we adopt the following prior with an exponentially decaying tail

\[
P(L(A) = l/N_L) = \beta_l \propto \exp \left[ -\eta n(A) f(|l/N_L - 0.5|) \right], \; l = 1, \ldots, N_L - 1,
\]

where \( \eta \geq 0 \) is a hyperparameter and \( f : [0, \infty) \to [0, \infty) \) is an increasing function with \( f(0) = 0 \). In the following, we shall use a function \( f(x) = x \), and so our prior on \( L(A) \) is a (discretized) Laplace distribution. We provide theoretical justification for adopting this prior with exponential tails in Section A.

Another generalization of the prior on \( L(A) \) is to incorporate a spike-and-slab set-up with a spike at the middle point 1/2. In particular, one can adopt a dependent spike prior among the nodes such that once a node \( A \) is divided exactly at the middle point, so are its descendants. This generalization will substantially reduce the amount of computation in regions of the sample space where the data are either sparse or lack interesting structure, e.g., close to the uniform distribution. We implement the spike-and-slab in our software but defer the details of this generalization to Section 2.8.6 to avoid distracting the reader from the main ideas.

Given the tree prior, our PT model now consists of the two components—tree generation and conditional probability assignment. Figures 2.1(a) and 2.1(b) present a graphical model representation for each.
2.2.3 Hidden Markov Pólya Tree Models

General framework

Next we extend the above model to accommodate two recent developments in the PT literature: (i) incorporating latent state variables along the tree structure and (ii) joint modeling of multiple groups of observations. Incorporating latent variables allow more flexibly characterizing distributional features through adding prior dependency. As in recent literature, we consider incorporating discrete state variables that follow a Markov process along the tree structure. Because the description in this section always pertains to the model given the randomly generated partition tree $T$, for brevity we shall not keep stating “given $T$”.

We generalize our notation to allow observing one or more groups of i.i.d. observations. Let $G$ be the number of groups of i.i.d. observations. For the $g$th group ($g = 1, 2, \ldots, G$), let $Q_g$ be the sampling measure for that group. Let $\mathbf{Q}$ denote the collection of all $G$ sampling measures. That is, $\mathbf{Q} = \{Q_g\}_{g=1}^G$. Let $\mathbf{x}_g = (x_{g,1}, \ldots, x_{g,n_g})$ denote the observations in the $g$th group, which are i.i.d. given $Q_g$, where $n_g$, the sample size for the group, is allowed to differ across the groups. We use $\mathbf{x} = \{\mathbf{x}_g\}_{g=1}^G$ to denote the collection of all observations from all groups.

Next we specify a prior on $\mathbf{Q}$ in terms of a joint prior on the conditional probability on each $A \in T$, $\theta_g(A) = Q_g(A_l \mid A) = 1 - Q_g(A_r \mid A)$. We use latent variable modeling to incorporate prior dependency among the tree nodes. Specifically, let
\{V(A) : A \in T\} denote a collection of latent state variables, one for each \(A\), and without loss of generality, assume that \(V(A)\) takes discrete values from \(\{1, \ldots, I\}\). (In practice, the number of states \(I\) can differ among \(A\).) Joint priors of \(\theta_g(A)\) for all \(g\) and \(A\) are then defined conditionally on these latent states.

Existing literature has exploited these latent states to characterize both the within-group structure of each distribution \(Q_g\) and the between-group relationship among the \(Q_g\). An example of within-sample structures is the smoothness of each underlying distribution, which is explored in the context of density estimation (Ma, 2017). An example of between-group structures is the difference between two (or more) distributions (Soriano and Ma, 2017).

Dependent modeling of the latent states over the partition tree is desirable as \textit{a priori} one would expect interesting structures (both within-group and between-group) to exhibit themselves in a correlated manner over the sample space. For example, functions tend to have similar smoothness over adjacent locations, and two-group difference tend to be clustered in space. A computationally efficient strategy for modeling such dependency over the tree is by a hidden Markov process along the tree (Crouse and Baraniuk, 1997), which starts from the root node, \(A = \Omega\), and sequentially generates the latent states in a coarse-to-fine fashion according to (possibly node-specific) transition matrices \(\xi(A)\) whose \((i, i')\)th element is

\[
\xi_{i,i'}(A) = P(V(A) = i' \mid V(A^p) = i) \quad \text{for} \ i, i' \in \{1, \ldots, I\},
\]

where \(A^p\) denotes \(A\)'s parent. (We shall use superscript “\(p\)” to indicate the parent of a node in \(T\).) For \(A = \Omega\), since \(\Omega\) has no parent, we can simply let \(\xi_{i,i'}(\Omega)\) be constant over \(i\), representing the initial state probabilities on \(\Omega\).

Given the \(V(A)\)'s, \(\{\theta_g(A)\}_{g=1}^{G}\) can then be modeled as conditionally independent \textit{a priori}. Figure 2.1(c) presents a graphical model representation for the latent state modeling on \(G\) probability distributions by PTs given \(T\), which along with our
generalized prior on the partition tree $T$ presented in Figure 2.1(a) forms the most general version of the model we consider in this work. The specific choices of these conditional priors are problem-dependent. We give two examples below.

**Example 1: Density Estimation with Adaptive Smoothness**

An example of within-group structures that the latent state $V(A)$ can characterize is the smoothness of the density functions for the random measures. For example, Ma (2017) proposed the adaptive Pólya tree (APT) model which incorporates latent states to allow different levels of local smoothness in the underlying distribution. This is achieved by modeling the $\theta_g(A)$’s as $\text{Beta}(m(A)\nu(A), (1-m(A))\nu(A))$, where $m(A)$ is the prior mean and $\nu(A)$ the precision parameter which characterizes the smoothness of the random measure with larger $\nu(A)$ corresponding to more smoothness. Then we model the precision parameters conditional on the latent state $V(A)$, which follows a Markov process along a tree. The detail of the APT model is provided in Section 2.8.3.

**Example 2: Two-group Comparison**

In two-group comparison, we are interested in testing and identifying differences between two measures $Q = \{Q_g\}_{g=1,2}$ based on an i.i.d. sample from each. The “global” testing problem can be formulated as testing the following null and alternative hypotheses: $H_0 : Q_1 = Q_2$ vs $H_1 : Q_1 \neq Q_2$. Noting that two-group differences may exist in parts of the sample space and not others, the coupling OPT (Ma and Wong, 2011) and the multi-resolution scanning (MRS) model (Soriano and Ma, 2017) are PT-based models that allow the measures to differ on some nodes $A \in T$ and not others. This more “local” perspective on two-group comparison enables these models to not only test for $H_0$ vs $H_1$, but to identify regions on which the two measures differ. To achieve this, these models incorporate state variables that characterize
whether the conditional probabilities on each $A$ are equal:

$$V(A) = 1 \iff Q_1(A_t \mid A) \neq Q_2(A_t \mid A), \quad V(A) = 2 \iff Q_1(A_t \mid A) = Q_2(A_t \mid A).$$

(2.3)

When $V(A) = 1$, the two corresponding conditional probabilities are given independent beta priors, whereas if $V(A) = 2$, they are tied and given a single beta prior. Markov dependency among the states on different nodes are incorporated to induce the desired spatial correlation of cross-group differences. The MRS model also incorporates "an absorbing state" with which we can ignore uninteresting regions, as detailed in Section 2.8.5.

2.3 Bayesian Inference

In sum, the models we consider all share a common structure consisting of the following components: (i) the partition tree $T$ defined by the dimension and location variables $D$’s and $L$’s, which follow the priors given in Eq. (2.1); (ii) the latent state variables $V(A)$ given $T$ which follow a Markov prior; (iii) the conditional probabilities along the given tree $T$, $\{\theta_g(A)\}_{g=1}^G$, whose joint prior are specified independently across the nodes on $T$ given the latent states; and finally (iv) given the random measures $Q_g$ defined by $T$ and $\theta_g(A)$’s, we observe an i.i.d. sample $x_g$ from each $Q_g$, independently across $g$.

We shall refer to this general model class as the Hidden Markov Pólya tree, or HMPT, and summarize it below:

$$T \mid \lambda, \eta \sim p(T \mid \lambda, \eta)$$

$$\{V(A) : A \in T\} \mid \xi, T \sim \text{Markov}(\xi)$$

$$(\theta_1(A), \ldots, \theta_G(A)) \mid V(A), T \sim \text{ind} p(\theta_1(A), \ldots, \theta_G(A) \mid V(A)) \quad \text{for} \quad A \in T$$

$$x_g = (x_{g,1}, x_{g,2}, \ldots, x_{g,n_g}) \mid Q_g \overset{\text{iid}}{\sim} Q_g \quad \text{for} \quad g = 1, 2, \ldots, G.$$
The key to Bayesian inference is the ability to either compute or sample from the joint posterior \((T, V, \theta)\) given all data \(x = (x_1, \ldots, x_G)\), where \(V\) and \(\theta\) represent the totality of all latent states and conditional probabilities given \(T\) respectively. While in some problems such as density estimation one may mainly be interested in just the marginal posterior of the \(Q_g\)'s, in others such as two-group comparison where one wants to characterize the between-group relationships among the distributions, the latent states (along with \(T\)), which characterizes such relationships, are often of prime interest. In multivariate and even high-dimensional problems, the tree structure \(T\) is also of great interest as it sheds light on the underlying structures in the distributions.

To this end, we shall take advantage of recent developments in sequential Monte Carlo (SMC) sampling for tree-based models (Lakshminarayanan et al., 2013; Lu et al., 2013) and advances in message passing algorithms for PT models with Markov dependency (Ma, 2017). We introduce a hybrid algorithm that combines these two computational strategies to effectively sample from the joint posterior in high-dimensional spaces. Overall, the hybrid algorithm consists of two stages:

1. **Sampling from the marginal posterior of the partition tree**

   We design an SMC sampler—that is, a particle filter—to sample a collection of tree structures \(T^1, \ldots, T^M\) by growing each tree from coarse to fine scales. It uses one-step look-ahead message passing to construct proposal distributions for \(D(A)\) and \(L(A)\), one node at a time.

2. **Computing the conditional posterior given the sampled trees**

   Given each tree sampled by the SMC, the conditional model essentially becomes a hidden Markov process, for which we can analytically compute the exact conditional posteriors of \(V(A)\)'s and \(\theta(A)\)'s using recursive message passing.
2.3.1 SMC to Sample from Tree Posterior

In the SMC stage to sample the trees, each particle stores a realized form of a finite tree structure, and one node of each tree is divided at each step of the SMC sampling. Suppose \( T_t \) is the finite tree obtained after dividing the sample space \( t \) times in a particle, and for this tree we define the target distribution

\[
\pi_t(T_t) = P(T_t \mid x) \propto P(T_t) P(x \mid T_t).
\]

Here \( P(T_t) \) is the joint prior of the variables \( D(A) \)'s and \( L(A) \)'s for the non-leaf nodes of \( T_t \), and \( P(x \mid T_t) \) is the marginal likelihood given the tree \( T_t \) under the hierarchical model, in which \( V \) and \( \theta \) are integrated out. To sample from this target distribution, we sequentially construct a set of \( M \) particles \( \{T^m_t, W^m_t\}_{m=1}^M \), where \( T^m_t \) is a realized tree and \( W^m_t \) is the associated importance weight for the \( m \)th particle. Examples of generated trees are given in Figure 2.2, where the sample space has been divided three times, and in the next step, new partition boundaries will be added in the gray nodes.

Following Lakshminarayanan et al. (2013), we adopt in each step of the SMC a breadth-first tree-growth strategy by dividing the “oldest active” leaf node—that is, the one generated in the earliest step and is yet to be terminated in division. Further division of a node is terminated once the number of observations in that node is below a pre-set threshold (e.g., 5 in our software implementation) to avoid excessive partitioning. (In Section 2.8.8 we show through an additional experiment where data points accumulate around the boundaries of the sample space to confirm that this sample size thresholding indeed helps avoid excessive partitioning.) Otherwise a node is bisected along a boundary whose dimension and location are randomly drawn from a proposal distribution. For each particle, a finite tree \( T_t \) is formed by a sequence of decisions \( \{J_s\}_{s=1}^t \), where \( J_s = (D_s, L_s) \) correspond to all of the variables \( D(A) \) and \( L(A) \) at the \( s \)th step of the SMC.
Figure 2.2: An example of realized finite trees in the particle system obtained after the step \( t = 3 \). The numbers in the squares indicate in which step the boundaries are drawn. Among the current leaf nodes, the nodes colored gray are the oldest nodes generated in the earliest step, so they are split in the next step.

As we will see in Proposition 2.3.1, the target distribution \( \pi_t(T_t) \) has a decomposition
\[
\pi_t(T_t) = C_t \pi_{t-1}(T_{t-1}) \pi_t(J_t | T_{t-1}) w_t(T_{t-1}),
\]
where \( C_t \) is a constant independent of \( T_t \), \( \pi_t(J_t | T_{t-1}) \) a conditional distribution on \( J_t \) given \( T_{t-1} \), and \( w_t(T_{t-1}) \) a function of \( T_{t-1} \). We will choose \( \pi_t(J_t | T_{t-1}) \) as the proposal for \( J_t \) under which the corresponding importance weight will simply be \( w_t(T_{t-1}) \), independent of \( J_t \).

More specifically, suppose at the current step \( t \), we are to divide \( A_t \in T_{t-1} \), into \( A_{t,l} \) and \( A_{t,r} \) with decision \( J_t \). Let \( M_t(A_t | J_t) \) be the marginal likelihood on the node \( A_t \) under the decision \( J_t \) evaluated based on the observations in \( A_t \). That is,
\[
M_t(A_t | J_t) = \int \left[ \prod_{g=1}^G \theta_g(A_t)^{n_g(A_{t,l})}(1-\theta_g(A_t))^{n_g(A_{t,r})} \right] d\mathcal{P}(\theta_1(A_t), \ldots, \theta_G(A_t) | V(A_t) = i),
\]
where \( n_g(A_t) \) is the number of observations of the \( g \)th group included in \( A_t \). To avoid cumbersome notation, we suppress in our notation the dependency of \( M_t(A_t | J_t) \) on the observations \( x \). For example, if the \( \{\theta_g(A_t)\}_{g=1}^G \) follow independent beta priors written as \( \text{Beta}(\alpha^t_1(A_t), \alpha^t_r(A_t)) \) given \( V(A_t) = i \), then the marginal likelihood has the following expression
\[
M_t(A_t | J_t) = \prod_{g=1}^G \frac{B(\alpha^t_1(A_t)+n_g(A_{t,l}), \alpha^t_r(A_t)+n_g(A_{t,r}))}{B(\alpha^t_1(A_t), \alpha^t_r(A_t))},
\]
where \( B(\cdot, \cdot) \) is the beta function. Based on the values of \( M_t(A_t | J_t) \), we can analytically compute the proposal and the importance weight using a general recursive algorithm.
as described in the following proposition.

**Proposition 2.3.1.** For every possible decision \( J_t \) and states \( i = 1, \ldots, I \), let \( \varphi_i(A_t) \) be a function defined recursively:

\[
\varphi_i(A_t) = \begin{cases} 
\frac{\xi_{i,t}(\Omega) \mu_{i,t}(\Omega | J_t)}{\sum_{j=1}^{I} \xi_{j,t}(\Omega) \mu_{j,t}(\Omega | J_t)} & \text{if } A_t = \Omega \\
\sum_{k=1}^{I} \varphi_j(A_t) \xi_{j,k}(A_t) \mu_{i,t}(A_t | J_t) & \text{Otherwise,}
\end{cases}
\quad (2.5)
\]

where \( A_t^p \) is \( A_t \)'s parent node. Also, let \( h(J_t | A_t) \) be a function of \( J_t \) defined as

\[
h(J_t | A_t) = \sum_{i=1}^{I} \left\{ \sum_{j=1}^{I} \varphi_j(A_t^p) \xi_{j,i}(A_t) \right\} M_i(A_t | J_t) \frac{h(A_t^r)^{-n(A_t^r)} \mu(A_t^r)^{-n(A_t^r)}}{\mu(A_t)^{-n(A_t)}}.
\quad (2.6)
\]

where \( n(A) \) denotes the total number of observations included in a node \( A \). Then the target distribution \( \pi_t(T_t) \) can be expressed in terms of \( \pi_{t-1}(T_{t-1}) \) as

\[
\pi_t(T_t) = C_t \pi_{t-1}(T_{t-1}) \pi_t(J_t | T_{t-1}) w_t(T_{t-1}),
\]

where \( C_t \) is a constant and

\[
\pi_t(J_t | T_{t-1}) = \frac{P(J_t) h(J_t | A_t)}{\sum_{j_t} P(j_t) h(j_t | A_t)}, \quad w_t(T_{t-1}) = \sum_{j_t} P(j_t) h(j_t | A_t)
\]

The summation over \( j_t \) is taken over all possible decisions.

**Corollary 2.3.1.** Let \( h(J_t | A_t) \) be the function defined in Proposition 2.3.1. Then the proposal distribution \( \pi_t(D_t | T_{t-1}) \) is given by

\[
\pi_t(D_t | T_{t-1}) = \pi_t(D_t | T_{t-1}) \pi_t(L_t | D_t, T_{t-1}), \quad \text{where}
\]

1. \( \pi_t(D_t | T_{t-1}) \) is \( \text{Mult}(\tilde{\lambda}_1(A_t), \ldots, \tilde{\lambda}_d(A_t)) \) with

\[
\tilde{\lambda}_j(A_t) \propto \sum_{l=1}^{N_l} \pi_t((j, l/N_L) | T_{t-1}) \propto \lambda_j(A_t) \beta_l(A_t) h((j, l/N_L) | A_t).
\]
2. Given $D(A_t) = j$, the conditional posterior of $L(A_t)$ is

$$
\pi_t(L_t = l/N_L \mid D_t = j, T_{t-1}) = \sum_{l=1}^{N_L-1} \tilde{\beta}_t(A_t) \delta_{l/N_L}(\cdot),
$$

for $j = 1, 2, \ldots, I$ and $l = 1, \ldots, N_L - 1$ with

$$
\tilde{\beta}_t(A_t) \propto \beta(A_t) h(j, l/N_L \mid T_{t-1}).
$$

We also have an analytical expression of the incremental weight:

$$
w_t(T_{t-1}) = \sum_{j=1}^{d} \sum_{l=1}^{N_L-1} \lambda_j(A_t) \beta_l(A_t) h((j, l/N_L) \mid A_t).
$$

Remark: The recursive function $\varphi_i(A_t)$ can be computed based on $\varphi_i(A_t^p)$ with the fixed computational cost. Hence, the optimal proposal $\pi_t(J_t \mid T_{t-1})$ and the incremental weight $w_t(T_{t-1})$, which are functions of $h(J_t \mid A_t)$, can be obtained at each step with constant computational cost with complexity $O(I^2 N_L d n(A_t))$. As such, our inference algorithm scales linearly in both the dimensionality and the sample size.

The pseudo-code of the new SMC algorithm that summarizes the discussion is provided in Section 2.8.1. In the algorithm, we stop dividing $A_t$ if either the depth of $A_t$ is equal to a pre-set maximum resolution $K$ (e.g., 15) or the number of observations in $A_t$ is less than a pre-set threshold (e.g., 5). The SMC algorithm terminates when all the nodes of all the particles have been stopped. The maximum resolution $K$ controls the level of local details that the HMPT model allows to infer, and larger values of $K$ require more computational time. In a wide range of applications we have found that setting $K$ to beyond 15 to 20 leads to minimal changes in the resulting inference.
2.3.2 Posterior Computation given Sampled Tree Structures

The second stage of our inference strategy is to compute the posterior distributions of the latent states $V(A)$ and the conditional probabilities $\theta_g(A)$ given each sampled tree. In this stage we first compute the the marginal posterior of the latent states given the tree, and it is possible with a recursive message-massing algorithm as shown in Ma and Wong (2011) and Soriano and Ma (2017). The algorithm written in our notation is provided in Section 2.8.1, and we note that this algorithm works for all models under consideration. We note that in this recursive algorithm we can compute the overall marginal likelihood given the tree $T$, $P(x \mid T)$, which can be used to find the maximum a posteriori (MAP) tree among the sampled trees, i.e., the sampled tree $T^m$ that maximizes $P(T^m \mid x) \propto P(T^m)P(x \mid T^m)$. We can use this “representative” tree, along with the conditional posterior of the latent states given this tree, to visualize and summarize the posterior inference in an interpretable way.

Given both the tree and the latent states, the posterior of $\theta_g(A)$ boils down to the corresponding posterior of standard PT models on a dyadic tree, which is problem-specific as provided in the literature on each such model. We specifically use the two examples from Section 2.2.3 to demonstrate how one may use the output of the algorithm—namely the sampled trees along with the conditional posterior given the trees—to carry out inference. We note that the inference strategies for these quintessential examples are generalizable to a variety of other tasks.

Example 1: Density estimation The problem of estimating an unknown density corresponds to $G = 1$ and so we can drop the subscript $g$ to simplify the notation. We shall use the posterior mean density, also called the predictive density—$E[q(\cdot) \mid x]$—as an estimate for the density $q = dQ/d\mu$. As shown in Wong and Ma (2010) and Ma (2011)
(2017), given a tree $T^m$, we can utilize the marginal posterior of the latent states to compute the conditional predictive measure $\mathbb{E}[Q(\cdot) \mid x, T^m]$ with a top-down recursive algorithm. The algorithm is described in our generic notation in Section 2.8.3. Hence given an SMC sample of $M$ trees and weights, it is possible to integrate out the random trees and compute the posterior predictive density as follows:

$$
\mathbb{E}[q(x) \mid x] \approx \sum_{m=1}^{M} W^m \frac{\mathbb{E}[Q(B^m(x)) \mid x, T^m]}{\mu(B^m(x))},
$$

where $B^m(x) \in \mathcal{L}(T^m)$ the leaf node to which $x$ belongs.

Example 2: Two-group comparison  To compare two groups of observations using generalizations to the PT models described in Section 2.2.3, we shall compute the posterior probability of the two hypotheses $H_0$ and $H_1$. For example, when $V(A)$ is defined as in Eq (2.3), the posterior probability of the “global” null hypothesis $H_0 : Q_1 = Q_2$ is given by

$$
P(H_0 \mid x) = \sum_{T \in \mathcal{T}} P(V(A) \neq 1 \text{ for all } A \in \mathcal{N}(T) \mid T, x) P(T \mid x) \approx \sum_{m=1}^{M} W^m P(V(A) \neq 1 \text{ for all } A \in \mathcal{N}(T^m) \mid T^m, x),
$$

where the sum over $\mathcal{T}$ in the first row is over all finite trees with maximum resolution $K$ and the quantity $P(V(A) \neq 1 \text{ for all } A \in \mathcal{N}(T^m) \mid T^m, x)$ again is available analytically by message passing (details given in Section 2.8.5).

We can also detect where and how the underlying distributions differ by computing the “posterior marginal alternative probability” (PMAP) on each node $A$, along any sampled tree $T^m$:

$$
P(\theta_1(A) \neq \theta_2(A) \mid T^m, x) = P(V(A) = 1 \mid T^m, x)
$$
Reporting the PMAPs along a representative tree such as the MAP among the sampled trees can be a particularly useful visualizing tool to help understand the nature of the underlying difference. One can also report on each $A$ the estimated magnitude of the difference using a notion of “effect size” based on the log-odds ratio (Soriano and Ma, 2017), 
\[ \text{eff}(A) = \left| \log \frac{\theta_1(A)}{1-\theta_1(A)} - \log \frac{\theta_2(A)}{1-\theta_2(A)} \right|. \]

In particular, one can report the posterior expected effect size $\mathbb{E}[\text{eff}(A) | x, T]$, which can be computed using a standard Monte Carlo (not MCMC) sample from the exact posterior given the representative tree. We will demonstrate this using a mass cytometry data set in Section 2.6.

### 2.4 Theoretical Properties

Next we investigate the theoretical properties of the HMPT model. Previous theoretical analysis on the PT had mostly focused on establishing the marginal posterior consistency and contraction of the random measures $Q_g$ with respect to an unknown fixed truth (Walker and Hjort, 2001; Castillo, 2017). We, however, shall take a different perspective and instead provide asymptotic theorems regarding convergence to address the following questions that are often of practical interest:

1. What tree structures does the marginal posterior of $T$ concentrate around?
2. How does the posterior of the latent states given the tree behave?

These two questions have broad relevance in inference using PT models, and previously several authors have investigated the second question in the two-group comparison context for their variants of the PT model (Holmes et al., 2015; Soriano and Ma, 2017). In addressing the second question more generally, we aim to provide results that encompass these previous analyses as special cases. According to our limited knowledge, we are not aware of previous studies on the first question.
We will address each of the two questions in turn. Throughout this section, we consider finite PTs with maximum depth of the trees set to some value $K$. We use $\mathcal{T}^K$ to denote this collection of trees. Also, the asymptotic results are derived under the prior for $L(A)$ provided in Section 2.2.2 which can depend on the (finite) sample size. The case of an uniform priors on $L(A)$ independent of the sample size is included as a special case where the hyperparameter $\eta = 0$. Finally, we consider models that satisfy Assumption 1 and Assumption 2 described below. The models discussed in Section 2.2.3 all meet this requirement.

**Assumption 1.** For each group $g \in \{1, ..., G\}$, let $n_g$ be the sample size and $P_g$ the true probability measure from which the observations are generated. We assume that

(i) There exists $\zeta_g \in (0, 1]$ such that $\zeta_g = \lim_{n \to \infty} \frac{n_g}{n}$ for $g \in \{1, \ldots, G\}$, where $n = n_1 + \cdots + n_G$ is the total number of observations across all groups.

(ii) The sampling distribution $P_g$ satisfies $P_g \ll \mu$, and the density $p_g = dP_g/d\mu$ is positive almost everywhere.

Additionally, given the tree $T$ and the latent states, the parameters $\{\theta_g(A)\}_{g=1}^G$ are given one of the following priors (the model can adopt a mix of these priors for different combinations of $A$ and $V(A)$ values):

**Prior A:** $\theta_g(A)$ independently follow a beta prior.

**Prior B:** $\theta_1(A) = \cdots = \theta_G(A)$ and follow a beta prior.

**Prior C:** $\theta_1(A) = \cdots = \theta_G(A) \equiv c(A)$, some constant in $(0, 1)$.

Establishing the theoretical properties also requires a condition on the latent states. In particular, under some states, the support of the prior of the parameters $\{\theta_g(A)\}_{g=1}^G$ needs to include the true conditional probabilities. To describe this requirement, given a tree $T \in \mathcal{T}^K$, let $S_i(A \mid T)$ be the support of the prior
on \((\theta_1(A), \ldots, \theta_G(A))\) under the state \(V(A) = i\). Then, let \(\tau(A \mid T)\) denote the collection of “feasible states” on \(A\). (A state is “feasible” if the true conditional probabilities are in the support of the corresponding prior given the state.) That is,

\[
\tau(A \mid T) := \{i \in \{1, \ldots, I\} : (P_1(A_1 \mid A), \ldots, P_G(A_G \mid A)) \in S_i(A \mid T)\}.
\]

The next assumption states that the prior for the latent states must give positive probability for all the latent states to all simultaneously be feasible.

**Assumption 2.** For every \(T \in T^K\), \(P(V(A) \in \tau(A \mid T) \text{ for all } A \in T) > 0\).

With these assumptions, we next derive asymptotic properties for the marginal posteriors for the tree and the state variables. In the following, we use the notation \(x_n\) instead of \(x\) for the data to indicate the total sample size.

In order to describe the posterior convergence of the partition trees, we introduce a notion for “tree-based approximation for probability measures”. Let \(T\) be a finite tree and \(H\) a probability measure. Then the “tree-based approximation of \(H\) under \(T\)”, denoted by \(H \mid T\), is defined as \(H \mid T(B) = \sum_{A \in L(T)} H(A) \frac{\mu(B \cap A)}{\mu(A)}\) for any \(B \in \mathcal{B}(\Omega)\).

The following theorem then characterizes the trees the posterior concentrates on as the sample size grows.

**Theorem 2.4.1.** Let \(T^K_M\) be the collection of trees under which the tree-based approximation of the measures \(P_g\) minimizes the Kullback-Leibler divergence from the \(P_g\)’s plus a penalty term on unbalanced splits. That is,

\[
T^K_M = \arg \min_{T \in T^K} \sum_{g=1}^G \zeta_g \left\{KL(P_g || P_g | T) + \eta B_g(T)\right\}, \tag{2.7}
\]

where

\[
B_g(T) = \sum_{A \in \mathcal{N}(T)} P_g(A) f \left(\left|\frac{\mu(A)}{\mu(A)} - 0.5\right|\right).
\]
Then the marginal posterior of $T$ concentrates on $T^K_M$. That is, as $n \to \infty$, 
\[ P(T \in T^K_M \mid x_n) \xrightarrow{p} 1. \]

For the state variables, it is desirable that their posterior distribution concentrates on a collection of feasible states. Moreover, when multiple configurations of the states are feasible, it is desirable that the posterior concentrates around such configurations that provide the most parsimonious representation of the true distributions. For example, if the true conditional distribution on a node is uniform, a model that introduces a possible non-uniform structure on this node is feasible but redundant. White and Ghosal (2011) and Li and Ghosal (2014) showed that, in quite general settings of multi-resolution inference, the posterior probability of such redundant models tends to concentrate its mass on 0. By adapting their techniques, we show that the same property holds in the case of the HMPT model.

To formally describe the results, we need to define the complexity of the model specified by the latent states. Given the state $V(A) = i$, the complexity of the $\{\theta_g(A)\}_{g=1}^G$, in other words, the number of free parameters of the prior distribution under the $i$th state is denoted by $C_i(A)$. For example, for two-group comparison, 
\[ C_i(A) = \begin{cases} 
1 & \text{if } \theta_1(A) = \theta_2(A) \\
2 & \text{if } \theta_1(A) \neq \theta_2(A). 
\end{cases} \]

Next we introduce the complexity of a combination of states on the tree $T$. Given a tree $T$, let $V$ denote a combination of the state variables $\{V(A)\}_{A \in N(T)}$ and let $v = \{v(A)\}_{A \in N(T)}$ ($v(A) \in \{1, \ldots, I\}$) be one of the possible realizations of $V$. Then we define the model complexity under $v$ as follows: 
\[ C(v) = \sum_{A \in N(T)} C_{v(A)}(A). \quad (2.8) \]

The next theorem shows that the posterior distribution of the states given the tree will concentrate on those that are feasible and most parsimonious.
Theorem 2.4.2. For $T \in \mathcal{T}^K$, let $\mathcal{V}_T = \{v : v(A) \in \tau(A \mid T) \text{ for all } A \in \mathcal{N}(T)\}$. Then $P(\{V \in \mathcal{V}_T\} \cap \{C(V) = \min_{v \in \mathcal{V}_T} C(v)\} \mid T, \mathbf{x}_n) \xrightarrow{\mathbb{P}} 1$.

Remark: Consistency results for several existing models are special cases of this theorem. For example, we derive the consistency of the MRS model for two-group comparison as a corollary in Section 2.8.3.

2.5 Experiments

In this section, we carry out simulation studies to examine the performance of the HMPT model and inference algorithm. We again consider the two quintessential examples—(i) density estimation and (ii) the two-group comparison—for inferring within-group and between-group structures respectively. Details such as the settings of hyper-parameters and simulated data sets are provided in Section 2.8.9 unless explicitly described in this section.

2.5.1 Density Estimation

We first consider 2D examples to observe what kind of tree structures are obtained under the HMPT model and how prior specification in Eq. (2.2) influences the performance. After that, we move to higher dimensional cases to examine the scalability of our new SMC method and the effect of incorporating the flexible partition. For this task we compare the HMPT model with the original APT model (Ma, 2017) which also incorporates a prior on the dimension to divide but restricts partitioning at middle points. Its posterior computation is implemented by the apt function in the R package PTT.

Two-dimensional cases Simulated data are generated from the three scenarios with the densities visualized in the first row of Figure 2.4 (Details on the simulation settings are provided in Section 2.8.9) Also presented in Figure 2.4 are examples of
the posterior mean densities $\mathbb{E}[q \mid x]$ as well as the partition blocks under the MAP tree. Note that the posterior mean is computed by integrating out the unknown tree, and the MAP tree is presented to visualize key distributional features. The results for the first scenario confirms that the HMPT model is much more effective in capturing the discontinuous boundaries of the true density. For the second scenario, our model tends to draw the boundaries that surround the true clusters. In the trees given under the different values of $\eta$, however, we can see that fewer nodes were divided inside the clusters when $\eta = 0.01$. In contrast, when $\eta = 0.1$, the representative tree draws outlines of the clusters and divides regions inside of the clusters at the same time. A similar phenomenon is observed in the third scenario—under our model with flexible partitioning points, partition lines are formed around the region with high density, when $\eta = 0.1$ for the boundaries were also drawn within the high probability region. The quantitative comparison based on the KL divergence is provided in Figure fig: APT KL, which is consistent with the explanation above.

![Figure 2.3](image)

**Figure 2.3:** The average KL divergences between the estimated density and the true density for the original APT (with fixed partitioning) and the HMPT model (with flexible partitioning) with $\eta = 0.01$ and $\eta = 0.1$.

*Higher-dimensional cases*  We generate $d$-dimensional i.i.d. observations from a density with independent pairs of margins, i.e., $f(x_1, x_2, \ldots, x_d) = \prod_{j=1}^{d/2} f_j(x_{2j-1}, x_{2j})$
Figure 2.4: The posterior means of the densities and the representative trees obtained under $n = 1000$. Each column corresponds to a simulation scenario. The first row shows the true densities, the second row corresponds to the APT model (with fixed partition), and the third and fourth rows correspond to the HMPT model with flexible partitioning with parameters $\eta = 0.01$ and 0.1 respectively.
where

\[ f_j(x_{2j-1}, x_{2j}) = p_j \text{Beta}(x_{2j-1} \mid 0.25, 1) \times \text{Beta}(x_{2j} \mid 0.25, 1) \]

\[ + (1 - p_j) \text{Beta}(x_{2j-1} \mid 50/j, 50/j) \times \text{Beta}(x_{2j} \mid 50/j, 50/j), \]

with \( p_j = 0.25 + 0.7/j \). We consider two different situations: (i) the dimension \( d = 6 \), and the sample size \( n \) changes from 5,000 to 50,000; and (ii) the sample size \( n = 10,000 \) and the dimensionality changes from 10 to 100. For our method, the maximum depth \( K \) is set to 15. We show the comparison with the original APT, and also with the classical PT method, the kernel density estimation, and the Dirichlet process Gaussian mixture model (Escobar and West, 1995; Müller et al., 1996).

Figure 2.5: The wall time under five different data sets. The HMPT model with \( \eta = 0.01 \) is used. The black dashed lines indicate the average times.

Figure 2.5 presents the computational time for five different data sets. To obtain the result, we used a singe-core environment using Intel Xeon Gold 6154 (3.00 GHz) CPU. The computational time is linear in both the sample size and the dimensionality.

The models are compared based on predictive scores, that is, the average of log-predictive densities, where as the predictive density the posterior mean of the
density $E[q \mid x]$ is used. The size of the test and training sets is both $n$, and we repeat the computation for 50 pairs and take the average. The results, given in Figure 2.6, show that the HMPT model substantially outperforms the competitors by this criteria both when $d = 6$ with varying sample size and when $n$ is fixed with varying dimensionality. It is worth noting that the poor performance of the APT in the $d = 6$ case is due to the fact that available software in the apt package, which does not utilizes SMC, cannot be fit for maximal resolution $> 9$. We also investigate the performance under sample sizes $< 1000$, and the results are similar (Figure 2.7).

2.5.2 Two-group Comparison

Next we consider the two-group comparison problem, evaluate the performance of the HMPT model, and compare it to the original MRS with the “divide-in-the-middle” restriction. We use three scenarios (“Local location shift”, “Local dispersion difference”, and “Correlation”) to generate 50-dimensional data sets. (The details are provided in Section 2.8.9.) The first two scenarios involve two-group difference that lies in only parts of the sample space, or “local” differences, which will help
Figure 2.7: The comparison of the predictive performance under small sample size settings. Each point corresponds to the average of the predictive score based on 50 data sets. In the right plot, the predictive scores of the DPM model for the over 30 dimensional cases are below the displayed range.

demonstrate the usefulness of inferring the partition tree in identifying the nature of the differences. The sample size is $n_1 = n_2 = 2,000$ in all scenarios.

The original algorithm for inference under the MRS model by message passing, which is implemented by the `mrs` function in the R package MRS, is not scalable beyond about 10 dimensions even with fixed partition locations. Hence we compute the posterior for both the HMPT model and the original MRS in all scenarios with our SMC and message passing hybrid algorithm with the maximum resolution fixed to 15. We compare the performance using receiver operating characteristic (ROC) curves computed based on 200 simulated data sets under each scenario.

Figure 2.8 presents the ROC curves. For the location shift and dispersion differences, the HMPT model with flexible partitioning results in substantially higher sensitivity. For the correlation scenario, the model with fixed partitioning locations performed slightly better. This is not surprising since in this scenario the difference exists smoothly over entire ranges of the dimensions without natural “optimal” division points, and so the performance gap is the cost for searching over more possible
partition locations, none of which improves the model fit than the middle point. We again note that while the model with fixed partitioning performs well here, it works only with our new computational algorithm for data of such dimensionality.

To demonstrate the posterior model can help understand the nature of the differences, we present under each scenario the node with the highest PMAP, or $P(V(A) =$

![Figure 2.8: The ROC curves for the 50-dimensional examples.](image)

![Figure 2.9: Examples of the node with the highest PMAP under the three scenarios for two-group comparison, under the MRS with flexible partitioning and $\eta = 0.1$. The solid lines mark the boundaries of the nodes and the partition line that divides them into the two children nodes. The red triangles and the blue circle are the observations from the two groups in the node. Gray points are the observations outside the node.](image)
$1 \mid \mathbf{x} = P(\theta_1(A) \neq \theta_2(A) \mid \mathbf{x})$, in Figure 2.9. In the location shift and dispersion difference scenarios the boundaries are away from the middle point to characterize the difference, which partly explains the sensitivity gain in adopting the flexible tree prior.

2.6 Application to A Mass Cytometry Data Set

Finally, we apply our model for two-group comparison to a mass cytometry data set collected by [Kleinsteuber et al. (2016)](#). The data set records 19 different measurements including physical measurements and biomarkers on single cells in blood samples from a group of HIV patients as well as in reference samples from healthy donors. For demonstration, we compare the sample from an individual patient sample (Patient #1) and to that from a healthy donor to identify differences in immune cell profiles from these samples. The sample sizes are 29,226 for the health donor and and 228,498 for the patient, with each observation corresponding to a cell. We set $\eta = 0.1$ and the maximum depth $K$ to 25.

Given the large sample sizes, the posterior probability for the global alternative $P(Q_1 \neq Q_2 \mid \mathbf{x})$ is almost 1 and so is of less interest. Our focus is instead on identifying the cell subsets on which the samples differ and on quantifying such differences. To this end, we report the effect size eff($A$) defined in Section 2.3.2 on each node in a representative tree—the MAP among the sampled trees.

The estimated eff($A$)'s on the MAP tree up to level 9 is visualized in Figure 2.10. The full tree and the nodes with large eff($A$) are provided in Figure 2.11 and 2.12. We note that the nodes on which there is significant evidence for two-group differences, as well as those with large estimated effect sizes tend to be nested or clustered in subbranches of the tree, which is consistent with our intuition that there is spatial correlation in the two-group differences, and justifies the hidden Markov structure embedded in the MRS model.
Figure 2.10: The MAP tree for the mass cytometry data set visualized up to the 9th level. The size and the color of each node indicate the estimated $\text{eff}(A)$, and the number above a node indicates dimension in which it is split. Only the nodes with more than 50 observations are shown.

Figure 2.11: The full MAP tree for the mass cytometry data set. The size and the color indicate the estimated $\text{eff}(A)$. Only the nodes with the sample size larger than 50 are drawn. Since there are a huge number of nodes, the information on the dimension is omitted in this figure.
Figure 2.12: The solid lines delineate the nodes with the highest values of $\text{eff}(A)$ and their two children. The red triangle points and the blue circle points are the observations from the two samples in the node. The observations outside the node are in gray. In this figure, the nodes with $n_g(A) \geq 10$ ($g = 1, 2$) are chosen. Note: We adopted a spike-and-slab with higher spike probability in very deep tree levels to further speed up the computation (details given in Section 2.8.2) and that explains why many of the very deep, small nodes plotted have partition lines in the middle under the MAP tree.
2.7 Concluding Remarks

We have proposed a general framework for the PT model that incorporates a flexible prior on the partition tree and can accommodate latent state variables with Markov dependency along the partition tree. We have proposed a sampling algorithm that combines SMC and recursive message passing that can scale up to moderately high-dimensional (∼100-dim) problems. Our numerical experiments confirm that our sampling algorithm scales linearly in the sample size and the flexible partitioning tree prior can result in substantial gain in performance in some settings. Though we have mainly used two inference tasks — namely density estimation and two-group comparison — to demonstrate the HMPT model and algorithm, our approach can be readily applied to other PT models with a hidden Markov structure.

Our proposed algorithm is currently designed to be run in a single computer environment, so though the computational cost is linear to the sample size $n$, direct application to problems involving huge $n$ (e.g., $> 10^9$) is not yet feasible. It is of future interest to develop versions of the algorithm for distributed systems, which could explore either the parallel structure over nodes or parallel SMC algorithms.

2.8 Technical Details

2.8.1 Details of Posterior Computation

SMC Algorithm

The pseudo-code for the SMC algorithm for sequentially constructing the particle system $\{T^m_{t-1}, W^m_{t-1}\}_{m=1}^M$ to $\{T^m_{t}, W^m_{t}\}_{m=1}^M$ is provided below. All operations are repeated for $m = 1, \ldots, M$.

1. Choosing the current node

   From $T^m_{t-1}$, choose the node generated in the earliest step among the current
leaf nodes that have not been terminated, which is denoted by $A_t$.

2. **Obtaining the information of the parent node**
   
   Locate $A_i$’s parent node, $A_p^t$, and fetch the values of $\varphi_i(A_p^t)$ for $i = 1, \ldots, I$.

3. **Computing the necessary quantities**
   
   For all possible $J_t = (D_t, L_t)$, compute $M_i(A | J_t)$ ($i = 1, \ldots, I$) and $h(J_t | A_t)$.

4. **Dividing the current node**
   
   Compute the parameters $\tilde{\lambda}_j(A_t)$ for $j = 1, \ldots, d$ and sample
   
   $$D_t^m \sim \text{Mult} (\tilde{\lambda}_1(A_t), \ldots, \tilde{\lambda}_d(A_t)).$$
   
   Given $D_t^m$, compute the parameters $\tilde{\beta}_l(A_t)$ for $l = 1, \ldots, N_L - 1$ and sample
   
   $$L_t^m \sim \sum_{l=1}^{N_L-1} \tilde{\beta}_l(A_t) \delta_{l/N_L}(\cdot).$$
   
   Divide the current node $A_t$ with $J_t^m = (D_t^m, L_t^m)$ to update the tree $T_t^m$.

5. **Updating the importance weight**
   
   Compute the incremental weight $w_t(T_{t-1}^m)$ and update the importance weights
   
   $$W_t^m = \frac{W_{t-1}^m w_t(T_{t-1}^m)}{\sum_{m'=1}^{M} W_{t-1}^{m'} w_t(T_{t-1}^{m'})}.$$
   
   If the effective sample size $1/\sum_{m=1}^{M} (W_t^m)^2$ is less than some prespecified threshold (e.g., $M/10$), resample the particles.

6. **Computing the information on the current node for its descendants**
   
   Given $J_t^m$, compute $\varphi_i(A_t)$ for $i = 1, \ldots, I$.

---

**Remark:** A common technique in SMC is to resample the particles according to the importance weights $\{W_t^m\}_{m=1}^M$ when the effective sample size of the particles drops
below a level. In sampling trees, however, the importance weights are affected by the choice of nodes to divide in multiple steps, and so the standard resampling scheme can be too “short-sighted” and often results in sacrificing promising trees prematurely. To address this issue we follow the strategy proposed in Lu et al. (2013) by resampling the particles according to weights $a_m^t \propto (W_m^t)^\kappa$ for some $\kappa \in (0, 1]$, and compute the new importance weights proportional to $W_m^t / a_m^t$. We generally recommend using a moderate choice of $\kappa$ such as 0.5, which we have found satisfactory in a variety of numerical experiments, and will be our default choice in all of our later examples.

**Message-passing algorithm to compute the posterior of the latent states**

Now suppose the SMC algorithm has produced a collection of finite trees $\{T_m^m\}_{m=1}^M$ along with the importance weights $\{W_m^m\}_{m=1}^M$. Given each tree $T_m$, it is possible to analytically calculate the exact posterior of $\{V(A)\}_{A \in T_m}$ with recursive message passing (a form of dynamic programming), which we describe below.

For $A \in \mathcal{N}(T_m)$, let $\phi_A(i)$ be the marginal likelihood on $A$ given that $V(A) = i$. (As before we suppress the dependency of the marginal likelihood on the data to simplify notation.) Specifically,

$$\phi_A(i) = \int q(x \mid A) P(dq \mid V(A) = i), \text{ where } q(x \mid A) = \prod_{g=1}^G \prod_{z \in x_g(A)} q_g(z \mid A). \quad (2.9)$$

In Eq. (2.9), taking the integration with respect to $P(dq \mid V(A) = i)$ is equivalent to integrating out $\theta_g(A)$ as well as the $\theta_g(A')$ and $V(A')$ terms for all descendants $A'$ of $A$. Another useful quantity is the marginal likelihood on a node $A$ given the state of its parent node $V(A_p^p) = i$, which we denote as $\Phi_A(i)$ and is given by

$$\Phi_A(i) = \begin{cases} \prod_{x \in x(A)} \mu(x \mid A) & \text{if } A \text{ is a leaf node}, \\ \sum_{i' = 1}^I \xi_{i,i'}(A) \phi_A(i') & \text{if } A \text{ is a non-leaf node}. \end{cases} \quad (2.10)$$

Note that the $\Phi_A(i)$ and $\phi_A(i)$ terms are related by

$$\phi_A(i) = M_i(A \mid J(A)) \Phi_{A_i}(i) \Phi_{A_r}(i), \quad (2.11)$$
where $M_i$ is the marginal likelihood defined in (1) in the paper given under the decision $J(A) = (D(A), L(A))$ to divide $A$ into $A_l$ and $A_r$. By iteratively computing Eqs. (2.10) and (2.11) in a bottom-up fashion (i.e., starting from the leaves all the way to the root), we can compute the pair \{$(\phi_A(i), \Phi_A(i)) : A, i$\} for all nodes in the tree, and these pairs are the “messages” passed along the tree from leaf to root.

Given the values of \{$(\phi_A(i), \Phi_A(i)) : A, i$\}, we can now obtain the posterior Markov transition probability matrices of the latent states given the tree $T$, $\tilde{\xi}(A) = (\tilde{\xi}_{i,i'}(A))_{i=1}^{I \times I}$ where the $(i, i')$th element $\tilde{\xi}_{i,i'}(A) = P(V(A) = i' \mid V(A^p) = i, x, T)$ and the posterior marginal probabilities of the latent states given the tree $T$,

$$\tilde{\gamma}(A) = (\tilde{\gamma}_i(A))_{i=1}^{I} = (P(V(A) = i \mid x, T))_{i=1}^{I}.$$

Specifically, by the Bayes’ theorem, $\tilde{\xi}(A) = D_1^{-1}(A)\xi(A)D_2(A)$, where $D_1(A)$ and $D_2(A)$ are diagonal matrices with $D_1(A)_{i,i} = \Phi_A(i)$ and $D_2(A)_{i,i} = \phi_A(i)$. After computing these transition matrices, we can compute $\tilde{\gamma}(A)$ (the feedback “message”) in the top-down manner (i.e., starting from the root and down to the leaves) as $\tilde{\gamma}(\Omega) = \tilde{\xi}_1(\Omega)$ and $\tilde{\gamma}(A) = \tilde{\gamma}(A^p)\tilde{\xi}(A)$ for $A \neq \Omega$.

We note that $\Phi_{\Omega}(1)$ computed in the recursive algorithm is the overall marginal likelihood given the tree $T$, $P(x \mid T)$, which can be used to find the maximum a posteriori (MAP) tree among the sampled trees, i.e., the sampled tree $T^m$ that maximizes $P(T^m \mid x) \propto P(T^m)P(x \mid T^m)$. We can use this “representative” tree, along with the conditional posterior of the latent states given this tree, to visualize and summarize the posterior inference in an interpretable way.
2.8.2 Spike-and-slab Type Prior for $L(A)$

Introducing Auxiliary Variables

The location variable $L(A)$ follows a spike-and-slab type prior which is expressed with an auxiliary variable $R(A)$ as

$$L(A) \mid R(A) \sim 1_{\{R(A)=1\}} \delta_{1/2}(\cdot) + 1_{\{R(A)=0\}} \sum_{l=1,l/N_L \neq 1/2}^{N_L-1} \beta_l(A) \delta_{l/N_L}(\cdot),$$

where $1$ is the indicator function and the sum of the parameters $\beta_l(A)$ is 1. Under this prior, $L(A)$ follows the prior degenerated at $1/2$ if $R(A) = 1$ and otherwise follows the distribution on grid points other than the middle point. $R(A)$ follows an asymmetric hidden Markov process

$$P(R(A) = 1 \mid R(A^p) = 0) = r(A),$$

$$P(R(A) = 1 \mid R(A^p) = 1) = 1,$$

where $r(A) \in [0,1]$. $R(A) = 1$ is the absorbing state, so once $A$ is divided at the middle point, $L(A') = 1/2$ for every $A$’s descendant node $A'$. In the estimation we especially set the parameters as follows:

$$r(A) = \beta_{N_L/2}, \ \hat{\beta}_l = \sum_{l=1,l/N_L \neq 1/2}^{N_L-1} \frac{\beta_l(A)}{\beta_l(A)},$$

where \{\beta_l(A)\}_{l=1,...,N_L} is given in (2) in the paper. Under this setting the prior of $L(A)$ satisfies

$$P(L(A) = l \mid R(A^p) = 0) = \beta_l(A), \ l = 1, \ldots, N_L - 1.$$ 

Hence, $L(A)$ follows the same prior as defined in (1) in the paper unless $A$’s parent node is divided at the middle point, so the spike-and-slab prior can be seen as a natural extension.
SMC Algorithm

In the SMC algorithm, we sample values of $R(A)$ in addition to $D(A)$ and $L(A)$. If $R(A) = 1$, which is equivalent to $L(A) = 1/2$, is sampled, we conclude there is no interesting structure on the node $A$ so fix $L$ to $1/2$ for all the subsequent nodes. Hence, we need to generalize the SMC algorithm discussed in Section 3.1 to sample from the joint posterior distribution of the finite trees and the auxiliary variables $R$.

To describe this joint posterior, let $T_t$ denote the finite tree structure, which is determined by the sequence of decisions $J_{1:t}$ dividing the nodes $A_{1:t}$, and let $R_{1:t}$ be a sequence of the re-fixing variables for $A_{1:t}$. Then the target distribution we want to sample from in the SMC is defined as

$$\pi_t(T_t, R_{1:t}) = P(T_t, R_{1:t} \mid x) \propto P(T_t, R_{1:t}) P(x \mid T_t, R_{1:t}).$$

The prior $P(T_t, R_{1:t}) = P(J_{1:t}, R_{1:t})$ have a Markov chain structure on the tree, and its transition probability is decomposed as

$$P(J_t, R_t \mid J^p_t, R^p_t) = P(R_t \mid R^p_t) P(J_t).$$

where $J^p_t = J(A^p_t)$ and $R^p_t = R(A^p_t)$ ($A^p_t$ is the parent node of $A_t$). On the other hand, because $R_{1:t}$ are conditionally independent of the observations given $T_t$, the likelihood only depends on $T_t$ as follows:

$$P(x \mid T_t, R_{1:t}) = P(x \mid T_t).$$

The likelihood has the same form as in the original case without the auxiliary variables $R$’s. Thus we can obtain the following proposition as a generalization of Proposition 3.1.

**Proposition 2.8.1.** Let $h(J_t \mid A_t)$ be a function of $J_t$ defined as

$$h(J_t \mid A_t) = \sum_{i=1}^{I} \left\{ \sum_{j=1}^{I} \varphi_j(A^p_t) \xi_{j,i}(A_t) \right\} M_i(A_t \mid J_t) \mu(x(A_{l,t}) \mid A_{l,t}) \mu(x(A_{r,t}) \mid A_{r,t}).$$

43
Then the target distribution $\pi_t(T_t, R_{1:t})$ is expressed with $\pi_t(T_{t-1}, R_{1:t-1})$ as

$$\pi_t(T_t, R_t) = C \pi_t(T_{t-1}, R_{1:t}) \pi_t(R_t \mid T_{t-1}, R_t^p) \pi_t(J_t \mid T_{t-1}, R_t) w_t(T_{t-1}, R_t^p),$$

where $C$ is a constant and

$$\pi_t(R_t \mid T_{t-1}, R_t^p) = \frac{P(R_t \mid R_t^p) \sum_{j_t} P(j_t \mid R_t) h(j_t \mid A_t)}{\sum_{i=0,1} P(R_t = i \mid R_t^p) \left\{ \sum_{j_t} P(j_t \mid R_t = i) h(j_t \mid A_t) \right\}},$$

$$\pi_t(J_t \mid T_{t-1}, R_t) = \frac{P(J_t \mid R_t) h(J_t \mid A_t)}{\sum_{j_t} P(j_t \mid R_t) h(j_t \mid A_t)},$$

$$w_t(T_{t-1}, R_t^p) = \sum_{i=0,1} P(R_t = i \mid R_t^p) \left\{ \sum_{j_t} P(j_t \mid R_t = i) h(j_t \mid A_t) \right\}.$$

The summation with $j_t$ is taken over all possible decisions.

Its proof is essentially the same as Proposition 3.1 and Corollary 3.1 so it is omitted in this material. The conditional posteriors $\pi_t(R_t \mid R_t^p, T_{t-1})$ and $\pi_t(J_t \mid R_t, T_{t-1})$ are analytically obtained as follows. First, if $R_t^p = 0$, $\pi_t(R_t \mid R_t^p, T_{t-1})$ is $\text{Bernoulli}(\tilde{r}(A_t))$, where

$$\tilde{r}(A_t) = r(A_t) \sum_{j=1}^J \lambda_j(A_t) h((j, 1/2) \mid A_t)$$

$$\times \left[ r(A_t) \sum_{j=1}^J \lambda_j(A_t) h((j, 1/2) \mid A_t) + (1 - r(A_t)) \sum_{j=1}^J \sum_{l=1}^{N_L-1} \lambda_j(A_t) \beta_l(A_t) h((j, l/N_L) \mid A_t) \right]^{-1},$$

If $R_t^p = 1$, then $R_t$ is fixed to 1. Second, if $R_t = 0$, the posterior of $D_t$ and $L_t$ is the same distribution given in Section 3.1. On the other hand, if $R_t = 1$, $L_t$ is fixed to 1/2, and the conditional posterior $\pi_t(J_t \mid L_t, R_t, T_{t-1})$ is $\text{Mult}(\tilde{\lambda}_1(A_t), \ldots, \tilde{\lambda}_d(A_t))$, where

$$\tilde{\lambda}_j(A_t) \propto \lambda_j(A_t) h(j, 1/2 \mid A_t).$$
After sampling \((R_t, J_t)\), the incremental weight \(w_t(T_{t-1}, R_p^t)\) \((R_t^p = 0, 1)\) is computed as

\[
w_t(T_{t-1}, R_t^p) = r(A_t) \sum_{j=1}^J \lambda_j(A_t) h((j, 1/2) \mid A_t) + (1 - r(A_t)) \sum_{j=1}^J \sum_{l=1}^{N_L-1} \lambda_j(A_t) \beta_l(A_t) h((j, l/N_L) \mid A_t),
\]

\[
w_t(T_{t-1}, 1) = \sum_{j=1}^J \lambda_j(A_t) h((j, 1/2) \mid A_t),
\]

with which we update the importance weight \(W_t\) as \(W_t \propto W_{t-1} w_t(T_{t-1}, R_t^p)\).

The procedure to update the particle system \(\{T_{t-1}^m, W_{t-1}^m\}_{m=1}^M\) to obtain \(\{T_{t}^m, W_{t}^m\}_{m=1}^M\) is described in the following algorithm. The operations involving the index \(m\) is repeated for \(m = 1, \ldots, M\).

---

1. **Choosing the current node**
   
   From \(T_{t-1}^m\), choose the oldest note from the leaf nodes, which is denoted by \(A_t\).

2. **Obtaining the information of the parent node**
   
   Pick up \(A_t\)'s parent node, which is denoted by \(A_p^t\), and load the values of \(\varphi_i(A_p^t)\) for \(i = 1, \ldots, I\) and \(R_{t}^{m,p} = R(A_p^t)\).

3. **Computing the necessary quantities**
   
   If \(R_{t}^{m,p} = 0\), compute \(M_i(A \mid j, l/N_L)\) \((i = 1, \ldots, I)\) and \(h(j, l/N_L \mid A_t)\) for \(j = 1, \ldots, d\) and \(l = 1, \ldots, N_L - 1\).
   
   If \(R_{t}^{m,p} = 1\), compute \(M_i(A \mid j, 1/2)\) \((i = 1, \ldots, I)\) and \(h(j, 1/2 \mid A_t)\) for \(j = 1, \ldots, d\).

4. **Deciding whether to fix the partition or not**
If $R_{mt}^{m,p} = 0$, compute the parameter $\tilde{r}(A_t)$ and draw $R_{mt}^m \sim Bernoulli(\tilde{r}(A_t))$.

If $R_{mt}^{m,p} = 1$, set $R_{mt}^m$ to be 1.

5. Dividing the current node

Sample $J_t^m = (D_t^m, L_t^m)$ as follows:

- If $R_{mt}^m = 0$, compute the parameters $\tilde{\lambda}_j(A_t)$ for $j = 1, \ldots, d$ and sample
  $$D_t^m \sim \text{Mult}(\tilde{\lambda}_1(A_t), \ldots, \tilde{\lambda}_d(A_t)).$$

  Given $D_t^m$, compute the parameters $\tilde{\beta}_l(A_t)$ for $l = 1, \ldots, N_L - 1$ and sample
  $$L_t^m \sim \sum_{l=1}^{N_L-1} \tilde{\beta}_l(A_t) \delta_l/N_L(\cdot).$$

- If $R_{mt}^m = 1$, compute the parameters $\hat{\lambda}_j(A_t)$ for $j = 1, \ldots, d$ and sample
  $$D_t^m \sim \text{Mult}(\hat{\lambda}_1(A_t), \ldots, \hat{\lambda}_d(A_t)),$$

  and set $L_t^m = 1/2$.

Divide the current node $A_t$ with $J_t^m = (D_t^m, L_t^m)$ to update the tree $T_t^m$.

6. Storing the information of the state’s posterior

Given $J_t^m$, compute $\varphi_i(A_t)$ for $i = 1, \ldots, I$ and store them to the memory.

7. Updating the importance weight

Compute the incremental weight $w_t(T_{t-1}^m, R_{t}^{m,p})$ and update the importance weights as

$$W_t^m = \frac{W_{t-1}^m w_t(T_{t-1}^m, R_{t}^{m,p})}{\sum_{m'=1}^M W_{t-1}^{m'} w_t(T_{t-1}^{m'}, R_{t}^{m',p})}.$$ 

If the effective sample size $1/\sum_{m=1}^M (W_t^m)^2$ is less than some prespecified threshold ($M/10$, say), resample the particles.
2.8.3 Density Estimation with the APT Model

This section provides details on density estimation with the adaptive Pólya tree model (Ma, 2017).

Prior of Smoothness

Recall that the prior distribution of the conditional probability $\theta(A)$ is defined as $\text{Beta}(m(A)\nu(A), (1 - m(A))\nu(A))$, where $m(A)$ is the prior mean and $\nu(A)$ the precision parameter which characterizes the smoothness of the random measure with larger $\nu(A)$ corresponding to more smoothness. We introduce a prior for smoothness through modeling the precision parameters conditional on the latent state $V(A)$ with a hyperprior $\nu(A) \mid V(A) = i \sim F_i$ with the (conditional) hyperprior $F_i$ given $V(A) = i$. The hyperpriors $F_1, F_2, \ldots, F_I$ are chosen to be stochastically increasing $F_1 \prec F_2 \prec \cdots \prec F_I$. That is, $F_i([x, \infty)) \leq F_{i+1}([x, \infty))$ for all $x > 0$. The rate at which the latent states transition into a higher latent state along each subbranch of the partition tree determines the local smoothness of the underlying density.

2.8.4 Computing the Conditional Predictive Distribution given a Tree

To describe the algorithm to obtain the conditional predictive distribution given a tree $T^m$, we define for all $A \in T^m$ and $i \in \{1, 2, \ldots, I\}$, the following quantity

$$e_A(i) := \mathbb{E}[Q(A)I_{V(A)=i} \mid T^m, x].$$

All of these $e_A(i)$ terms can be computed together by a single top-down (i.e., root-to-leaf) recursion on the tree as given in the following proposition, which is essentially a generalization of results provided in Wong and Ma (2010) and Ma (2017). The proof is provided in Section 2.8.6.

**Proposition 2.8.2.** For the root node, $e_\Omega(i) = P(V(\Omega) = i \mid T^m, x)$. For a non-root
node $A$, $e_A(i)$ can be computed recursively as

$$e_A(i') = \sum_{i=1}^I \xi_{i,i'}(A) \mathbb{E}[\vartheta(A^p) \mid V(A^p) = i, T^m, x] e_{A^p}(i),$$

where $\xi_{i,i'}(A) = P(V(A) = i' \mid V(A^p) = i, T^m, x)$ and

$$\vartheta(A^p) = \begin{cases} \theta(A^p) & \text{if } A \text{ is the left child of } A^p, \\ 1 - \theta(A^p) & \text{if } A \text{ is the right child of } A^p. \end{cases}$$

Now with a recipe for obtaining all $e_i(A)$’s for a given tree, we can obtain the conditional predictive measure given the tree $T$,

$$\mathbb{E}[Q(B) \mid x, T] = \sum_{A \in \mathcal{L}(T)} \frac{\mu(B \cap A)}{\mu(A)} \sum_{i=1}^I e_A(i) \quad \text{for any Borel set } B.$$

### 2.8.5 Two-groups Comparison with the MRS Model

#### Introducing an Absorbing State

The MRS incorporates an additional state $V(A) = 3$, which introduces the same coupled prior as $V(A) = 2$, but works as an absorbing state that once $V(A) = 3$, all descendants of $A$ will remain in that state, corresponding to the case that the conditional distributions $Q_1(\cdot \mid A)$, and $Q_2(\cdot \mid A)$ are completely equal. This additional state is introduced to ignore regions that seemingly have no difference and instead explore interesting regions more in detail.

#### Additional Algorithm for the MRS Model

To describe the algorithm proposed in [Soriano and Ma (2017)](#), we keep using the same notations in Section 3.2 and 2.8.1. Given the tree structure $T$, we compute functions $\tilde{\psi}(A)$ for $A \in \mathcal{N}(T)$ in the bottom-up (from the leaf nodes to the root node) manner as follows:

$$\tilde{\psi}(A) = \begin{cases} \xi_{2,2}(A) + \xi_{2,3}(A) & \text{if } A \in \mathcal{L}(T), \\ \xi_{2,2}(A)\tilde{\psi}(A_l)\tilde{\psi}(A_r) + \xi_{2,3}(A) & \text{if } A \in \mathcal{N}(T) \setminus \{\Omega\}, \\ \xi_{1,2}(A)\tilde{\psi}(A_l)\tilde{\psi}(A_r) + \xi_{1,3}(A) & \text{if } A = \Omega. \end{cases}$$
Recall that only the first row of $\xi(\Omega)$ is meaningful as the initial distribution. Then we obtain $\tilde{\psi}(\Omega) = P(H_0 \mid T, x)$.

### 2.8.6 Proofs

**Posterior Computation**

**Lemma 2.8.1.** For the finite tree $T_t$, let $A_s$ be a node whose children nodes are leaf nodes. Then we have

$$
\pi_t(V_s = i \mid T_t) = \frac{\pi_t(T_t, V_s = i)}{\pi_t(T_t)} = \varphi_i(A_s),
$$

where $\varphi_i(A_s)$ is defined in (5) in the paper.

(Proof) Suppose that $A_s$ belongs to the $k$th layer of $T_t$. Then there is a sub-sequence $\{\rho(l)\}_{l=1}^k$ such that $A_{\rho(l)}$ belongs to the $l$th layer and

$$
\Omega = A_{\rho(1)} \supset A_{\rho(2)} \supset \cdots \supset A_{\rho(k)} = A_s.
$$

By the definition of $\pi_t(T_t, V_{1:t})$, for a sequence $\{v_l\}_{l=1}^k$ such that $v_l \in \{1, \ldots, I\}$, we obtain the expression of the conditional posterior of $\{V_{\rho(l)}\}_{l=1}^k$ as

$$
\pi_t(\{V_{\rho(l)}\}_{l=1}^k = \{v_l\}_{l=1}^k \mid T_t) \propto P(\{V_{\rho(l)}\}_{l=1}^k = \{v_l\}_{l=1}^k) \prod_{l=1}^k M_{v_l}(A_{\rho(l)} \mid J_{\rho(l)})
$$

$$
= \prod_{l=1}^k \xi_{v_{l-1}, v_l}(A_{\rho(l)}) M_{v_l}(A_{\rho(l)} \mid J_{\rho(l)}),
$$

where $v_0 = 1$. We show that for every $k = 1, \ldots, K$

$$
\pi_t(V_{\rho(k)} = v_k \mid T_t) \propto \sum_{v_1=1}^I \cdots \sum_{v_{k-1}=1}^I \left\{ \prod_{l=1}^k \xi_{v_{l-1}, v_l}(A_{\rho(l)}) M_{v_l}(A_{\rho(l)} \mid J_{\rho(l)}) \right\}
$$

$$
\propto \varphi_{v_k}(A_s)
$$

(2.12)

holds by induction. First, if $k = 1$, which is equivalent to $s = 1$, $\rho(1) = 1$, and $A_s = \Omega$, the posterior of $V(\Omega)$ is written as

$$
\pi_t(V(\Omega) = v_1 \mid T_t) \propto \xi_{1, v_1}(\Omega) M_{v_1}(A_1 \mid J_1) \propto \varphi_{v_1}(\Omega).
$$

49
Second, assume that (2.12) holds for \( k = \bar{k} \). Then, if \( k = \bar{k} + 1 \), we have

\[
\pi_t(V_k = v_k, V_{k+1} = v_{k+1} \mid T_t)
\]

\[
\propto \sum_{v_1=1}^{l} \cdots \sum_{v_{\bar{k}}=1}^{l} \pi_t(\{V_{\rho(l)}\}_{l=1}^{\bar{k}+1} = \{v_l\}_{l=1}^{\bar{k}+1} \mid T_t)
\]

\[
\propto \sum_{v_1=1}^{l} \cdots \sum_{v_{\bar{k}}=1}^{l} \left\{ \prod_{l=1}^{\bar{k}} \xi_{v_l,v_{l+1}}(A_{\rho(l)}) M_{v_l}(A_{\rho(l)} \mid J_{\rho(l)}) \right\} \xi_{v_{\bar{k}},v_{\bar{k}+1}}(A_s) M_{v_{\bar{k}+1}}(A_s \mid J_s)
\]

\[
\propto \varphi_{v_{\bar{k}}} (A_{\rho(\bar{k})}) \xi_{v_{\bar{k}},v_{\bar{k}+1}}(A_s) M_{v_{\bar{k}+1}}(A_s \mid J_s),
\]

from which we obtain

\[
\pi_t(V_{\bar{k}+1} = v_{\bar{k}+1} \mid T_t) = \sum_{v_{\bar{k}}=1}^{l} \pi_t(V_k = v_k, V_{k+1} = v_{k+1} \mid T_t)
\]

\[
\propto \sum_{v_{\bar{k}}=1}^{l} \varphi_{v_{\bar{k}}} (A_{\rho(\bar{k})}) \xi_{v_{\bar{k}},v_{\bar{k}+1}}(A_s) M_{v_{\bar{k}+1}}(A_s \mid J_s)
\]

\[
\propto \varphi_{v_{\bar{k}+1}}(A_s). \quad \square
\]

**Proof of Proposition 3.1** Let the finite tree \( T_t \) consist of a sequence of decisions \( J_{1:t} = \{J_s\}_{s=1}^{t} \), which sequentially divides nodes \( A_{1:t} = \{A_s\}_{s=1}^{t} \). To derive the proposition for the marginal posterior of \( T_t \), we first consider the joint posterior of \( T_t \) and a sequence of the state variables \( V_{1:t} \) which are defined for the nodes \( A_{1:t} \). From the structure of the HMPT model, the joint posterior is written as

\[
\pi_t(T_t, V_{1:t}) = P(J_{1:t}, V_{1:t} \mid \mathbf{x})
\]

\[
= \frac{1}{Z_t} P(J_{1:t}) P(V_{1:t}) \prod_{s=1, A_s \in \mathcal{N}(T_t)} M_{V_s}(A_s \mid J_s) \prod_{s=1, A_s \in \mathcal{L}(T_t)} \mu(\mathbf{x}(A_s) \mid A_s),
\]

(2.13)

where \( Z_t \) is the normalizing constant, and \( A_{s,l} \) and \( A_{s,r} \) are the children nodes of \( A_s \).
For $T_{t-1}$ and $T_t$, since $A_t$ is divided into $A_{t,l}$ and $A_{t,r}$, we have
\[ N(T_t) = N(T_{t-1}) \cup \{ A_t \}, \]
\[ L(T_t) = L(T_{t-1}) \setminus \{ A_t \} \cup \{ A_{t,l}, A_{t,r} \}. \]

With the expression of the joint posterior in (2.13), we obtain
\[
\pi_t(T_t, V_{1:t}) = \frac{Z_t}{Z_{t-1}} \pi_{t-1}(T_{t-1}, V_{1:t-1}) P(J_t) P(V_t \mid V_{1:t-1}) M_{V_t}(A_t \mid J_t) 
\times \frac{\mu(\mathbf{x}(A_{s,l}) \mid A_{t,l}) \mu(\mathbf{x}(A_{s,r}) \mid A_{s,r})}{\mu(\mathbf{x}(A_{s}) \mid A_{s})}. \tag{2.14}
\]

Let $A^p_t$ denote the parent node of $A_t$ and $V^p_t = V(A^p_t)$. Then, since the state variables follow the hidden Markov process, $P(V_t \mid V_{1:t-1}) = \xi_{V^p_t,V_t}(A_t)$. Integrating out $V_{1:t-1} \setminus V^p_t$ in (2.15) gives
\[
\pi_t(T_t, V^p_t, V_t) = \frac{Z_t}{Z_{t-1}} \pi_{t-1}(T_{t-1}, V^p_t) P(J_t) \xi_{V^p_t,V_t}(A_t) M_{V_t}(A_t \mid J_t) 
\times \frac{\mu(\mathbf{x}(A_{s,l}) \mid A_{t,l}) \mu(\mathbf{x}(A_{s,r}) \mid A_{s,r})}{\mu(\mathbf{x}(A_{s}) \mid A_{s})}.
\]

Because $A_t$ is a leaf node of $T_{t-1}$, by Lemma 2.8.1, we have
\[
\pi_{t-1}(T_{t-1}, V^p_t = j) = \pi_{t-1}(T_{t-1}) \pi_{t-1}(V^p_t = j \mid T_{t-1}) = \pi_{t-1}(T_{t-1}) \varphi_j(A^p_t).
\]

Hence, we obtain the expression of the marginal distribution of $T_t$ as
\[
\pi_t(T_t) = \sum_{i=1}^I \sum_{j=1}^I \pi_t(T_t, V^p_t = j, V_t = i)
\]
\[ = \frac{Z_t}{Z_{t-1}} \pi_{t-1}(T_{t-1}) P(J_t) \sum_{i=1}^I \left\{ \sum_{j=1}^I \varphi_j(A^p_t) \xi_{j,i}(A_t) \right\} M_{i}(A_t \mid J_t) 
\times \frac{\mu(\mathbf{x}(A_{s,l}) \mid A_{t,l}) \mu(\mathbf{x}(A_{s,r}) \mid A_{t,r})}{\mu(\mathbf{x}(A_t) \mid A_t)},
\]

which completes the proof. \qed
Proof of Corollary 3.1  For \( \pi_t(D_t \mid T_{t-1}) \), by Proposition 3.1, we obtain

\[
\pi_t(D_t = j \mid T_{t-1}) = \sum_{l=1}^{N_L-1} \pi_t((j, l/N_L) \mid T_{t-1})
\]

\[
\propto \sum_{l=1}^{N_L-1} P(D_t = j, L_t = l/N_L) h((j, l/N_L) \mid A_t)
\]

\[
\propto \lambda_j(A_t) \sum_{l=1}^{N_L-1} \beta_i(A_t) h((j, l/N_L) \mid A_t).
\]

On the other hand, the conditional probability of \( L_t \) is obtained as follows:

\[
\pi_t(L_t = l/N_L \mid D_t = j, T_{t-1}) \propto P(D_t = j, L_t = l/N_L) h((j, l/N_L) \mid A_t)
\]

\[
\propto \beta(A_t) h(j, l/N_L \mid T_{t-1}).
\]

The expression of \( w_t(T_{t-1}) \) immediately follows Proposition 3.1.

\[\square\]

Proof of Proposition 2.8.2  In this discussion, we suppress \( x \) and \( T^m \) in the expectation for simplicity. First, when \( A = \Omega \), by the definition \( e_\Omega(i) = \tilde{\gamma}_{1,i}(\Omega) \). Next, if \( A \) is not the root node, we can decompose \( e_A(i') \) as

\[
e_A(i') = \sum_{i=1}^{I} \mathbb{E}[Q(A)I[V(A) = i']I[V(A^p) = i]].
\]

For the summand, because \( \theta(A^p) \) and \( V(A) \) are conditionally independent given \( V(A^p) \), we obtain

\[
\mathbb{E}[Q(A)I[V(A) = i']I[V(A^p) = i]]
\]

\[
= \mathbb{E}[\mathbb{E}[\vartheta(A^p)I[V(A) = i'] \mid V(A^p)]I[V(A^p) = i]Q(A^p)]
\]

\[
= \mathbb{E}[\mathbb{E}[\vartheta(A^p)I[V(A) = i'] \mid V(A^p) = i]I[V(A^p) = i]Q(A^p)]
\]

\[
\]

\[
= \tilde{\xi}_{i,i'}\mathbb{E}[\vartheta(A^p) \mid V(A^p) = i]e_{A^p}(i).
\]
Therefore, we obtain
\[ e_A(i') = \sum_{i=1}^{l} \tilde{\xi}_{i,i'}(A) \mathbb{E}[\vartheta(A^p) \mid V(A^p) = i] e_{A^p}(i). \]

### Asymptotic Properties of the Tree Posteriors

**Lemma 2.8.2.** Let \( \{X_n^i\}_{n=1,2,...} \) (\( i = 1, \ldots, L \)) be sequences of random variables that satisfy the following conditions:

1. \( X_n^i > 0 \) for every \( i \) and \( n \).
2. As \( n \to \infty \), the following convergence occurs

\[
\frac{\log X_n^i}{n} \overset{p}{\to} c_i,
\]

where \( c_1 \geq c_2 \geq \cdots \geq c_L \).

Then we have
\[ \frac{\log \sum_{i=1}^{L} X_n^i}{n} \overset{p}{\to} c_1. \]

(Proof) We only discuss the case where there exists \( l \) such that \( c_l > c_{l+1} \). Let \( \epsilon > 0 \).

Then, by the first condition,
\[ P \left( \frac{\log \sum_{i=1}^{L} X_n^i}{n} < c_1 - \epsilon \right) \leq P \left( \frac{\log X_1^1}{n} < c_1 - \epsilon \right) \to 0 \]
as \( n \to \infty \). On the other hand,
\[ P \left( \frac{\log \sum_{i=1}^{L} X_n^i}{n} < c_1 + \epsilon \right) \]
\[ \geq P \left( \left\{ \frac{\log L \max_{i=1}^{L} X_n^i}{n} < c_1 + \epsilon \right\} \cap \bigcap_{i=1}^{l} \bigcap_{j=i+1}^{L} \{X_n^i > X_n^j\} \right) \]
\[ \geq P \left( \left\{ \frac{\log L \max_{i=1}^{l} X_n^i}{n} < c_1 + \epsilon \right\} \cap \bigcap_{i=1}^{l} \bigcap_{j=l+1}^{L} \{X_n^i > X_n^j\} \right) \to 1. \]
Lemma 2.8.3. For $T \in \mathcal{T}^K$ and $A \in \mathcal{N}(T)$, if $i \in \tau(A \mid T)$, then

$$\log M_i(A \mid j_A) = \frac{1}{n} \sum_{g=1}^{G} \zeta_g P_g(A) [P_g(A_l \mid A) \log P_g(A_l \mid A) + P_g(A_r \mid A) \log P_g(A_r \mid A)].$$

where $j_A$ is the splitting rule that divides $A$ into $A_l$ and $A_r$.

(Proof) By the result of Schwarz [1978], since the parameter $\theta(A)$ follow the beta distribution, which belongs to a continuous exponential family, the log of the marginal likelihood is written as

$$\log M_i(A \mid T) = \hat{l}_A(i, T) - \frac{r_i}{2} \log n(A) + O_p(1),$$

$$\hat{l}_A(i, T) = \log \left( \prod_{g=1}^{G} \hat{\theta}_g(A)^{n_g(A_l)} (1 - \hat{\theta}_g(A))^{n_g(A_r)} \right)$$

$$= \sum_{g=1}^{G} \left[ n_g(A_l) \log \hat{\theta}_g(A) + n_g(A_r) \log (1 - \hat{\theta}_g(A)) \right], \quad (2.16)$$

where the definition of $\hat{\theta}_g(A)$ (the MLE) and $r_i$ (the number of parameters) depend on which type of priors in Assumption 1 in (the main paper) is introduced by the state $i$:

$$\hat{\theta}_g(A) = \begin{cases} \frac{n_g(A_l)}{n(A_l)} & \text{(Prior A)}, \\ \frac{n_g(A)}{n(A)} & \text{(Prior B)}, \\ \frac{P_g(A_l)}{P_g(A)} & \text{(Prior C)}, \end{cases} \quad r_i = \begin{cases} G & \text{(Prior A)}, \\ 1 & \text{(Prior B)}, \\ 0 & \text{(Prior C)}. \end{cases} \quad (2.17)$$

Notice that, for Prior C, the constant $c(A)$ and the true measures $P_g$ must satisfy

$$c(A) = P_g(A_l \mid A)$$

for every $g = 1, \ldots, G$ because if this does not hold, the state $i$ is not included in the set of feasible states $\tau(A \mid T)$. 

54
Since \( i \in \tau(A \mid T) \), the law of large numbers gives \( \hat{\theta}_g(A) \xrightarrow{p} P_g(A_t \mid A) \). Hence, we obtain the limit of \( \hat{l}_A(i, T)/n \) as

\[
\frac{\hat{l}_A(i, T)}{n} = \sum_{g=1}^{G} \frac{n_g(\Omega) n_g(A)}{n(\Omega)} \left[ \frac{n_g(A_t)}{n_g(\Omega)} \log \hat{\theta}_g(A) + \frac{n_g(A_r)}{n_g(\Omega)} \log(1 - \hat{\theta}_g(A)) \right]
\]

\[
P \xrightarrow{p} \sum_{g=1}^{G} \zeta_g P_g(A) \left[ P_g(A_t \mid A) \log P_g(A_t \mid A) + P_g(A_r \mid A) \log P_g(A_r \mid A) \right].
\]

Lemma 2.8.4. For \( T \in T^K \), \( A \in \mathcal{N}(T) \), \( i \in \tau(A \mid T) \) and \( j \in \{1, \ldots, I\} \), we have

\[
\frac{\log M_i(A \mid T) - \log M_j(A \mid T)}{n} \xrightarrow{p} c_{i,j},
\]

where \( c_{i,j} = 0 \) if \( j \in \tau(A \mid T) \) and \( c_{i,j} > 0 \) if \( j \in \{1, \ldots, I\} \setminus \tau(A \mid T) \).

(Proof) If \( j \in \tau(A \mid T) \), obtaining the result

\[
\frac{\log M_i(A \mid T) - \log M_j(A \mid T)}{n} \xrightarrow{p} 0
\]

is straightforward from the proof of Proposition 2.8.3. Hence we consider the case of \( j \in \{1, \ldots, I\} \setminus \tau(A \mid T) \). Under the state \( j \), for every \( g \), the estimator \( \hat{\theta}_g(A) \) is defined as in (2.17), and there exists \( C_g \in (0, 1) \) such that \( \hat{\theta}_g(A) \xrightarrow{p} C_g \). By the definition of \( \tau(A \mid T) \), there exists \( g^* \) such that \( C_{g^*} \neq P_{g^*}(A_t \mid A) \). As in the proof of Proposition 2.8.3 for the difference of the marginal likelihoods, we obtain

\[
\frac{\log M_i(A \mid T) - \log M_j(A \mid T)}{n} \xrightarrow{p} \sum_{g=1}^{G} \zeta_g P_g(A) \Lambda_g,
\]

\[
\Lambda_g = P_g(A_t \mid A) \log \frac{P_g(A_t \mid A)}{C_g} + P_g(A_r \mid A) \log \frac{P_g(A_r \mid A)}{1 - C_g}.
\]

Because \( \Lambda_g \) is the KL divergence of the two discrete distributions, \( \Lambda_g \geq 0 \) for all \( g \) and \( \Lambda_{g^*} > 0 \). By Assumption 1, this result implies that

\[
\sum_{g=1}^{G} \zeta_g P_g(A) \Lambda_g > 0.
\]
Proof of Theorem 4.1 (non-informative prior) and Theorem 4.2 This section provides the proofs of Theorem 4.1 under the assumption that the prior of the location variables is independent of the sample size, that is, \( \eta = 0 \) in Eq. (2) in the paper. The general case is discussed in the next section.

In this proof, we modify the notation for the marginal likelihood defined in Eq. (4) in the paper and use \( M_i(A \mid T) \) to represent the likelihood on \( A \) of the tree \( T \) under the \( i \)th state to reflect its dependency on the tree structure.

Let \( T \in T^K \) and \( V \) denote a set of a combination of the states for all of the non-leaf nodes of \( T \). Notice that an element of \( V \) does not need to satisfy \( P(V = v) > 0 \), where \( V \) is the totality of the state variables. In the following proof, for \( v \in V \), \( v(A) \) denotes a state on a node \( A \). Let \( l(v, T) \) denote the log of the joint likelihood function

\[
l(v, T) = \log P(x_n \mid T, v) = \sum_{A \in N(T)} l_A(v(A), T) + \sum_{A \in L(T)} \log \mu(x_n(A) \mid A), \tag{2.18}
\]

where \( l_A(v(A), T) = \log M_A(v(A) \mid T) \). By [Schwarz (1978)](1978), this likelihood \( l_A \) has the following expression

\[
l_A(v(A), T) = \hat{l}_A(v(A), T) - \frac{r_v(A)}{2} \log n(A) + O_p(1),
\]

where \( \hat{l}_A \) and \( r_i \) are defined in (2.16). Let \( \tilde{v} \in V \) be a collection of states such that, for all \( A \in N(T), \theta_g(A) \) is fixed to \( \mu(A_l)/\mu(A) \). For \( \tilde{v} \), we have

\[
l_A(\tilde{v}, T) = \sum_{A \in N(T)} \left\{ n(A_l) \log \left( \frac{\mu(A_l)}{\mu(A)} \right) + n(A_r) \log \left( \frac{\mu(A_r)}{\mu(A)} \right) \right\} + \sum_{A \in L(T)} \log \mu(x_n(A) \mid A)
\]

\[= \log \mu(x) = 0.\]

Hence \( l(v, T) \) is rewritten as

\[
l(v, T) = l(v, T) - l(\tilde{v}, T) = \sum_{A \in N(T)} \left\{ \hat{l}_A(v(A), T) - \hat{l}_A(\tilde{v}(A), T) \right\} - \frac{C(\tilde{v})}{2} \log n + O_p(1).
\]
For the part inside of the braces, when \( v \) is replaced with \( v_T \in V_T \), where \( V_T = \{ v : v(A) \in \tau(A \mid T) \text{ for all } A \in \mathcal{N}(T) \} \), the definition of \( \hat{l}_A \) gives

\[
\hat{l}(v_T(A), T) - \hat{l}(v(A), T) = \sum_{g=1}^{G} \sum_{A \in \mathcal{N}(T)} n_g(A) \left[ \frac{n_g(A_l)}{n_g(A)} \log \frac{\hat{\theta}_g(A)}{\mu(A_l \mid A)} + \frac{n_g(A_r)}{n_g(A)} \log \frac{1 - \hat{\theta}_g(A)}{\mu(A_r \mid A)} \right]
\]

\[
P \sum_{g=1}^{G} \zeta_g P_g(A) \left[ P_g(A_l \mid A) \log \frac{P_g(A_l \mid A)}{\mu(A_l \mid A)} + P_g(A_r \mid A) \log \frac{P_g(A_r \mid A)}{\mu(A_r \mid A)} \right].
\]

For all \( A \in L(T) \), there exists an unique sequence of nodes

\[
\Omega = B_{A,0} \supset B_{A,1} \supset \cdots \supset B_{A,K} = A,
\]

where \( B_{A,k} \in T (k = 0, \ldots, K) \) is a node in the \( k \)th level. With this sequence, we obtain the limit of the scaled log-likelihood as

\[
l(v_T, T)
\]

\[
P \sum_{A \in \mathcal{N}(T)} \sum_{g=1}^{G} \zeta_g P_g(A) \left[ P_g(A_l \mid A) \log \frac{P_g(A_l \mid A)}{\mu(A_l \mid A)} + P_g(A_r \mid A) \log \frac{P_g(A_r \mid A)}{\mu(A_r \mid A)} \right]
\]

\[
= \sum_{g=1}^{G} \zeta_g \sum_{A \in \mathcal{N}(T)} \left[ P_g(A_l) \log \frac{P_g(A_l \mid A)}{\mu(A_l \mid A)} + P_g(A_r) \log \frac{P_g(A_r \mid A)}{\mu(A_r \mid A)} \right]
\]

\[
= \sum_{g=1}^{G} \zeta_g \sum_{A \in L(T)} P_g(A) \log \frac{P_g(A)}{\mu(A)} = \sum_{g=1}^{G} \zeta_g KL(P_g \mid \mid \mu).
\]

Because \( P_g \mid T \) admits the density function

\[
p_g \mid T(x) = \sum_{A \in L(T)} 1_A(x) \frac{P_g(A)}{\mu(A)}, \quad x \in \Omega
\]
the KL divergence in (2.20) is rewritten as

\[ KL(P_g|T||\mu) = \sum_{A \in \mathcal{L}(T)} P_g(A) \log \frac{P_g(A)}{\mu(A)} \]

\[ = \int p_g \sum_{A \in \mathcal{L}(T)} 1_A \log \frac{P_g(A)}{\mu(A)} d\mu \]

\[ = \int p_g \log p_g|T d\mu \]

\[ = \int p_g(x) \log \frac{p_g(x)}{\mu(x)} d\mu(x) - \int p_g \log \frac{p_g|T}{p_g} d\mu \]

\[ = KL(P_g||\mu) - KL(P_g||P_g|T). \]

Because \( KL(P_g||\mu) \) is independent of \( T \), we obtain another expression of \( T^K_M \) in (7) (in the paper) as

\[ T^K_M = \arg \max_{T \in T^K} G \sum_{g=1}^G \zeta_g KL(P_g|T||\mu) \]

By Lemma 2.8.4 and (2.18), for \( v \in \mathcal{V} \setminus \mathcal{V}_T \), we can show that

\[ \lim_{n \to \infty} \frac{l(v, T) - l(v_T, T)}{n} = \lim_{n \to \infty} \frac{\sum_{A \in \mathcal{N}(T)} l_A(v, T) - l_A(v_T, T)}{n} > 0, \quad (2.21) \]

and \( \lim_{n \to \infty} l(v, T)/n \) exists. Hence, for \( T_M \in T^K_M, v' \in \mathcal{V}_M, T \in T^K \setminus T^K_M \) and \( v \in \mathcal{V} \), we have

\[ \lim_{n \to \infty} \frac{l(v', T_M) - l(v, T)}{n} \geq \sum_{g=1}^G \zeta_g KL(P_g|T_M||\mu) - \sum_{g=1}^G \zeta_g KL(P_g|T||\mu) > 0. \]

Hence, for such \( T_M \) and \( v' \), we obtain

\[ \frac{P(x_n | T)}{P(x_n | T_M)} = \frac{P(T) \sum_{v \in \mathcal{V}} \exp(l(v, T)) P(v)}{P(T_M) \sum_{v \in \mathcal{V}} \exp(l(v, T_M)) P(v)} \leq \sum_{v \in \mathcal{V}} \frac{\exp(l(v, T)) P(T) P(v)}{\exp(l(v', T_M)) P(T_M) P(v')} \to 0. \]

This result implies \( P(T \in T^K_M | x_n) \overset{p}{\to} 1 \), which completes the proof of Theorem 4.1.
To prove Theorem 4.2, we fix $T \in \mathcal{T}_K$ and define a set $\mathcal{S}_T$ as

$$\mathcal{S}_T = \left\{ v \in \mathcal{V}_T \mid v \in \arg \min_{v' \in \mathcal{V}_T} C(v') \right\}.$$ 

Then we want to show $P(\mathbf{V} \in \mathcal{S}_T \mid T, \mathbf{x}_n) \overset{p}{\to} 1$. The result (2.21) implies

$$p(\mathbf{V} \in \mathcal{V}_T \mid T, \mathbf{x}_n) \overset{p}{\to} 1,$$

so we only need to compare the elements of $\mathcal{V}_T$. Let $v \in \mathcal{V}_T \setminus \mathcal{S}_T$ and $v' \in \mathcal{S}_T$. For the difference of the log likelihoods, we have

$$l(v', T) - l(v, T) = \sum_{A \in \mathcal{N}(T)} \left[ \hat{l}_A(v'(A), T) - \hat{l}_A(v(A), T) \right] + \frac{C(v) - C(v')}{2} \log n + O_p(1),$$

where $\hat{l}_A$ and $C$ is defined in (2.17) and (8) (in the paper), respectively. If $v(A)$ and $v'(A)$ introduce the same type of the prior (e.g., Prior A and Prior A), because the corresponding estimators $\hat{\theta}_g(A)$ have the same form,

$$\hat{l}_A(v'(A), T) - \hat{l}_A(v(A), T) = 0.$$

On the other hand, if $v(A)$ and $v'(A)$ introduce different types of the prior (e.g., Prior A and Prior B), because they are the maximized log-likelihood under the two nested hypotheses,

$$-2[\hat{l}_A(v'(A), T) - \hat{l}_A(v(A), T)]$$

weakly converges to the $\chi^2$ distribution (Wilks, 1938). Hence, we obtain

$$\frac{l(v', T) - l(v, T)}{\log n} \overset{p}{\to} \frac{C(v) - C(v')}{2} > 0,$$

which implies $P(\mathbf{V} \in \mathcal{S}_T \mid T, \mathbf{x}_n) \overset{p}{\to} 1$. 

**Proof of Theorem 4.1 (informative prior)** This section provides the proofs of Theorem 4.1 for the case of the informative prior ($\eta > 0$). In the proof, we often use
the results provided in the previous section on the non-informative prior and use the same notations.

We first discuss the asymptotic behavior of the prior of trees, which is in this case dependent on the sample size \( n \). We only need to focus on the prior of the location variables \( L \) since this is only the component that depends on the sample size as given in Eq. (2) (in the paper). By the fact that

\[
\frac{n(A)}{n} \xrightarrow{p} \sum_{g=1}^{G} \zeta_g P_g(A)
\]

and Lemma 2.8.2 we obtain the limit

\[
\frac{\log P(L(A) = \mu(A)/\mu(A))}{n} \xrightarrow{p} -\eta \sum_{g=1}^{G} \zeta_g P_g(A) f \left( \left| \frac{\mu(A)}{\mu(A)} - 0.5 \right| \right).
\]

Since given \( T \in \mathcal{T}^K \), the location variables are all independent and the prior of the dimension variables is independent of the sample size \( n \), the log-tree prior \( l(T) = \log P(T) \) has a limit as follows:

\[
\frac{l(T)}{n} \xrightarrow{p} -\eta \sum_{g=1}^{G} \zeta_g B_g(T),
\]

which is the penalty term introduced in Theorem 4.1.

We next define \( \psi(T) \) as

\[
\psi(T) = \sum_{g=1}^{G} \zeta_g \left\{ KL(P_g||\mu) - \eta B_g(T) \right\}.
\]

By the the proof of Theorem 4.1 for the non-informative case, this is a limit under \( \mathbf{v} \in \mathcal{V}_T \) (this \( \mathcal{V}_T \), and \( \mathcal{V} \), a collection of possible combination of the state variables on the tree, are defined in the proof for the non-informative case), that is,

\[
\frac{l(T) + l(\mathbf{v}, T)}{n} \xrightarrow{p} \psi(T)
\]
as \( n \to \infty \). On the other hand, for \( \mathbf{v} \in \mathcal{V} \setminus \mathcal{V}_T \), the limit exists and

\[
\text{p-lim} \frac{l(T) + l(\mathbf{v}, T)}{n} < \psi(T).
\]

As discussed in the proof of Theorem 4.1, we have the equivalence

\[
\arg \max_{T \in \mathcal{T}_M} \sum_{g=1}^{G} \zeta_g KL(P_g | \mu) = \arg \min_{T \in \mathcal{T}_M} \sum_{g=1}^{G} \zeta_g KL(P_g || P_g | T),
\]

which implies that

\[
T_M^K = \arg \max_{T \in \mathcal{T}_K} \psi(T).
\]

Hence, for \( T_M \in \mathcal{T}_M^K \), \( \mathbf{v}' \in \mathcal{V}_{T_M} \), \( T \in \mathcal{T}_K^M \setminus \mathcal{T}_M^K \), and \( \mathbf{v} \in \mathcal{V} \), we have

\[
\text{p-lim}_{n \to \infty} \frac{(l(T_M) + l(\mathbf{v}', T_M)) - (l(T) + l(\mathbf{v}, T))}{n} \geq \psi(T_M) - \psi(T) > 0.
\]

Hence, for such \( T, \mathcal{T}_M \) and \( \mathbf{v}' \), it follows that

\[
\frac{P(T, \mathbf{x}_n)}{P(T_M, \mathbf{x}_n)} = \frac{P(T)P(\mathbf{x}_n | T)}{P(T_M)P(\mathbf{x}_n | T_M)} = \frac{P(T) \sum_{\mathbf{v} \in \mathcal{V}} P(\mathbf{v}) \exp(l(\mathbf{v}, T))}{P(T_M) \sum_{\mathbf{v} \in \mathcal{V}} P(\mathbf{v}) \exp(l(\mathbf{v}, T_M))} \leq \sum_{\mathbf{v} \in \mathcal{V}} \frac{\exp(l(T) + l(\mathbf{v}, T))P(\mathbf{v})}{\exp(l(T_M) + l(\mathbf{v}', T_M))P(\mathbf{v}') \to 0.}
\]

Therefore, \( P(T \in \mathcal{T}_M^K | \mathbf{x}_n) \xrightarrow{p} 1. \)

### 2.8.7 Consistency for the MRS Model with the Flexible Partitioning

To describe the consistency, for a possible node \( A \), we define a variable \( Z(A) \) as follows:

\[
Z(A) = \begin{cases} 
1 & \text{if } V(A) = 1, \\
0 & \text{if } V(A) \in \{2, 3\}.
\end{cases}
\]

Hence, \( \theta_1(A) = \theta_2(A) \) if \( Z(A) = 0 \) and \( \theta_1(A) \neq \theta_2(A) \) with probability one if \( Z(A) = 1 \). Then we can obtain the following consistency result.
Corollary 2.8.1. Let $Z = \{Z(A)\}_{A \in \mathcal{N}(T)}$ and $z = \{z(A)\}_{A \in \mathcal{N}(T)}$ be a collection of $Z(A)$ on $T \in \mathcal{T}^K$ and one of its realizations, respectively. If $P(Z = z) > 0$ for any possible $z$, then

$$P(Z(A) = 1_{\{P_1(A|A) \neq P_2(A|A)\}} \text{ for all } A \in \mathcal{N}(T) | T, x_n) \xrightarrow{p} 1,$$

where $1$ is the indicator function, and

$$P(Z(A) = 1_{\{P_1(A|A) \neq P_2(A|A)\}} \text{ for all } A \in \mathcal{N}(T) | x_n) \xrightarrow{p} 1,$$

where $T$ is random.

(Proof) In this case, $\mathcal{V}_T$ in Theorem 4.2 is written as

$$\mathcal{V}_T = \{v \mid v(A) = 1 \text{ if } P_1(A_t | A) \neq P_2(A_t | A)\}.$$

We additionally define $\tilde{\mathcal{V}}_T$ as

$$\tilde{\mathcal{V}}_T = \{v \mid v(A) = 2 \text{ if } P_1(A_t | A) = P_2(A_t | A)\}.$$

Then, under the condition that $v \in \mathcal{V}_T$, the complexity $C(v)$ is minimized if and only if $v \in \tilde{\mathcal{V}}_T$. Hence, by Theorem 4.2 we obtain

$$P(Z(A) = 1_{\{P_1(A|A) \neq P_2(A|A)\}} \text{ for all } A \in \mathcal{N}(T) | T, x_n) = P(\mathcal{V} \in \mathcal{V}_T \cap \tilde{\mathcal{V}}_T | T, x_n) \xrightarrow{p} 1.$$

We can show the second result by using Theorem 4.1 as follows:

$$P\left(V \in \mathcal{V}_T \cap \tilde{\mathcal{V}}_T | x_n\right) = \sum_{T \in \mathcal{T}^K} P\left(V \in \mathcal{V}_T \cap \tilde{\mathcal{V}}_T | T, x_n\right) P(T | x_n) \geq \arg\min_{T_M \in \mathcal{T}_M^K} P\left(V \in \mathcal{V}_T \cap \tilde{\mathcal{V}}_T | T_M, x_n\right) P(T \in \mathcal{T}_M^K | x_n) \xrightarrow{p} 1.$$
2.8.8 Numerical Evaluation of the Density Estimators under Different Complexity of Data Generating Processes

In this section, we observe the behavior of our proposed SMC algorithm in density estimation in a case where observed points concentrate around certain points or the boundary of the sample space, which occurs in real data analysis in many cases. We consider the following scenario: the variables $X_j$ ($j = 1, \ldots, d$) independently follow the beta distribution $\text{Beta}(2^a, 2^a)$, where the parameter $a$ takes values from $\mathbb{R}$. When $a = 0$, the data is uniformly distributed in the sample space $(0, 1]^d$, and when $a$ is positive, the distribution concentrates on the central point $(0.5, 0.5, \ldots, 0.5)$. When $a$ is negative, the distribution concentrates on the vertices of the hyper-cube $(0, 1]^d$. For the cases of $a < 0$, many values generated from the beta distribution tend to be indistinguishable from 0 and 1, and we found it often made the behavior of our proposed SMC algorithm, which is designed for continuous distributions without mass points, unstable. As such, for the negative $a$’s, we use an approximative distribution in which the conditional distributions on $(0, 10^{-5}]$ and $(0.9 - 10^{-5}, 1.0]$ are replaced with the uniform.

In the estimation, we used the APT model as described in Section 5.1. The number of particles is 1,000, and the maximum resolution is set to 10 or 15. The data is generated under $a = (-3, -2, -1, 0, 1, 2, 3)$, $n = 10^2, 10^3, 10^4$, and $d = 1, 2, 3, 4$, and for each combination, 50 data sets under different random seeds are generated. The performance is assessed based on $L_1$ distance between the estimated density (posterior mean) and the true distribution. Since the $L_1$ distance is difficult to compute analytically, we use the Monte Carlo approximation with 10000 values generated from the true distributions.

Figure 2.13 provides the average $L_1$ distance under the different settings. When
the distribution is uniform \((a = 0)\), we can see that the \(L_1\) distance is very small, and this can be understood as an advantage of using the APT model to learn the shrinkage level adaptively. On the other hand, especially when \(a < 0\) and \(d\) is large, the distance is larger for \(a \neq 0\) though the performance is slightly improved if the maximum resolution \(K\) is larger. One possible reason to explain this result is that the APT model, with which we learn the smoothness of unknown densities, is not designed to capture such extremely spiky distributions. Thus the performance would be improved by modifying the prior distributions but this discussion is out of this research’s scope because our main contribution is proposing the algorithm that works for many types of PT-based models.

Figure 2.14 compares the number of nodes included in the MAP trees. From this result, we can see that the number is larger for the larger maximum resolution \(K = 15\) as naturally expected but tends to be smaller when the parameter \(a\) is positive. The latter phenomenon is explained by the rule that the SMC algorithm stops splitting nodes when the number of included observations is below the threshold such as 5. When \(a\) is positive, since the distribution concentrates around the central point, nodes with only a few observations tend to be generated in early levels and thus are no longer split, resulting in a smaller number of nodes on the tree. Hence this numerical result clarifies our proposed algorithm’s tendency of “ignoring” nodes with only a few observations “zooming up” nodes including many observations.

2.8.9 Details of the Experiments

Density Estimation

Hyper-parameter settings For the HMPT model and the original APT model, we use the common settings following [Ma (2017)]. The transition matrix for the latent states
on each $A$ that characterize different smoothness levels of the density is given by

$$\xi_{i,i'}(A) \sim \begin{cases} e^{\beta(i-i')} & \text{if } i \leq i', \\ 0 & \text{if } i > i', \end{cases}$$

where $\beta = 0.1$ and the number of states $I = 5$. For the $i(< I)$th state, given $V(A) = i$, the precision $\nu(A)$ follows the prior

$$\log_{10} \nu(A) \sim \text{Unif}(a(i), a(i + 1)],$$

where $a(i) = L + (i - 1)(U - L)/(I - 1)$ with $L = -1$ and $U = 4$. For the $I$th state, $F_I = 1_{\infty}$, so $\nu(A)$ is fixed to $\infty$. This is equivalent to stopping the partition and putting the conditional distribution $\mu(\cdot \mid A)$ on $A$. In the computation, this uniform distribution is approximated by 5 evenly spaced grid points.

**Simulation scenarios in the two-dimensional cases** The data sets are simulated from the following distributions.

1. “Blocks”:

$$\frac{1}{3} 1_{[0.1,0.45] \times [0.35,0.9]}(x_1, x_2) + \frac{1}{3} 1_{[0.2,0.8] \times [0.45,0.5]}(x_1, x_2)$$

$$+ \frac{1}{3} 1_{[0.7,0.9] \times [0.05,0.6]}(x_1, x_2)$$

2. “Clusters”:

$$\frac{1}{10} \text{Beta}(x_1 \mid 1, 1) \times \text{Beta}(x_2 \mid 1, 1) + \frac{3}{10} \text{Beta}(x_1 \mid 15, 45) \times \text{Beta}(x_2 \mid 15, 45)$$

$$+ \frac{3}{10} \text{Beta}(x_1 \mid 45, 15) \times \text{Beta}(x_2 \mid 22.5, 37.5)$$

$$+ \frac{3}{10} \text{Beta}(x_1 \mid 37.5, 22.5) \times \text{Beta}(x_2 \mid 45, 15)$$

3. “Smooth”:

$$\text{Beta}(x_1 \mid 10, 20) \text{Beta}(x_2 \mid 10, 20)$$
Implementation details for the higher dimensional cases  For the algorithm of the original APT (implemented by the apt function in the PTT package), in the first case with \(d = 6\), the maximum resolution is fixed to 9 because setting higher values lead to insufficient memory. Also, because the apt function does not scale if the dimension is beyond \(d \approx 10\), in the second case with large \(d\), we used the proposed SMC algorithm to carry out inference for the original APT model as well, which corresponds to setting \(N_L = 2\).

In the classical PT method, the Dirichlet prior is set to \(\text{Dir}(0.1k^2, ..., 0.1k^2)\) (\(k\): the depth of the node), and the maximum depth is set to 15. For the Dirichlet process Gaussian mixture model, we used the \text{PYdensity} function in the R package \text{BNPmix} [Corradin et al., 2021], in which the strength and discounting parameters are set to 10 and 0, respectively, the covariance matrices are all diagonal (\texttt{model = "DLS"}), and the size of the burn-in period and the sampling is both set to 1000. For the Gaussian kernel density estimation, it is implemented by the \text{kde} function in the R package \text{ks}. We set the bandwidth matrix by using \text{Hpi.diag} function which selects the optimal diagonal matrix.

Two-group Comparison

Hyper-parameter Settings  For the transition matrix \(\xi(A)\), we use the form proposed in [Soriano and Ma, 2017] for incorporating multiple testing control

\[
\xi(A) = \begin{bmatrix}
(1 - \rho)\gamma & (1 - \rho)(1 - \gamma) & \rho \\
(1 - \rho)\gamma 2^{-k} & (1 - \rho)(1 - \gamma 2^{-k}) & \rho \\
0 & 0 & 1
\end{bmatrix},
\]

where \(\gamma \in (0, 1)\), and \(\rho \in (0, 1)\), and \(k\) is the depth of \(A\), and we set \((\gamma, \rho) = (0.3, 0.3)\) following recommendations in that paper.

Simulation scenarios
1. “Local location shift”: For \( j = 1, \ldots, 25 \),

\[
(X_{1,2(j-1)+1}, X_{1,2j}) \sim \frac{1}{3} N(\mu_1, \Sigma) + \sum_{l=2}^{3} \frac{1}{3} N(\mu_l, \Sigma),
\]

\[
(X_{2,2(j-1)+1}, X_{2,2j}) \sim \frac{1}{3} N(\mu_1 + \delta_j, \Sigma) + \sum_{l=2}^{3} \frac{1}{3} N(\mu_l, \Sigma),
\]

where \( \delta_j = -0.5 \) for \( j = 1, \ldots, 5 \) and 0 for \( j = 6, \ldots, 25 \).

2. “Local dispersion difference”: For \( j = 1, \ldots, 25 \),

\[
(X_{1,2(j-1)+1}, X_{1,2j}) \sim \frac{1}{3} N(\mu_1, \Sigma) + \sum_{l=2}^{3} \frac{1}{3} N(\mu_l, \Sigma),
\]

\[
(X_{2,2(j-1)+1}, X_{2,2j}) \sim \frac{1}{3} N(\mu_1 + \Delta_j, \Sigma) + \sum_{l=2}^{3} \frac{1}{3} N(\mu_l, \Sigma),
\]

where \( \Delta_j = -0.4 \) for \( j = 1, \ldots, 5 \) and 0 for \( j = 6, \ldots, 25 \).

3. “Correlation”: For \( j = 1, \ldots, 25 \),

\[
(X_{1,2(j-1)+1}, X_{1,2j}) \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)
\]

\[
(X_{2,2(j-1)+1}, X_{2,2j}) \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \delta_j \\ \delta_j & 1 \end{bmatrix}\right),
\]

where \( \delta_j = 0.75 \) for \( j = 1, \ldots, 5 \) and \( \delta_j = 0 \) for \( j = 6, \ldots, 25 \).

In the “local location shift” and “local dispersion difference”, the parameters are

\[
\mu_1 = (-2.5, 1.0), \quad \mu_2 = (1.0, -2.0), \quad \mu_3 = (2.0, 2.5), \quad \Sigma = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.7 \end{bmatrix}.
\]
Figure 2.13: The average $L_1$ distance from the true distribution with error bars that indicate the standard deviation obtained under different sample size, dimensionality, and values of the parameter $a$. 
Figure 2.14: The average number of nodes in the MAP trees with error bars that indicate the standard deviation obtained under different sample size, dimensionality, and values of the parameter $a$. 
3

Unsupervised Tree Boosting for Learning Probability Distributions

3.1 Introduction

In supervised learning such as classification and regression, boosting is acknowledged as one of the most powerful algorithms. It is acclaimed for the ability to overcome the curse of dimensionality and achieve a desirable balance in bias-variance trade-off. The most popular boosting algorithms can be thought of as sequentially fitting an additive ensemble of weak learners, often in the form of regression or classification trees (e.g., Friedman, 2001; Hastie et al., 2009). The success of tree boosting in supervised problems suggests that a similar strategy might also prevail in unsupervised problems, where the ultimate objective involves learning the structures of some unknown probability distribution based on a collection of training data from that distribution.

We aim to formulate a new additive tree model framework for probability distributions along with an unsupervised boosting algorithm that circumvents these difficulties. Our approach is motivated by the observation that some highly effec-
tive supervised tree boosting algorithms are fit in each iteration based on a set of residuals rather than the original observations, thereby substantially simplifying the optimization task in each iteration. To realize this strategy in the unsupervised context, we introduce a new notion of addition for probability measures which leads to a natural concept of the residual of an observation after “subtracting” a probability measure from it. While a natural idea to define addition on probability measures is to embed them as elements in a space where the operation of addition—such as through taking weighted averages—is readily available, it is not straightforward to find such an embedding that renders a conceptually and computationally simple notion of “residualization” of an observation, which removes a fitted measure from the underlying sampling distribution.

Our new notions of “addition” and “residualization” for probability distributions are formulated in terms of cumulative distribution function (CDF) transforms and compositions. We start from the case of univariate distributions, for which the addition of two measures can be defined simply in terms of a composition of their CDFs whereas the residual of an observation from subtracting a measure is simply the application of the corresponding CDF transform to that observation. In generalizing this notion of addition to multivariate distributions on $\mathbb{R}^d$ with $d > 1$, however, the classical notion of the multivariate CDF, which maps $\mathbb{R}^d$ to the interval $(0, 1]$ is unsatisfactory, as easily seen, for example, by the fact that one can neither define a composition of two such CDFs nor define residuals that still lie in $\mathbb{R}^d$. More fundamentally, multivariate CDFs do not preserve a set of “group-like” properties of 1D CDFs that underly the notion of addition and residualization. Interestingly, we show that a proper notion of the CDF for multivariate distributions, which maps $\mathbb{R}^d$ to $\mathbb{R}^d$, does exist and it naturally generalizes the notions of addition and residuals to multivariate settings.

Based on these new notions, we introduce an unsupervised tree boosting algo-
algorithm for learning probability measures based on forward-stagewise (FS) fitting of an additive tree ensemble. Our algorithm in each iteration completes two operations that resemble those in supervised boosting: (i) computing the current residuals by subtracting the fitted measure at the current iteration from the observations and (ii) fitting a tree-based weak learner on the residuals and adding the estimated distribution to the current fit. The algorithm enables straightforward analytical evaluation of the probability density of the fitted distribution and produces a generative model for the fitted measure that can be directly sampled from.

We investigate the theoretical and methodological properties of our unsupervised boosting framework, many of which draw an interesting analogy to those of supervised boosting. First, we show that the unsupervised boosting algorithm can be justified from a decision-theoretic perspective in terms of sequentially reducing the Kullback-Leibler divergence between the model and the unknown true measure. In addition, just as in supervised learning [Breiman, 2004], the unsupervised boosting is “highly expressive” in the sense that a wide class of distributions can be represented or well approximated by a finite combination of highly constrained (or “weak”) tree-based density models. Moreover, we show that helpful considerations in applying boosting in supervised problems—including choosing the number of trees and setting the appropriate level of shrinkage/regularization—can be accomplished in the unsupervised counterpart and can have a substantial impact on the performance. We introduce a novel scale-dependent shrinkage strategy, which utilizes a scale-specific learning rate specified in terms of the size of the corresponding nodes in the tree-based weak learner. We also incorporate a two-stage approach that fits the marginal distributions and the copula of the unknown measure separately which substantially improves the model fit. Lastly, we present a measure of variable importance based on the respective contribution of each dimension in reducing in the overall KL divergence, which provides additional insights on the relevance of each dimension in
characterizing the underlying distribution and allows effective variable screening.

It is worth noting that boosting for unsupervised learning has been previously considered by Rosset and Segal (2002) under a different approach, through constructing an ensemble in the form of weighted average along with a gradient boosting algorithm (Mason et al., 1999) for minimizing a large range of loss functions. They successfully demonstrated the approach on discrete data using Bayesian networks as the weak learner. While the framework itself is general and in principle applicable to other types of weak learners and continuous data, implementing the strategy in practice for tree-based weak learners, incurs computational difficulties. We believe this is the main reason why there is yet to exist a widely used unsupervised tree boosting algorithm based on their strategy.

While we are motivated from the boosting perspective, our method can also be viewed as a so-called normalizing flow (NF), a framework for density estimation recently popularized in the machine learning community, often used in conjunction with deep neural networks (see i.e., Papamakarios (2019), Papamakarios et al. (2021), and Kobyzev et al. (2020)). In fact, the vanilla version of our algorithm, without several boosting-based performance enhancing specifications, can be shown to be algorithmically equivalent to a type of tree-based NF approach called “tree density destructors” proposed in Inouye and Ravikumar (2018). Nevertheless, as we will show, formulating the algorithm from the boosting perspective provides fresh theoretical insights as well as critical practical guidance for achieving competitive performance. We will discuss in more detail the connection and compare with several state-of-the-art deep learning-based NF methods using multiple benchmark datasets.

All technical details are provided in Section 3.5.
3.2 Method

3.2.1 CDF-based Addition and Residualization for Univariate Distributions

We start by defining notions of addition and residuals in the one-dimensional setting before generalizing them to multivariate distributions. Without loss of generality, let $(0, 1]$ represent the one-dimensional sample space. For ease of exploration, we shall assume that the distributions are absolutely continuous on the sample space with respect to the Lebesgue measure and have full support.

We first make an observation that if a random variable $X \sim G$, its sampling distribution, then $G(X) \sim \text{Unif}(0, 1]$, where $G$ denotes the CDF of $G$. (Throughout we will use bold font letters to indicate the CDFs of the corresponding distributions.) As such, the CDF transform “removes” the distributional structure of $G$ from the sampling distribution of $X$. Thus one can think of $r = G(X)$ as a “residual”. Moreover, Unif$(0, 1]$ serves as the notion of “zero”, whose CDF is the identity map, in the space of probability distributions as it is the remaining distribution after “subtracting” the true sampling distribution $G$ from $X$.

Next we define a notion of “adding” two distributions $G$ and $H$ that is consistent with the above notion of “subtraction” or “residualization”. Specifically, the addition of $G_1$ and $G_2$, denoted as “$G_1 \oplus G_2$”, should satisfy the property that if $X \sim G_1 \oplus G_2$, then $r^{(1)} = G_1(X) \sim G_2$. In other words, if the sampling distribution of $X$ is the “sum” of $G_1$ and $G_2$, then taking the “residual” of $X$ with respect to $G_1$ should result in a random variable distributed as $G_2$. Such a notion of addition indeed exists:

$$G_1 \oplus G_2 \text{ is the distribution whose CDF is } G_2 \circ G_1$$

where “$\circ$” denotes function composition. Note that this notion of addition is not commutative. That is, $G_1 \oplus G_2 \neq G_2 \oplus G_1$. Fortunately, as we will see, the operation of fitting an additive ensemble as in supervised boosting requires only a non-abelian
group-structure, which does not require the commutativeness of the underlying addition. As such, the loss of commutativeness will pose no difficulty in our construction of additive tree models and later an unsupervised boosting algorithm based on the new notions of addition and residuals.

By iteratively applying such an addition, one can define the “sum” of \( k \) (\( \geq 1 \)) probability measures \( G_1, \ldots, G_k \). Specifically, the sum of \( G_1, \ldots, G_k \),

\[
G_1 \oplus \cdots \oplus G_k \text{ is the distribution whose CDF is } G_k \circ \cdots \circ G_1.
\]

The following property provides the basis for sequential addition and residualization, analogous to those in supervised boosting.

**Proposition 3.2.1.** If \( G_1, \ldots, G_k \) have full support on \((0,1]\), (i.e., when the CDFs \( G_1, \ldots, G_k \) are strictly increasing,) then for any \( i = 1, 2, \ldots, k - 1 \)

\[
X \sim G_1 \oplus \cdots \oplus G_k \text{ if and only if } r^{(i)} = G_i \circ \cdots \circ G_1(X) \sim G_{i+1} \oplus \cdots \oplus G_k.
\]

Proposition 3.2.1 then implies that such residualization can be applied sequentially. That is, if \( r^{(k)} \) is the residual of \( x \) after subtracting \( G_1 \oplus \cdots \oplus G_k \), then \( r^{(0)} = x \), and for \( k \geq 1 \)

\[
r^{(k)} = G_k(r^{(k-1)}) = G_k \circ G_{k-1}(r^{(k-2)}) \cdots = G_k \circ G_{k-1} \circ \cdots \circ G_1(r^{(0)}).
\] (3.1)

The “additivity” induced by the composition of CDFs also induces an additivity on the corresponding log-likelihood. Specifically, suppose \( g_i = dG_i/d\mu \) is the probability density function (pdf) of \( G_i \) for \( i = 1, 2, \ldots, k \) with respect to Lebesgue measure \( \mu \).

Then the density \( f_k = dF_k/d\mu \) satisfies

\[
f_k(x) = \prod_{i=1}^{k} g_i(r^{(i-1)}) \quad \text{or} \quad \log f_k(x) = \sum_{i=1}^{k} \log g_i(r^{(i-1)}).
\]
Table 3.1: **Key concepts in boosting**

<table>
<thead>
<tr>
<th></th>
<th>Boosting for regression</th>
<th>Boosting for probability measures</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Addition</strong></td>
<td>$h_1 + \cdots + h_k$</td>
<td>$G_1 \oplus \cdots \oplus G_k$</td>
</tr>
<tr>
<td><strong>Residual</strong></td>
<td>$y - \sum_{t=1}^{k-1} h_t(x)$</td>
<td>$G_{k-1} \circ \cdots \circ G_1(x)$</td>
</tr>
<tr>
<td><strong>Zero</strong></td>
<td>0</td>
<td>Uniform distribution on $(0, 1]^d$</td>
</tr>
</tbody>
</table>

Table 3.1 summarizes the corresponding notions of addition, residuals, and zero in supervised boosting (in particular regression) and those in our unsupervised formulation. With these new notions, we are ready to introduce a boosting algorithm for learning one-dimensional distributions. However, the more interesting application involves multivariate (in fact high-dimensional) distributions. As such, we first generalize these notions to multivariate cases, and then introduce a multivariate version of our boosting algorithm that contains the (less interesting) univariate scenario as a special case.

### 3.2.2 Generalization to Multivariate Distributions

The above notions of addition and residuals do not find direct counterparts for multivariate measures if one uses the traditional definition of CDFs for multivariate distributions. In particular, because the traditional CDF is a mapping from $(0, 1]^d$ to $(0, 1]$ instead of $(0, 1]^d$, we cannot even take the composition of the CDFs or compute the residuals, which should remain in the same space as the original observations. Beyond the minimal requirement that the appropriate notion of “CDF” should map from $(0, 1]^d$ to $(0, 1]^d$, it must also enjoy several group-like properties of univariate CDF’s.

We summarize four such properties that the “CDF” must satisfy to allow the definition of addition and residualization to carry over into the multivariate setting:

(C1) $G$ is a mapping from $(0, 1]^d$ to $(0, 1]^d$.

(C2) $G$ is uniquely determined by $G$.
(C3) If \( X \sim G \), then \( G(X) \sim \text{Unif}((0,1]^d) \), the “zero”.

(C4) If \( X \sim G_1 \oplus G_2 \), the distribution uniquely determined by its “CDF” \( G_2 \circ G_1 \), then \( G_1(X) \sim G_2 \).

Remark: (C1) and (C2) are needed for defining addition in terms of compositions. (C3) and (C4) are needed for the proper notion of residuals.

For the purpose of constructing a tree additive ensemble model and a boosting algorithm, one type of “CDFs” that satisfy these conditions are particularly useful as they are very easy to compute for probability distributions with piecewise constant densities defined on leafs of a recursive dyadic partitioning of the sample space. As one can imagine, efficient computation of the “CDFs” for tree-based models is critical as they will be computed many times during the fit to an additive ensemble.

**Characterizing probability measures on a recursive dyadic partition tree**

Next we describe the construction of this generalized notion of multivariate CDFs, which we call the “tree-CDF”, due to its connection to recursive bifurcating partition trees. We start by introducing some additional notation related to recursive dyadic partitions.

A recursive dyadic partition of depth \( R \) is a sequence of nested dyadic partitions \( \mathcal{A}^1, \mathcal{A}^2, \ldots, \mathcal{A}^R \) on the sample space \( \Omega \). The first partition \( \mathcal{A}^1 \) only includes \( \Omega \), and for \( k = 2, \ldots, R \), the partition \( \mathcal{A}^k \) consists of all the sets generated by dividing each \( A \in \mathcal{A}^{k-1} \) into two children \( A_l \) and \( A_r \), where \( A_l \cup A_r = A \) and \( A_l \cap A_r = \emptyset \). (Throughout, we use subscripts \( l \) and \( r \) to indicate left and right children respectively.) We can denote the recursive partition using a tree \( T = \bigcup_{k=1}^{R} \mathcal{A}^k \). As such, we refer to the sets in the partitions as “nodes”. We call the collection of nodes in \( \mathcal{A}^R \) the “terminal” nodes or “leafs” of \( T \) and denote it by \( \mathcal{L}(T) \); the nodes in other levels are the “non-leaf” nodes or “interior” nodes, which we denote by \( \mathcal{N}(T) = T \setminus \mathcal{L}(T) \).
We consider partition trees with axis-aligned partition lines. In this case, a node $A \in T$ is of the following rectangular form:

$$A = (a_1, b_1] \times \cdots \times (a_d, b_d). \quad (3.2)$$

For a non-leaf node $A \in \mathcal{N}(T)$, the children $A_l$ and $A_r$ are generated by dividing $A$ in one of the $d$ dimensions, say $j^*$,

$$A_l = (a_1, b_1] \times \cdots \times (a_{j^*}, c_{j^*}] \times \cdots (a_d, b_d] \quad \text{and} \quad A_r = (a_1, b_1] \times \cdots (c_{j^*}, b_{j^*}] \times \cdots (a_d, b_d]. \quad (3.3)$$

In the following, for each partition tree $T$, we let $\mathcal{P}_T$ be the class of probability measures that are conditionally uniform on the leafs of $T$ and have full support on $\Omega$. That is,

$$\mathcal{P}_T = \{G : G \text{ has full support on } (0, 1]^d \text{ and } G(\cdot\mid A) = \mu(\cdot\mid A) \text{ for every } A \in \mathcal{L}(T)\},$$

where $\mu$ is the uniform distribution, and $G(\cdot\mid A)$ and $\mu(\cdot\mid A)$ are the corresponding conditional distributions on $A$.

![Figure 3.1: Visualization of the tree-based decomposition of a univariate CDF into three local moves. The dotted lines indicate $G(A_l \mid A)/\mu(A_l \mid A)$ or $G(A_r \mid A)/\mu(A_r \mid A)$ on each node $A$.](image)

To generalize the CDF transform from univariate to multivariate cases, first we note an interesting multi-scale decomposition of the univariate CDF—a univariate
CDF transform $\mathcal{G}$ for any distribution $G \in \mathcal{P}_T$ on an observation $x$ can actually be computed sequentially in a fine-to-coarse fashion along the branch in the partition tree $T$ in which $x$ falls. Specifically, suppose $T$ has depth $R$, then for $x \in (0, 1]$, let $\{A_k\}_{k=1}^R$ be a sequence of nodes in $T$ such that $A_k \in \mathcal{A}^k$ and

$$x \in A_R \subset A_{R-1} \subset \cdots \subset A_1 = (0, 1].$$

The CDF transform $G(x)$ can be decomposed into the composition of a sequence of “local move functions”:

$$G(x) = \mathcal{G}_{A_1} \circ \cdots \circ \mathcal{G}_{A_{R-1}}(x),$$  \hspace{1cm} (3.4)

where for any $A = (a, b) \in \mathcal{N}(T)$ with two children $A_l = (a, c]$ and $A_r = (c, b]$, the mapping $\mathcal{G}_A : A \rightarrow A$ is (up to a normalizing constant $\mu(A)$) the CDF of a dyadic piecewise constant density equal to $G(A_l|A)/\mu(A_l)$ on $A_l$ and $G(A_r|A)/\mu(A_r)$ on $A_r$. More precisely, $\mathcal{G}_A : A \rightarrow A$ is given by

$$\frac{\mathcal{G}_A(x) - a}{x - a} = \frac{G(A_l|A)}{\mu(A_l|A)} \text{ for } x \in A_l \quad \text{and} \quad \frac{b - \mathcal{G}_A(x)}{b - x} = \frac{G(A_r|A)}{\mu(A_r|A)} \text{ for } x \in A_r.$$  

Note that the conditional measures and the input and output of $\mathcal{G}_A$ have the following relationship:

$$G(A_l|A) > \mu(A_l|A) \iff G(A_r|A) < \mu(A_r|A) \iff \mathcal{G}_A(x) > x,$$

$$G(A_l|A) < \mu(A_l|A) \iff G(A_r|A) > \mu(A_r|A) \iff \mathcal{G}_A(x) < x.$$  

We call $\mathcal{G}_A$ a “local move” function because it moves a point in $A$ in the direction of the child node with less (conditional) probability mass than the (conditional) uniform measure as illustrated in Figure 3.1. The amount of movement on $A$ is proportional to the probability mass differential between the two children of $A$ in $G$ relative to $\mu$.

If we think of applying the univariate CDF transform as “subtracting” the information contained in a probability measure from an observation, the decomposition in Eq. (3.4) indicates that such subtraction can be done sequentially through the
local moves, each subtracting a piece of information regarding the measure from the
observation. This perspective leads to a generalization of the CDF transform for the
multivariate case as we describe below.

For a point \( x = (x_1, \ldots, x_d) \in \Omega = [0, 1)^d \), again let \( T \) be a recursive dyadic
partition tree of depth \( R \) on the sample space \( \Omega \), and \( \{A_k\}_{k=1}^R \) the sequence of nodes
in \( T \) that contains \( x \) as before. Then we define a mapping \( \mathcal{G} : (0,1]^d \to (0,1]^d \) in
terms of a sequence of fine-to-coarse local moves along that branch in \( T \). Specifically,
for a node \( A \in \mathcal{N}(T) \) as in Eq. (3.2) with children \( A_l \) and \( A_r \) attained from dividing
\( A \) in the \( j^* \)th dimension as described in Eq. (3.3), we define a local move mapping
\( \mathcal{G}_A : A \to A \) such that for any \( x \in A \), \( \mathcal{G}_A(x) = (\mathcal{G}_{A,1}(x), \ldots, \mathcal{G}_{A,d}(x)) \) where \( \mathcal{G}_{A,j}(x) = x \) for all \( j \neq j^* \), and
\[
\frac{\mathcal{G}_{A,j^*}(x) - a_{j^*}}{x_{j^*} - a_{j^*}} = \frac{G(A_l|A)}{\mu(A_l|A)} \quad \text{for} \quad x \in A_l \quad \text{and} \quad \frac{b_{j^*} - \mathcal{G}_{A,j^*}(x)}{b_{j^*} - x_{j^*}} = \frac{G(A_r|A)}{\mu(A_r|A)} \quad \text{for} \quad x \in A_r.
\]

As illustrated in Figure 3.2, similar to the univariate case, the local move mapping
is nothing but (up to a normalizing constant \( \mu(A) \)) the CDF of a dyadic piecewise
constant density on \( A \), except that now in the multivariate setting there are a total
of \( d \) directions in which such a dyadic split can take place. As a transform, it moves
\( x \) in the direction of the child node with less probability mass relative to the uniform
measure.

As before, we now define a mapping \( \mathcal{G} : (0,1]^d \to (0,1]^d \), called a “tree-CDF”, as
the composition of these local move functions. That is,
\[
\mathcal{G}(x) = \mathcal{G}_{A_1} \circ \cdots \circ \mathcal{G}_{A_{R-1}}(x).
\]
\( \mathcal{G} \) is injective from \((0,1]^d\) to \((0,1]^d\) for any \( \mathcal{G} \) with full support on \((0,1]^d\). Addition-
ally, because \( \mathcal{G}_A \) is surjective for every \( A \), \( \mathcal{G} \) is also surjective. One can also show
that \( \mathcal{G} \) is measurable. Hence we have the following proposition that establishes Con-
ditions (C1) for tree-CDFs, which is essential to defining the addition of multivariate
distributions in terms of tree-CDF compositions.
Figure 3.2: Top: Visualization of the local move functions in each level under $R = 3$. The nodes with the darker color have higher conditional probabilities relative to the uniform measure. Bottom: An example of $G_A$ with $j^* = 2$. The input and the output of $G_A$ are indicated by $\times$ and $\circ$, respectively.

Proposition 3.2.2. The tree-CDF mapping $G : (0, 1]^d \mapsto (0, 1]^d$ is bijective and measurable for any $G \in \mathcal{P}_T$.

The next two theorems show that our construction of $G$ satisfies Conditions (C2) and (C3) as well. That is, $G$ uniquely determines $G$ and applying the $G$ mapping to an observation effectively “subtracts” the distributional structure in $G$ from the sampling distribution of that observation.

Theorem 3.2.1. A measure $G \in \mathcal{P}_T$ for some partition tree $T$ can be determined by the tree-CDF mapping $G$ as follows

$$G(B) = \mu(\{G(x) : x \in B\}) \quad \text{for all } B \in \mathcal{B}(\Omega).$$

Remark: Theorem 3.2.1 establishes (C2) and implies that while the mapping $G$ for $G$ is not unique but tree-specific, $G$ uniquely determines $G$, regardless of the tree from which $G$ is defined.
Theorem 3.2.2. If \( X \sim G \in \mathcal{P}_T \), then \( G(X) \sim \text{Unif}((0,1]^d) \). Conversely, if \( U \sim \text{Unif}((0,1]^d) \), then \( G^{-1}(U) \sim G \).

Remark: Theorem 3.2.2 establishes (C3) and shows that if one can compute the inverse map \( G^{-1} \) then one essentially has a generative model, which allows generating samples from \( G \) based on “inverse-CDF” sampling. More details on this will be given in Section 3.2.3.

Addition and residualization for multivariate settings

Let \( G_1, \ldots, G_k \) be a collection of probability measures such that \( G_l \in \mathcal{P}_{T_l} \) for \( l = 1, 2, \ldots, k \), and let \( \mathcal{G}_1, \ldots, \mathcal{G}_k \) be the corresponding tree-CDFs. As a generalization to the univariate case, next we define addition of distributions by composing their tree-CDFs. We first show that such a composition indeed pins down a unique probability measure.

Lemma 3.2.1. For \( G_l \in \mathcal{P}_{T_l} \) \( (l = 1, 2, \ldots, k) \), the mapping \( F_k : \mathcal{B}(\Omega) \mapsto (0,1] \) defined as

\[
F_k(B) = \mu(\{G_k \circ \cdots \circ G_1(x) : x \in B\}) \quad \text{for } B \in \mathcal{B}(\Omega).
\]  

(3.5)

is a probability measure.

Now we can define the sum of \( k \) distributions, \( G_1 \oplus \cdots \oplus G_k \), as the measure \( F_k \) given in Eq. (3.5). This definition of addition contains the univariate case presented earlier as a special case. We note that the addition implicitly involves the tree structures \( T_1, \ldots, T_k \). This dependency on the trees, however, is suppressed in the “\( \oplus \)” notation for simplicity without causing confusion.

Next we turn to the notion of residuals and generalize Proposition 3.2.1 to multivariate distributions, which establishes Condition (C4) for tree-CDFs.
Proposition 3.2.3. Let $G_1, \ldots, G_k$ be a collection of probability measures such that $G_l \in \mathcal{P}_{T_l}$ for $l = 1, 2, \ldots, k$. Then for any $i = 1, 2, \ldots, k - 1$

$$X \sim G_1 \oplus \cdots \oplus G_k \text{ if and only if } r^{(i)} = G_i \circ \cdots \circ G_1(X) \sim G_{i+1} \oplus \cdots \oplus G_k.$$  

Remark: This proposition implies Condition (C4) by setting $k = 2$ and $i = 1$.

Moreover, the sequential update of the residuals given in Eq. (3.1) remains valid. The only difference is that now the residualization in each step depends on an implicit partition tree structure, encapsulated in the corresponding tree-CDF.

3.2.3 An Unsupervised Boosting Algorithm based on Forward-stagewise (FS) Fitting

Equipped with the new notions of addition and residuals, we are ready to generalize our unsupervised boosting algorithm to the multivariate setting based on forward-stagewise (FS) fitting. Suppose we have an i.i.d. sample $x_1, \ldots, x_n$ from an unknown distribution $F$, which we model as an additive ensemble of $K$ probability measures

$$F = G_1 \oplus \cdots \oplus G_K \quad (3.6)$$

where each $G_k$ is modeled as a member in $\mathcal{P}_{T_k}$ for some (unknown) $T_k$. We introduce an FS algorithm in which we compute the residuals step-by-step and at the $k$th step, fit $G_k$ to the current residuals. The fit at the $k$th step produces an estimate for $G_k$ along with a partition tree $T_k$, which is used to define the tree-CDF in the next step for computing the new residuals.

Initialization

Set $r^{(0)} = (x_1, \ldots, x_n)$ and set $G_0 = \mu$, the uniform distribution.

Forward-stagewise fitting

Repeat the following steps for $k = 1, \ldots, K$: 

83
1. Update the residuals $r^{(k)} = (r_1^{(k)}, \ldots, r_n^{(k)})$, where $r_i^{(k)} = G_k(r_i^{(k-1)})$.

2. Fit a weak learner that produces a pair of outputs $(G_k, T_k)$ to the residualized observations $r^{(k)}$, where $T_k$ is an inferred partition tree and $G_k$ is the tree-CDF for a measure $G_k \in \mathcal{P}_{T_k}$.

The output of the boosting algorithm in terms of the collection of pairs $(G_k, T_k)$ for $k = 1, 2, \ldots, K$ contains all of the information from the data regarding the underlying distribution. (In fact the $G_k$’s alone contain all the relevant information, but the $T_k$’s are indispensable for effectively representing and storing the $G_k$’s.)

Next we demonstrate two ways to extract such information. In particular, we show (i) how to compute the density function of the fitted measure $F$ at any point in the sample space analytically, and (ii) how to use the resulting generative model to draw Monte Carlo samples from the fitted measure $F$ based on “inverse-CDF” sampling.

**Evaluating the Density Function of $F$.** Density estimation is a common objective in learning multivariate distributions. The next proposition generalizes the additive decomposition of the log-likelihood for the univariate case and provides a recipe for evaluating the density for the fitted measure $F$ analytically based on the output of the FS algorithm.

**Proposition 3.2.4.** For any $x \in (0, 1]^d$, the density $f = dF/d\mu$ for $F$ of the additive form in Eq. (3.6) is given by

$$f(x) = \prod_{k=1}^K g_k(r^{(k-1)}) \quad \text{or} \quad \log f(x) = \sum_{k=1}^K \log g_k(r^{(k-1)})$$
where \( g_k = dG_k/d\mu \) is the density of \( G_k \), \( r^{(k-1)} = G_{k-1} \circ \cdots \circ G_1(x) \), is the residual for \( x \) after subtracting \( G_1 \oplus \cdots \oplus G_{k-1} \), and in particular \( r^{(0)} = x \). In other words, the density \( f(x) \) is exactly the product of the fitted density of each weak learner evaluated at the corresponding sequence of residuals.

**A generative model for \( F \).** It turns out that one can use the classical idea of inverse-CDF sampling to construct a generative model for \( F \) as a result of Theorem 3.2.2. Specifically, we can generate samples from \( F \) by first generating \( U \sim \text{Unif}((0, 1]^d) \) and then compute the following transform

\[
F^{-1}(U) := G_1^{-1} \circ \cdots \circ G_K^{-1}(U) \tag{3.7}
\]

where \( G_k^{-1} \) is the corresponding inverse for the tree-CDF \( G_k \) for \( k = 1, 2, \ldots, K \). To implement the sampler, we next obtain the analytic form of the inverse of a tree-CDF.

Recall that in Section 3.2.2 we showed that a tree-CDF \( G \) for a measure \( G \in \mathcal{P}_T \) can be expressed as the composition of a sequence of local move mappings \( G_A : A \to A \) along each subbranch of \( T \). The inverse of the local move mapping \( G_A \) can be expressed as \( G_A^{-1}(y) = (G_{A,1}^{-1}(y), \ldots, G_{A,d}^{-1}(y)) \) for any \( y = (y_1, \ldots, y_d) \in A \), where

\[
G_{A,j}^{-1}(y_j) = \begin{cases} y_j, & (j \neq j^*) \\ G_{A,j}^{-1} \left( \frac{y_j - a_j}{b_j - a_j} \right), & (j = j^*) \end{cases}
\]

and

\[
G_{A,j}^{-1}(z_j) = \begin{cases} a_j + \frac{c_j - a_j}{G(A_l \mid A)} z_j & \text{if } y_j \leq a_j + G(A_l \mid A)(b_j - a_j), \\ c_j + \frac{b_j - c_j}{G(A_r \mid A)} \{z_j - G(A_l \mid A)\} & \text{if } y_j > a_j + G(A_l \mid A)(b_j - a_j). \end{cases}
\]

With the inverse local move function \( G_A^{-1} \) available for all \( A \in \mathcal{N}(T) \), we can obtain the explicit form for the inverse tree-CDF \( G^{-1} \) as

\[
G^{-1}(y) = G^{(R-1)} \circ \cdots \circ G^{(1)}(y),
\]
where for \( k = 1, \ldots, R - 1 \),
\[
G^{(k)}(y) = \sum_{A \in \mathcal{A}^k} G^{-1}_A(y) 1_A(y).
\]

### 3.2.4 Decision-theoretic Considerations

In this subsection we show that our boosting algorithm can be interpreted as fitting the additive model in Eq. (3.6) by sequentially reducing the Kullback-Leibler divergence.

Let \( F^* \) be the true sampling distribution for the observations and \( f^* = dF^*/d\mu \) its density function. Again, let \( F = G_1 \oplus \cdots \oplus G_K \) be the additive model for the distribution and \( f = dF/d\mu \) its density. We consider the entropy loss, i.e., the Kullback-Leibler (KL) divergence between \( F^* \) and \( F \)
\[
KL(F^*||F) = \int \log \frac{f^*}{f} dF^*.
\]  
(3.8)

The next lemma states that the entropy loss in Eq. (3.8) can be decomposed into \( K \) components (ignoring a constant) each of which only depends on \( G_k \).

**Lemma 3.2.2.** The Kullback-Leibler divergence can be written as
\[
KL(F^*||F) = \int \log \frac{f^*}{f} dF^* - \sum_{k=1}^{K} \{KL(\tilde{F}_k||\mu) - KL(\tilde{F}_k||G_k)\}. \tag{3.9}
\]

where \( \tilde{F}_k \) is the true distribution of the residualized observation after subtracting \( G_1, G_2, \ldots, G_{k-1} \). That is, \( \tilde{F}_k \) is the true distribution of \( r^{(k-1)} = G_{k-1} \circ \cdots \circ G_1(X) \), where \( X \sim F^* \).

Remark: Note that because \( X \sim F^* \), we have \( \tilde{F}_1 = F^* \), and for \( k = 2, \ldots, K \), \( \tilde{F}_k \) is given by
\[
\tilde{F}_k(B) = F^*(G_1^{-1} \circ \cdots \circ G_{k-1}^{-1}(B)) \quad \text{for all } B \in \mathcal{B}((0,1]^d).
\]

The first term on the right-hand side of Eq. (3.9) is a constant. The summand in the second term is positive as long as the measure \( G_k \) is closer to \( \tilde{F}_k \) than the
uniform measure \( \mu \) in terms of KL divergence. Hence, unless \( \tilde{F}_k = \mu \), the entropy loss could be reduced by adding an additional measure \( G_k \) that is closer to \( \tilde{F}_k \) than \( \mu \). In this way, fitting a measure \( G_k \) to the residuals \( r^{(k)} \) in the \( k \)th step of our boosting algorithm can be understood as an operation to sequentially reduce the KL divergence. Next we turn from the above insight at the population level to the practical strategy at the finite-sample level for fitting \( F \) based on \( n \) i.i.d. observations \( \{x_i\}_{i=1}^n \) from \( F^* \). First note that minimizing the divergence \( \text{KL}(F^*||F) \) is equivalent to maximizing the average log-density \( \int \log f dF^* \). Thus with a finite sample, we aim to maximize the sample (average) log-density of the training data

\[
\frac{1}{n} \sum_{i=1}^{n} \log f(x_i).
\]

It follows from Proposition 3.2.2 that the log-density can also be decomposed into the sum of \( K \) components, which we call “improvements”.

**Lemma 3.2.3.** The sample average log-density can be written as

\[
\frac{1}{n} \sum_{i=1}^{n} \log f(x_i) = \sum_{k=1}^{K} D_k^{(n)}(G_k),
\]

where for \( k = 1, 2, \ldots, K \), the improvement \( D_k^{(n)}(G_k) \) is

\[
D_k^{(n)}(G_k) = \frac{1}{n} \sum_{i=1}^{n} \log g_k(r_i^{(k-1)}) \quad \text{with} \quad r_i^{(k-1)} = G_{k-1} \circ \cdots \circ G_1(x_i).
\]

Accordingly, the next proposition characterizes the “optimal” pair \((G_k, T_k)\) that maximizes \( D_k^{(n)}(G_k) \).

**Proposition 3.2.5.** A pair of \((G_k, T_k)\) maximizes \( D_k^{(n)}(G_k) \) if and only if

\[
T_k \in \arg \max_{T \in \mathcal{T}} \sum_{A \in \mathcal{L}(T)} \tilde{F}_k^{(n)}(A) \log \frac{\tilde{F}_k^{(n)}(A)}{\mu(A)},
\]

(3.10)
and

\[ G_k(A) = \tilde{F}_k^{(n)}(A) \quad \text{for all } A \in \mathcal{L}(T_k), \]  

(3.11)

where \( \tilde{F}_k^{(n)} \) is the empirical measure of the residuals \( r^{(k-1)}_i = \{r^{(k-1)}_i\}_{i=1}^n \). That is,

\[ \tilde{F}_k^{(n)}(B) = \frac{1}{n} \sum_{i=1}^{n} \delta_B(r^{(k-1)}_i) \quad \text{for } B \in \mathcal{B}((0,1]^d). \]

Remark: The summation in Eq. (3.10) is the KL divergence between two discrete probability measures with masses given by \( \{\tilde{F}_k^{(n)}(A)\}_{A \in \mathcal{L}(T)} \) and \( \{\mu(A)\}_{A \in \mathcal{L}(T)} \) respectively. Eq. (3.10) implies that the “optimal” tree \( T_k \) should allow maximal differentiation in KL divergence between the induced discretizations of \( \tilde{F}^{(n)} \) and \( \mu \) on its leaves.

3.2.5 Practical Considerations

In this subsection we describe several practical considerations in implementing and applying the boosting algorithm. While they might first appear as technical details, we have found that they are critical in achieving competitive performance and thus worth elaborating on. Several of these considerations are drawn from similar considerations in supervised boosting.

Choice of a Weak Learner

Searching over all possible trees to solve Eq. (3.10) in each step of the FS algorithm is computationally prohibitive. Nevertheless, Proposition 3.2.5 provides hints on how to choose good weak learners that improve the KL divergence efficiently over the iterations. The simplest possible choice of a weak learner, as is often implemented in supervised boosting is to implement a top-down greedy tree learning algorithm that maximizes Eq. (3.10) one split at a time, as is done in fitting classification and regression trees (CART) (Hastie et al., 2009).
In our numerical examples and software, we adopt a weak learner based on a simplified version of an unsupervised (Bayesian) CART model for probability distributions proposed in Awaya and Ma (2021). Fitting this weak learner uses a stochastic one-step look-ahead strategy to choose splitting decisions on each tree node, which generally produces closer approximation to the “optimal” tree splits than greedy tree algorithms. (See Theorem 4.1 in Awaya and Ma (2021) for an asymptotic justification—as the sample size grows, it produces trees that satisfy Proposition 3.2.5 with probability increasing to 1.) Additional details about the weak learner can be found in the Section 3.5.6.

It is worth emphasizing that because we are only building “weak” learners that extract a small fraction of the distributional structure in each iteration, one does not need to be precisely “optimal” in each iteration. More importantly than being “optimal”, the weak learner should facilitate the appropriate shrinkage to avoid overfitting, which we elaborate in the next subsection.

**Regularization through Scale-specific Shrinkage**

Just as in supervised boosting, simply adopting the solution for Eq. (3.10) (either exact or approximate) as the fit for $G_k$ in each iteration will typically lead to overfitting even when the complexity of the tree $T_k$ is restricted to be small. In particular, the fitted density will tend to have spikes at or near the training points. To avoid such overfitting, it is necessary to regularize or penalize the non-smoothness in the fit for each $G_k$. This can be achieved through shrinkage toward “zero”, or the uniform measure $\mu$, thereby discounting the influence of the residuals (or its empirical measure $\tilde{F}_k^{(n)}$) on fitting $G_k$. In supervised boosting it is typical to introduce a learning rate $c_0 \in (0, 1]$ that controls how much shrinkage toward zero is applied in each iteration. In the current context, this traditional strategy would correspond to setting

$$G_k = (1 - c_0)\mu + c_0 \tilde{F}_k^{(n)}.$$
We found that in practice one can further improve upon this shrinkage strategy by allowing different levels of shrinkage at different scales. The intuition is that depending on the smoothness of the underlying function, overfitting can be more (or less) likely to happen in learning local details of the distribution and thus one may benefit from enforcing a level of shrinkage that increases (or not) with the depth in the tree $T_k$. Following this intuition, we specify a scale-dependent learning rate

$$c(A) = c_0 \cdot (1 + \log_2 \text{vol}(A))^{-\gamma},$$

where $A$ is a node in $T_k$, $\text{vol}(A)$ is a volume of $A$. Then the shrinkage toward the uniform can be specified on each node $A \in \mathcal{N}(T_k)$ in terms of the conditional probability on the children of $A$

$$G_k(A_l | A) = (1 - c(A))\mu(A_l | A) + c(A)\tilde{F}^{(n)}_k(A_l | A) \text{ for } A \in \mathcal{N}(T_k),$$

$$G_k(\cdot | A) = \mu(\cdot | A) \text{ for } A \in \mathcal{L}(T_k),$$

where $A_l$ and $A_r$ are the children nodes of $A$ in $T_k$, $\tilde{F}^{(n)}_k(A_l | A) = \tilde{F}^{(n)}_k(A_l) / \tilde{F}^{(n)}_k(A)$ if $\tilde{F}^{(n)}_k(A) > 0$ and $\tilde{F}^{(n)}_k(A_l | A) = \mu(A_l | A)$ otherwise.

The node-specific learning rate $c(A)$ controls how strongly one “pulls” the empirical measure $\tilde{F}^{(n)}$ toward the uniform measure $\mu$ at the corresponding scale of $A$. It is specified with two tuning parameters $c_0 \in (0, 1]$ and $\gamma \geq 0$. The parameter $c_0$ controls the global level of shrinkage, and when $\gamma > 0$ we introduce stronger shrinkage for small nodes, imposing stronger penalty on local spikes. When $\gamma = 0$, this shrinkage reduces to the standard single learning rate specification described above.

In practice, we recommend setting these tuning parameters by cross-validation.

Our next proposition shows that with shrinkage, the sample average log-density is steadily improved in each step of the FS algorithm until the residual distribution becomes the uniform measure.
Proposition 3.2.6. For any finite tree $T_k$, under the definition of $G_k$ given in Eq. (3.12), the improvement satisfies $D_k^{(n)}(G_k) \geq 0$ if $c(A) \in (0, 1]$ for all $A \in \mathcal{L}(T_k)$ unless $\tilde{F}_k^{(n)}(A)$ is indistinguishable from the uniform distribution on the tree $T_k$, that is, $\tilde{F}_k^{(n)}(A) = \mu(A)$ for all $A \in \mathcal{L}(T_k)$.

Evaluating Variable Importance

As in supervised learning, it is often desirable to evaluate the contribution of each dimension to the approximation of the unknown measure $F^*$. Thus we provide a way to quantify variable importance in a conceptually similar manner to what is often used in supervised boosting (see Hastie et al. (2009)). We note that Ram and Gray (2011) also introduced a notion of the variable importance in density trees. While their definition is based on improvement in the $L_2$ loss, ours is based on the KL divergence, which is consistent with our earlier decision-theoretic discussion.

Specifically, because our boosting algorithm reduces the KL divergence from the unknown measure $F^*$, a natural way of quantifying the importance of a variable is adding up the decrease in the KL divergence due to splitting a tree node in the corresponding dimension. Lemma 3.2.3 shows that this quantity can be expressed as the sum of the improvements $D_k^{(n)}(G_k)$. In particular, the improvement $D_k^{(n)}(G_k)$ can be further decomposed over the splits of the tree $T_k$ as follows

$$D_k^{(n)}(G_k) = \sum_{A \in \mathcal{N}(T)} \tilde{F}_k^{(n)}(A) \left\{ \tilde{F}_k^{(n)}(A_l | A) \log \frac{G_k(A_l | A)}{\mu(A_l | A)} + \tilde{F}_k^{(n)}(A_r | A) \log \frac{G_k(A_r | A)}{\mu(A_r | A)} \right\},$$

where the empirical measure $\tilde{F}_k^{(n)}$ is as defined in Proposition 3.2.5. Note that the summation inside of the brackets can be written as

$$\text{KL}(\tilde{F}_k^{(n)}(A_l | A)||\mu(A_l | A)) - \text{KL}(\tilde{F}_k^{(n)}(A_l | A)||G_k(A_l | A)),$$

where $\text{KL}(p||q) = p \log(p/q) + (1-p) \log[(1-p)/(1-q)]$, and it quantifies the extent to which splitting $A$ makes $G_k$ closer to the distribution of the residuals. Based on
the decomposition, a natural definition of the total contribution of dividing in the
\( j \)th dimension is

\[
I_{Gk,j} = \sum_{A \in N_j(T)} \left\{ \tilde{F}_k^{(n)}(A | A) \log \frac{G_k(A_l | A)}{\mu(A_l | A)} + \tilde{F}_k^{(n)}(A_r | A) \log \frac{G_k(A_r | A)}{\mu(A_r | A)} \right\}
\]

where \( N_j(T) \) represents the collection of all nodes in \( T \) that are split in the \( j \)th dimension. Finally we can define the importance of the \( j \)th variable in the additive
measure \( F = G_1 \oplus \cdots \oplus G_K \) by summing over the variable importance across the
\( G_k \)'s,

\[
I_j = \sum_{k=1}^{K} I_{Gk,j}.
\]

**Fitting the Margins and the Copula Separately and Addressing Technical Ties**

In the density estimation literature, [Lu et al. (2013)](#) suggested a two-stage strategy for estimating multivariate densities using tree-based models, which separately fits the marginal distributions and then the dependence (or copula). From our experience, this strategy can often substantially improve the fit of our unsupervised boosting algorithm.

This two-stage strategy is easy to realize in our algorithm. In the first stage, for each of the dimensions, one can adopt weak learners that are constrained to involving tree-CDFs based on partitions along that single dimension. Computing the residuals with tree-CDFs defined on such a tree only removes the marginal distributions from the observations. “Subtracting” all of the marginal distributions from the original observations results in a sample of residuals the remaining distribution with uniform marginals (i.e., the corresponding copula). Then in the second stage, the single-
dimension constraint on the partition trees is removed, and tree-CDFs are then fitted.
to the copula. The final fit is simply the sum, in terms of tree-CDF compositions, of all of the marginals and the copula.

A related practical consideration regards tied values in the training data. (Ties in the margins occur much more frequently than ties that occur simultaneously in all margins and thus the issue is particularly relevant during the fitting of the marginal distributions in this two-stage strategy.) When tied values occur, either from the actual data generative mechanism or due to technical reasons such as rounding, the additive tree model itself will only assume that it is due to the actual data generative mechanism and therefore there must be positive probability mass at those tied values, leading to spikes of estimated densities at those values. In practice, if the data generative mechanism is assumed to be continuous and the ties are due to technical reasons such as rounding, one can avoid this issue by a simple preprocessing step for the training data that “smooths out” those spikes by adding small perturbations before fitting the model. We have found a simple strategy to be effective—when there are ties in the training data at the same value \( x \) and the adjacent values are \( x_- \) and \( x_+ \) (\( x_- < x < x_+ \)), we add uniform perturbation to the training data at \( x \) on the support \((-(x - x_-)/2, (x_+ - x)/2)\).

3.2.6 Expressive Power of Additive Tree Ensembles

One interesting question is what kind of probability measures can be well approximated by the ensemble when relatively simple (e.g., shallow) tree-based weak learners are combined. This is often referred to as the “expressive power” of the model in the machine learning literature, or the “support” of the model in the statistical literature. The expressive power for several normalizing flows have been analyzed (Huang et al., 2018; Jaini et al., 2019; Kong and Chaudhuri, 2020), and the large support property of linear combinations of classification trees has been established in Breiman (2004). We show next that a similar property holds for our unsupervised tree ensemble under
the following set of conditions.

**Assumption 3.** For $k = 1, 2, ..., K$, the pair of the tree $T_k$ and the measure $G_k$ that forms a component of the tree ensemble satisfies the following conditions:

1. The tree $T_k$ can be any finite tree formed by the dyadic splitting rule described in Eq. (3.2) and Eq. (3.3). That is, it incorporates an axis-aligned splitting rule with flexible split points.

2. Each $T_k$ has at least $d + 1$ leaf nodes, where $d$ is the dimension of the sample space.

3. The measure $G_k$ can be any conditionally uniform measure on $T_k$, namely, for every non-terminal node $A \in \mathcal{N}(T_k)$, the conditional probability $G_k(A_l | A)$ can be any value in $(0, 1)$.

Most notable in the assumption is the second condition, which is in sharp contrast to theories on single tree-based density models in the statistical literature. For instance, a popular tree-based density model, the Pólya tree (PT) model is shown to have the large support under the assumption that a single tree has infinite depth (Lavine, 1992; Ghosal and Van der Vaart, 2017). This is not surprising in the context of additive trees, however, since Proposition 2 of Breiman (2004) for supervised boosting essentially requires the same condition. With additive ensembles, we can combine small trees to express general continuous distributions, as formally stated in the next theorem.

**Theorem 3.2.3.** Let $F^*$ be a probability measure that has a bounded density function. Then, under Assumption 3 for any $\epsilon > 0$, there exists a tree ensemble with a finite number of tree measures, $G_1 \oplus \cdots \oplus G_K$, that approximates $F^*$ in terms of the KL divergence with this precision, i.e.,

\[
\text{KL}(F^* || G_1 \oplus \cdots \oplus G_K) < \epsilon.
\]
3.2.7 Connection to Normalizing Flows

Under our definition of additive tree ensembles, the unknown distribution of the observation is modeled as the transformation of the uniform distribution that takes the form

$$T_1 \circ \cdots \circ T_K(U), \ U \sim \text{Unif}([0,1]^d),$$

where $T_k = F_k^{-1}$. Given this expression, we can find a connection between our new boosting and the normalizing flow (NF) methods, a class of machine learning algorithms for density estimation. For comprehensive reviews of the NF, see Papamakarios (2019), Papamakarios et al. (2021), and Kobyzev et al. (2020). In NF methods, one approximates the observation’s distribution with the transformation of known distributions such as the uniform and the Gaussian, and the transformation is represented as a composition of multiple functions. From this viewpoint, our boosting method can be considered an NF method in which we use the inverse tree-CDFs $F_k^{-1}$ as base transformations.

Some of the NF algorithms are similar to our boosting algorithm in that they adopt an iterative approach, that is, sequentially transform the observations to make the distribution close to the known distributions, as we sequentially residualize the observations with $G_k$’s. In particular, Inouye and Ravikumar (2018) introduces an NF method called “density destructors”, which “substracts” information from i.i.d. samples until they are uniform samples. Notably, one particular type of density destructors is defined based on subtracting tree-based transforms, which can be shown to be equivalent to our notion of tree-CDF transform. With this connection found, our contribution can be seen as an elaboration of the method and theory under the boosting perspective.

For most NF models, there tends to be a trade-off between the ease in evaluating the fitted density or generating samples from the fitted distribution and that
in achieving large expressive power (Papamakarios et al., 2021). It is worth noting that density evaluation and simulation given the fitted additive model are both straightforward to implement under the boosting algorithm because these tasks only require transforming inputs with the tree-CDF and its inverse function respectively, both of which are available in closed forms. The computational cost of these tasks is $O(RK)$ and in practice, the cost is much smaller because the node splitting is often terminated in shallow levels on much of the sample space. At the same time, the proposed additive tree ensemble is capable of expressing or approximating general continuous distributions as shown in Section 3.2.6.

3.3 Numerical Experiments

In this section, we first carry out simulations in two-dimensional sample spaces for ease of visualization on the fitted distribution. In this context we demonstrate the effect of the tuning parameters on the performance of the unsupervised boosting algorithm for a few representative forms of the density. Then we provide a comparison in density estimation with several state-of-the-art normalizing flow methods on several popular benchmark datasets. Finally we demonstrate the computation of variable importance in the well-known MNIST handwritten digits data (LeCun et al., 1998). Except in the 2D cases, we adopt the two-stage strategy discussed in Section 3.2.5 and set the number of measures for the first stage (estimation of the marginal distributions) to 100 per dimension and 2,500 for the second stage (estimation of the dependence structures). The maximum depth of trees $R$ is 50.

3.3.1 Simulation Study in 2D Sample Spaces

Let us consider three two-dimensional scenarios whose true densities are illustrated in the first row of Figure 3.3. Presented in the second row in Figure 3.3 is a comparison of the KL divergence from the true distribution obtained by our boosting algorithm
under different values of $c_0$ with $\gamma$ fixed to 0.1 as the number of trees grow. We can see that the divergence is minimized when $c_0$ is small (0.1) for Scenario A and B, in which the true densities have smooth surfaces, and when $c_0$ is large (0.9) for Scenario C, in which the true density has axis-aligned discontinuities.

A sample of the estimated densities is visualized in Figure 3.4. In Scenario A, when the learning rate is high ($c_0 = 0.9$), the estimated distribution has very high densities in a small portion of the sample space due to overfitting. The estimated densities also have many clear discontinuities and this pixelation phenomenon is a result of combining tree measures without sufficient shrinkage. The densities are smoothed under stronger shrinkage, especially when $c_0 = 0.1$. We note that in many practical applications Scenarios A and B are more realistic than C, and in the case of higher-dimensional real data sets we use in Section 3.3.2 selecting small learning rates generally leads to desirable inference.

We also provide a comparison under different values of $\gamma$ with $c_0$ fixed to 0.1 and $K$ set to 200 in the third row of Figure 3.3. Though the values $\gamma$ seem less influential to the estimation accuracy compared to $c_0$, we can see the tendency that the KL divergence is improved when $\gamma$ is positive for Scenario A and when $\gamma$ is close to 0 for Scenarios B and C. Optimal values of $\gamma$ can vary also for real data sets as we see in Section 3.3.2. In general, we suggest choosing $c_0$ and $\gamma$ based on cross-validation.

3.3.2 Performance Comparison with State-of-the-art Density Estimators

We evaluate the performance of our boosting algorithm using seven popular benchmark data sets recorded in the University of California, Irvine (UCI) machine learning repository [Dua and Graff 2017]. We preprocessed the four data sets (“POWER”, “GAS”, “HEPMASS”, and “MINIBOONE”) following Papamakarios et al. (2017) with the code provided at https://github.com/gpapamak/maf and the three data sets (“AReM”, “CASP”, and “BANK”) with the code provided at https://zenodo.
Figure 3.3: The average KL divergence from the true distribution computed over 30 different data sets obtained under the different values of $c_0$ and $\gamma$ (the parameters of the learning rate). The first row shows the true density functions. The KL divergence is compared under different number of measures and values of $c_0$ in the second row, and the comparison under different values of $\gamma$ is shown in the third row. In the second row, the range of the second axis is limited to make the difference visible.

The density functions are estimated based on the training data sets, and the performance is measured by the predictive score, i.e., the average log-density evaluated at the held-out testing sets, one for each dataset.

We compare our approach with three normalizing flow (NF) methods using deep neural networks to construct transforms, which represent the state-of-the-art for density estimation in machine learning: MADE (Germain et al., 2015), Real NVP (Dinh et al., 2017), and MAF (Papamakarios et al., 2017). Their predictive scores
Figure 3.4: The estimated two-dimensional density functions. Each column corresponds to a simulation scenario, and each row corresponds to a value of $c_0$ (the scale of the learning rate).

are taken from Papamakarios et al. (2017) and Liu et al. (2021), where the detailed settings of the NF models are provided. For our boosting algorithm, the optimal values of $c_0$ and $\gamma$, tuning parameters for regularization, are chosen from $(0.1, 0.2, 0.3)$ and $(0.0, 0.1, \ldots, 1.0)$, respectively, with the 10-fold cross validation. The selected values are shown in Table 3.4 in Section 3.5.10.

A comparison of the predictive scores is provided in Figure 3.5, where our method is labeled as “boostPM” (which stands for “boosting probability measures”), and Table 3.2. Our unsupervised tree boosting is overall competitive with the NF methods.
and even shows the best predictive performance for two data sets (“POWER” and “ARem”). A visual comparison of the training data sets and replicated data sets simulated from the fitted generative model is provided in Figure 3.7, 3.8, and 3.9 in Section 3.5.10 and it confirms that the distributional structures are successfully captured.

It should be noted that among the considered methods, ours is the only one that is not based on neural networks but is a combination of the tree-based learners, and therefore requires only a tiny fraction of the computational cost to train. Table 3.3 provides the computation time measured in a single-core Intel Xeon Gold 6154 (3.00 GHz) CPU environment and shows our proposed boosting method is scalable for large \( n \) and \( d \) settings.

Figure 3.5: The comparison of the predictive scores. The points indicate the means, and the length of the error bars is two standard deviations.

3.3.3 Evaluating Variable Importance

In this section we compute the variable importance measure defined in Section 3.2.5 using the MNIST handwritten digits data (LeCun et al., 1998). The gray scale
Table 3.2: The average predictive scores and the standard deviations based on 30 repetition of applying the boosting algorithm under different random seeds.

<table>
<thead>
<tr>
<th></th>
<th>POWER</th>
<th>GAS</th>
<th>HEPMASS</th>
<th>MINIBOONE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.18 ± 0.003</td>
<td>9.32 ± 0.13</td>
<td>-19.48 ± 0.01</td>
<td>-17.22 ± 0.05</td>
</tr>
<tr>
<td>AReM</td>
<td>12.14 ± 0.04</td>
<td>22.07 ± 0.03</td>
<td>36.36 ± 0.05</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.3: The average training time for the benchmark datasets, measured in minutes (standard error in parentheses) in a single-core Intel Xeon Gold 6154 (3.00 GHz) CPU environment. Also provided are the sample size and dimensionality of the datasets.

<table>
<thead>
<tr>
<th></th>
<th>POWER</th>
<th>GAS</th>
<th>HEPMASS</th>
<th>MINIBOONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>1,659,917</td>
<td>852,174</td>
<td>315,123</td>
<td>29,556</td>
</tr>
<tr>
<td>d</td>
<td>6</td>
<td>8</td>
<td>21</td>
<td>43</td>
</tr>
<tr>
<td>Time</td>
<td>66.8 (2.6)</td>
<td>49.5 (2.4)</td>
<td>342.9 (10.2)</td>
<td>212.3 (7.2)</td>
</tr>
<tr>
<td>AReM</td>
<td>34,215</td>
<td>37,042</td>
<td>36,621</td>
<td></td>
</tr>
<tr>
<td>CASP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BANK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td>1.1 (0.1)</td>
<td>12.7 (0.6)</td>
<td>6.2 (0.5)</td>
<td></td>
</tr>
</tbody>
</table>

Images contained in this data consist of $28 \times 28 = 784$ pixels, and each pixel takes integer values ranging from 0 (black) to 255 (white). We obtain the data with the read_mnist function in the R package dslabs [Irizarry and Gill, 2021] and as in Papamakarios et al. (2017), scale them into [0, 1]. The tuning parameters $c_0$ and $\gamma$ are both set to 0.1.

Recall that our notion of variable importance characterizes how each variable contributes to the deviation from the uniform measure in the underlying sampling distribution. For the particular application of zip-code digit recognition, a pixel is more informative about the underlying digit if it has large variation over the range of intensities. As such, the practical meaning of “importance” in this particular application is the opposite to the statistical importance—it is exactly those pixels with intensities spread out over large ranges (and thus more uniform) that are informative.
about the underlying digit. As such, we want to emphasize the difference between the “practical importance” and that of the “distributional importance” in terms of KL as we defined before.

The computed “distributional importance” obtained for the ten different digits is visualized on the left of Figure 3.6 and a sample of handwriting of 0 is provided on the right. We can see that the pixels with relatively low “distributional importance” and hence high practical importance lie along the outlines of the digits. Hence in this case the “distributional importance” on the left side characterizes “the average shapes” of the handwritten numbers.

![Figure 3.6: Left: The importance of pixels (variables) computed for the 10 different digits. Yellow/purple colors correspond to low/high importance, which indicates large/small difference in the handwriting styles found at the pixels. Right: A sample of handwriting of 0.](image)

3.4 Concluding Remarks

We have proposed an unsupervised boosting method for learning multivariate probability measures by introducing new notions of addition and residuals based on tree-CDF transforms, and demonstrated how one can carry out density estimation and
simulate from the fitted measure based on the output of the algorithm. Given its similarity to classical boosting for regression and classification, we expect other techniques for the boosting in such contexts, for example subsampling (Friedman, 2002), could further improve the performance of our boosting method. Due to the limited space, we could not exploit all possible techniques in supervised boosting for improving the performance, but we expect many of them may be effective.

3.5 Technical Details

3.5.1 Proofs

In the following proofs, for a tree CDF $G : (0, 1]^d \mapsto (0, 1]^d$ and $B \in B(\Omega)$, the image is denoted by

$$G(B) = \{G(x) : x \in B\},$$

and the same notation rule is applied for the inverse $G^{-1}$. The Lebesgue measure is denoted by $\mu$.

Proof of Proposition 3.2.1 By the assumption on the full support, the CDFs are invertible.

Suppose $X \sim G_1 \oplus \cdots \oplus G_k$. For $x \in (0, 1]$, we have

$$P(G_i \circ \cdots \circ G_1(X) \leq x) = P(X \leq G_i^{-1} \circ \cdots \circ G_1^{-1}(x))$$

$$= G_k \circ \cdots \circ G_1(G_i^{-1} \circ \cdots \circ G_1^{-1}(x))$$

$$= G_k \circ \cdots \circ G_{i+1}(x),$$

so $G_i \circ \cdots \circ G_1(X) \sim G_{i+1} \oplus \cdots \oplus G_k$. The converse can be shown by transforming $P(X < x)$, where $G_i \circ \cdots \circ G_1(X)$, in the same way.
Proof of Proposition 3.2.2 The tree CDFs are already shown to be bijective in Proposition 3.2.2, so we only show the measurability here.

Let \( \mathcal{E} \) be a set of hyper-rectangles that are written in the form of

\[
(a_1, b_1] \times \cdots \times (a_d, b_d]
\]

including the null set \( \emptyset \). Since \( \mathcal{B}((0, 1]^d) \) is the Borel \( \sigma \)-field, \( E \in \mathcal{B}((0, 1]^d) \) holds for every \( E \in \mathcal{E} \). To show the measurability of the tree CDF \( \mathcal{G} \), which is defined by the measure \( G \in \mathcal{P}_T \), it suffices to show that \( \mathcal{G}^{-1}(E) \in \mathcal{B}((0, 1]^d) \) for every \( E \in \mathcal{E} \) since \( \mathcal{E} \) generates \( \mathcal{B}((0, 1]^d) \).

By the definition of \( \mathcal{G} \), the image \( \mathcal{G}(A) \ (A \in \mathcal{L}(T)) \) is also a hyper-rectangle included in \( \mathcal{E} \), and their collection \( \{ \mathcal{G}(A) : A \in \mathcal{L}(T) \} \) forms a partition of \( (0, 1]^d \). Hence \( E \) is written as a union of disjoint sets

\[
E = \bigcup_{A \in \mathcal{L}(T)} (E \cap \mathcal{G}(A)),
\]

where each \( E \cap \mathcal{G}(A) \) is a hyper-rectangle that belongs to \( \mathcal{E} \) and a subset of \( \mathcal{G}(A) \). Hence \( \mathcal{G}^{-1}(E \cap \mathcal{G}(A)) \) also belongs to \( \mathcal{E} \subset \mathcal{B}((0, 1]^d) \). Therefore, their finite union

\[
\mathcal{G}^{-1}(E) = \bigcup_{A \in \mathcal{L}(T)} \mathcal{G}^{-1}(E \cap \mathcal{G}(A))
\]

is also an element of \( \mathcal{B}((0, 1]^d) \).

Proof of Theorem 3.2.2 To prove the first assertion, define \( \mathcal{G}^{[r]} \) for \( r = 1, \ldots, R-1 \) as

\[
\mathcal{G}^{[r]}(x) = \sum_{A \in \mathcal{A}^r} \mathcal{G}_A(x) \mathbf{1}_A(x),
\]

. Then the first assertion is equivalent to that if \( X \sim G \), then

\[
\mathcal{G}^{[1]} \circ \cdots \circ \mathcal{G}^{[R-1]}(X) \sim Unif((0, 1]^d),
\]

(3.13)
which we prove here.

In the proof, we let $X^{[R]} = X$ and for $r = 1, \ldots, R - 1$

$$X^{[r]} = G^{[r]} \circ \cdots \circ G^{[R-1]}(X).$$

Let $G^{[r]}$ denote a probability measure for $X^{[r]}$. With these notations, we prove (3.13) by induction: We show that, if for $A \in \mathcal{N}(T)$, and

$$G^{[r+1]}(A_l) = G(A_l), \quad G^{[r+1]}(A_r) = G(A_r)$$

$$G^{[r+1]}(\cdot \mid A_l) = \mu(\cdot \mid A_l), \quad G^{[r+1]}(\cdot \mid A_r) = \mu(\cdot \mid A_r),$$

then

$$G^{[r]}(A) = G(A), \quad G^{[r]}(\cdot \mid A) = \mu(\cdot \mid A). \quad (3.15)$$

The conditions in Eq. (3.15) holds if $r = R - 1$ because $G^{[R]} = G$ and $G \in \mathcal{P}_T$, and the statement in Eq. (3.15) being true for $r = 1$ implies that $G^{[1]} \circ \cdots \circ G^{[R]}(\cdot) = \mu(\cdot)$, which is equivalent to Eq. (3.13).

Assume Eq. (3.14) holds for some $r$. By the definition, $G^{[r]}$ is bijective, and $G^{[r],-1}(A) = A$ for every $A \in \mathcal{A}^r$. Then for $X \sim G^{[r]}$ and $A \in \mathcal{A}^r$, we have,

$$G^{[r]}(A) = P(X^{[r]} \in A)$$

$$= P(G^{[r],-1}(X^{[r]} \in G^{[r],-1}(A))) = P(X^{[r+1]} \in A)$$

$$= G^{[r+1]}(A_l) + G^{[r+1]}(A_r) = G(A).$$

Hence the first equation in Eq. (3.15) holds. To prove the second equation, let $X^{[r]} = (X_1^{[r]}, \ldots, X_d^{[r]})$. Then, we show that for $z_j \in (a_j, b_j]$,

$$P(X_1^{[r]} \in (a_1, z_1], \ldots, X_d^{[r]} \in (a_d, z_d] \mid X^{[r]} \in A) = \prod_{j=1}^d \frac{z_j - a_j}{b_j - a_j} \quad (3.16)$$
holds. The probability in the left hand side can be written as

\[ P(X_1^r \in (a_1, z_1], \ldots, X_d^r \in (a_d, z_d] \mid X^r \in A) \]

\[ = P(X_1^r \in (a_1, z_1], \ldots, X_d^r \in (a_d, z_d] \mid X^{r+1} \in A) \]

\[ = \frac{P(X^{r+1} \in A_l)}{P(X^{r+1} \in A)} P(X_1^r \in (a_1, z_1], \ldots, X_d^r \in (a_d, z_d] \mid X^{r+1} \in A_l) \]

\[ + \frac{P(X^{r+1} \in A_r)}{P(X^{r+1} \in A)} P(X_1^r \in (a_1, z_1], \ldots, X_d^r \in (a_d, z_d] \mid X^{r+1} \in A_r) \]

\[ = G(A_l \mid A) P(X_1^r \in (a_1, z_1], \ldots, X_d^r \in (a_d, z_d] \mid X^{r+1} \in A_l) \]

\[ + G(A_r \mid A) P(X_1^r \in (a_1, z_1], \ldots, X_d^r \in (a_d, z_d] \mid X^{r+1} \in A_r). \quad (3.17) \]

Let \( A \) be divided in the \( j^* \)th dimension. By the definition of \( G_A \), for \( j \neq j^* \), \( X_j^r \in (a_j, z_j] \iff X_j^{r+1} \in (a_j, z_j] \). For \( j^* \), because \( G_{A,j^*}(\cdot) \) is strictly increasing,

\[ X_j^{r^*} \in (a_j^*, z_j^*] \iff G_{A,j^*}^{-1} \left( X_j^{r^*} \right) \in \left( G_{A,j^*}^{-1}(a_j^*), G_{A,j^*}^{-1}(z_j^*) \right) \]

\[ \iff X_j^{r^*+1} \in (a_j^*, y_{j^*}], \]

where \( y_{j^*} = G_{A,j^*}^{-1}(z_{j^*}) \). The expression of \( y_{j^*} \) changes depending on whether \( y_{j^*} \leq c_{j^*} \) or not, where \( c_{j^*} \) is a partition point at which \( A \) is divided. We first assume that \( y_{j^*} \leq c_{j^*} \). In this case, the second term in Eq. (3.17) is 0 because \( X_j^{r^*+1} \in (a_j^*, y_{j^*}] \) does not happen if \( X^{r+1} \in A_r \). Also, by the definition of \( G_{A,j^*} \),

\[ \frac{z_{j^*} - a_{j^*}}{y_{j^*} - a_{j^*}} = \frac{G(A_l \mid A)}{\mu(A_l \mid A)} = \frac{G(A_l \mid A)}{c_{j^*} - a_{j^*}} \]

\[ \iff \frac{y_{j^*} - a_{j^*}}{c_{j^*} - a_{j^*}} = \frac{1}{G(A_l \mid A)} \frac{z_{j^*} - a_{j^*}}{b_{j^*} - a_{j^*}}. \]
Therefore, it follows that
\[
P(X^r_1 \in (a_1, z_1], \ldots, X^r_d \in (a_d, z_d] \mid X^r \in A) = G(A_r \mid A) \prod_{j \neq j^*} \frac{z_j - a_j}{b_j - a_j} \frac{y_{j^*} - a_{j^*}}{c_{j^*} - a_{j^*}}
\]

\[
= G(A_r \mid A) \prod_{j \neq j^*} \frac{z_j - a_j}{b_j - a_j} \frac{1}{G(A_l \mid A)} \frac{z_{j^*} - a_{j^*}}{b_{j^*} - a_{j^*}}
\]

\[
= \prod_{j=1}^d \frac{z_j - a_j}{b_j - a_j}
\]

We can prove (3.16) for the case of \(y_{j^*} > c_{j^*}\) in the same way.

To prove the second result, let \(U \sim \text{Unif}((0, 1]^d)\). The multi-scale CDF \(\mathcal{G}\) is bijective (Proposition 3.2.2), so we obtain for \(B \in \mathcal{B}(\Omega)\)
\[
P(\mathcal{G}^{-1}(U) \in B) = P(U \in \mathcal{G}(B)) = \mu(\mathcal{G}(B)) = G(B),
\]
where the last line follows Theorem 3.2.1. (Note that the proof of Theorem 3.2.1 only uses the first result of Theorem 3.2.2. Therefore \(\mathcal{G}^{-1}(U) \sim G\).)

**Proof of Theorem 3.2.1** Let \(X \sim G\). By the first result of Theorem 3.2.2 we obtain
\[
G(B) = P(X \in B) = P(\mathcal{G}(X) \in \{\mathcal{G}(x) : x \in B\}) = \mu(\{\mathcal{G}(x) : x \in B\}).
\]

**Proof of Lemma 3.2.1** We only need to check the countable additivity. By Proposition 3.2.2 \(\mathcal{G}_k \circ \cdots \circ \mathcal{G}_1\) is bijective. Hence, for disjoint sets \(A_l \in \mathcal{B}(\Omega) (l \in \mathbb{N})\), it
follows that
\[ G_k \circ \cdots \circ G_1 \left( \bigcup_l A_l \right) = \bigcup_l G_k \circ \cdots \circ G_1 (A_l). \]

Because \( \{G_k \circ \cdots \circ G_1 (A_l)\}_{l=1,2,\ldots} \) are disjoint, this result implies that
\[ F_k \left( \bigcup_l A_l \right) = \mu \left( \bigcup_l G_k \circ \cdots \circ G_1 (A_l) \right) \]
\[ = \sum_l \mu(G_k \circ \cdots \circ G_1 (A_l)) \]
\[ = \sum_l F(A_l). \]

\[ \square \]

**Proof of Proposition 3.2.3** First we suppose \( X \sim G_1 \oplus \cdots \oplus G_k \). By Proposition 3.2.2, \( G_1, \ldots, G_k \) are all bijective, so for \( i = 1, \ldots, k \), the composition
\[ F_i = G_i \circ \cdots \circ G_1 \]
is also bijective. For \( B \in \mathcal{B}(\Omega) \), we obtain
\[ P(F_i(X) \in B) = P(X \in F_i^{-1}(B)) \]
\[ = F_k((F_i^{-1}(B))) \]
\[ = \mu(F_k(F_i^{-1}(B))) \]
\[ = \mu(G_k \circ \cdots \circ G_{i+1}(B)). \]

Hence \( F_i(X) \sim G_{i+1} \oplus \cdots \oplus G_k \). Showing the converse is now straightforward. \( \square \)

**Proof of Proposition 3.2.4** We first show the following lemma, which implies that the conditional distributions \( F_{k-1} \) and \( F_k \) (\( k = 2, \ldots, K \)) are the same on subsets in a partition defined by \( T_k \) and \( F_{k-1} = G_{k-1} \circ \cdots \circ G_1 \).

**Lemma 3.5.1.** Let \( \mathcal{L}(T_k) \) be a set of the terminal nodes in \( T_k \). Also, we let \( A' \in \mathcal{L}(T_k) \) and \( A = F_{k-1}^{-1}(A') \). Then for any \( B \subset A \), \( F_{k-1}(B \mid A) = F_k(B \mid A) \).
(Proof) For $B \subset A$, by Theorem 3.2.1, we have
\[
F_k(B) = \mu(F_k(A))
\]
\[
= \mu(G_k(F_{k-1}(B)))
\]
\[
= G_k(F_{k-1}(B)). \tag{3.18}
\]

Since $B \subset A$, $F_{k-1}(B) \subset F_{k-1}(A) = A'$. Hence $F_k(B)$ is further rewritten as follows:
\[
F_k(B) = G_k(A' \cap F_{k-1}(B))
\]
\[
= G_k(A')G_k(F_{k-1}(B) \mid A')
\]
\[
= G_k(A')\mu(F_{k-1}(B) \mid A')
\]
\[
= G_k(A')\frac{\mu(F_{k-1}(B))}{\mu(A')}
\]
\[
= G_k(A')\frac{F_{k-1}(B)}{\mu(A')}. \tag{3.18}
\]

By the definition of $A$, $F_{k-1}(A) = \mu(F_{k-1}(A)) = \mu(A')$, and by replacing $B$ with $A$ in (3.18), we have
\[
F_k(A) = G_k(F_{k-1}(A)) = G_k(A').
\]

Therefore, we obtain
\[
F_k(B \mid A) = \frac{F_k(B)}{F_k(A)}
\]
\[
= G_k(A')\frac{F_{k-1}(B)}{\mu(A')} \frac{1}{F_k(A)}
\]
\[
= \frac{F_{k-1}(B)}{F_{k-1}(A)}
\]
\[
= F_{k-1}(B \mid A).
\]

(Proof of Proposition 3.2.4) Let $\mathcal{L}_k = \{F_{k-1}(A') : A' \in \mathcal{L}(T_k)\}$. By Lemma 3.5.1, the conditional distributions $F_{k-1}(\cdot \mid A)$ and $F_k(\cdot \mid A)$ are the same for $A \in \mathcal{L}_k$. Hence
the density functions of $F_k$ and $F_{k-1}$ denoted by $f_k$ and $f_{k-1}$ are expressed as

$$f_{k-1}(x) = \sum_{A \in \mathcal{L}_k} F_{k-1}(A)f_{k-1}(x \mid A)1_A(x),$$

$$f_k(x) = \sum_{A \in \mathcal{L}_k} F_k(A)f_{k-1}(x \mid A)1_A(x),$$

where $1_A$ is the indicator function. Fix $x \in (0, 1]^d$ and let $A_k \in \mathcal{L}_k$ such that $x \in A_k$ and $A_k' = \mathcal{F}_{k-1}(A_k)$. By Theorem 3.2.1, we have

$$F_k(A_k) = \mu(\mathcal{F}_k(A_k)) = \mu(A_k'),$$

$$F_{k-1}(A_k) = \mu(\mathcal{F}_{k-1}(A_k)) = \mu(\mathcal{F}_{k-1}(A_k)).$$

Hence, we have

$$\frac{f_k(x)}{f_{k-1}(x)} = \frac{F_k(A_k)}{F_{k-1}(A_k)} = \frac{G_k(A_k')}{\mu(A_k')}.$$ (3.19)

Since $x \in A_k$,

$$\mathcal{F}_{k-1}(x) = \mathcal{G}_{k-1} \circ \ldots \circ \mathcal{G}_1(x) \in \mathcal{F}_{k-1}(A_k) = A_k'.$$

Thus the density ratio in (3.19) is rewritten as

$$\frac{f_k(x)}{f_{k-1}(x)} = G_k(A_k')\mu(\mathcal{G}_{k-1} \circ \ldots \circ \mathcal{G}_1(x) \mid A_k')$$

$$= G_k(A_k')g_k(\mathcal{G}_{k-1} \circ \ldots \circ \mathcal{G}_1(x) \mid A_k')$$

$$= g_k(\mathcal{G}_{k-1} \circ \ldots \circ \mathcal{G}_1(x)),$$

where the second equation follows that $A_k' \in \mathcal{L}(T_k)$. Because the discussion above
holds for $k = 2, \ldots, K$, we obtain the following expression:

$$f_K(x) = f_1(x) \prod_{k=2}^{K} \frac{f_k(x)}{f_{k-1}(x)}$$

$$= f_1(x) \prod_{k=2}^{K} g_k(G_{k-1} \circ \cdots \circ G_1(x)).$$

\[\square\]

**Proof of Lemma 3.2.2** By the definition of the KL divergence, we have

$$\text{KL}(F^*||F) = \int \log f^* dF^* - \int \log f dF^*$$

By Proposition 3.2.4, the second term $\int \log f^* dF^*$ is decomposed as

$$\int \log f dF^* = \sum_{k=1}^{K} \int \log g_k(G_{k-1} \circ \cdots \circ G_1(x)) dF^*(x)$$

By the change-of-variable formula (e.g., Theorem 3.6.1 in [Bogachev (2007)]), the right hand side can be written in a form of integration with respect to $\widetilde{F}_k$:

$$\int \log g_k(G_{k-1} \circ \cdots \circ G_1(x)) dF^*(x) = \int \log g_k(x) d\widetilde{F}_k(x).$$

Since the measure $F^*$ is absolutely continuous with respect to the Lebesgue measure $\mu$, by the definition of $\widetilde{F}_k$, that is,

$$\widetilde{F}_k(B) = F^*(G_{1}^{-1} \circ \cdots \circ G_{k-1}^{-1}(B)) \quad \text{for all } B \in \mathcal{B}((0,1]^d),$$

$\widetilde{F}_k$ is also absolutely continuous with respect to $\mu$. Hence $\widetilde{F}_k$ admits the density function denoted by $\tilde{f}_k(x)$, and the right hand side is further rewritten as follows:

$$\int \log g_k(x) d\widetilde{F}_k(x) = \int \log \frac{\tilde{f}_k(x)}{\mu(x)} d\widetilde{F}_k(x) - \int \log \frac{\tilde{f}_k(x)}{g_k(x)} d\widetilde{F}_k(x)$$

$$= \text{KL}(\widetilde{F}_k||\mu) - \text{KL}(\widetilde{F}_k||G_k). \quad \square$$
Proof of Proposition 3.2.5 Since $T_k$ is a finite tree, the log of $g_k$, which is piece-wise constant on $T_K$, is written as

$$\log g_k(x) = \sum_{A \in \mathcal{L}(T_k)} \log \frac{G_k(A)}{\mu(A)} 1_A(x) \quad \text{for} \ x \in (0, 1]^d.$$ 

Hence the improvement $D_k^{(n)}(G_k)$ is rewritten as follows:

$$D_k^{(n)}(G_k) = \sum_{A \in \mathcal{L}(T_k)} \tilde{F}_k^{(n)}(A) \left( \log \frac{G_k(A)}{\mu(A)} \right) = \sum_{A \in \mathcal{L}(T_k)} \tilde{F}_k^{(n)}(A) \log \frac{\tilde{G}_k^{(n)}(A)}{\mu(A)} - \sum_{A \in \mathcal{L}(T_k)} \tilde{F}_k^{(n)}(A) \log \frac{\tilde{G}_k^{(n)}(A)}{G_k(A)}.$$ 

Because the second term in the bottom line takes a form of KL divergence defined for the two discrete distributions, it is minimized if $G_k(A) = \tilde{F}_k^{(n)}(A)$ for all $A \in \mathcal{L}(T_k)$. Under this $G_k$, since the second term is 0, the improvement is maximized if $T_k$ satisfies the condition provided in Proposition 3.2.5. \hfill \square

Proof of Proposition 3.2.6 In this proof, we suppose the learning rate $c(A)$ is independent to a node $A$ for simplicity. For every leaf node $A \in \mathcal{L}(T_k)$, there is a sequence of nodes $\{B_{A,r}\}_{r=1}^R$ such that $B_{A,r}$ belongs to the $r$th level of $T_k$, and

$$(0, 1]^d = B_{A,1} \supset B_{A,2} \supset \cdots \supset B_{A,R} = A.$$ 

With such sequences, based on the discussion in Section 3.5.1, the improvement $D_k^{(n)}(G_k)$ is decomposed as

$$D_k^{(n)}(G_k) = \sum_{A \in \mathcal{L}(T_k)} \tilde{F}_k^{(n)}(A) \left( \log \frac{G_k(A)}{\mu(A)} \right) = \sum_{A \in \mathcal{L}(T_k)} \tilde{F}_k^{(n)}(A) \log \frac{G_k(B_{A,r} | B_{A,r-1})}{\mu(B_{A,r} | B_{A,r-1})}.$$ 

112
For the bottom line the summand is 0 if $\tilde{F}_k^{(n)}(A) = 0$. Otherwise, the conditional probabilities $\tilde{F}_k^{(n)}(A_l \mid A)$ and $\tilde{F}_k^{(n)}(A_r \mid A)$ are defined. In such a case, by the definition of $G_k(A_l \mid A)$,

$$
\log \frac{G_k(A_l \mid A)}{\mu(A_l \mid A)} = \log \left[ \frac{(1 - c)\mu(A_l \mid A) + c\tilde{F}_k^{(n)}(A_l \mid A)}{\mu(A_l \mid A)} \right] \\
\geq (1 - c) \log 1 + c \log \frac{\tilde{F}_k^{(n)}(A_l \mid A)}{\mu(A_l \mid A)} \\
= c \log \frac{\tilde{F}_k^{(n)}(A_l \mid A)}{\mu(A_l \mid A)},
$$

where the second line follows the Jensen’s inequality. The same result holds for $A_r$.

Hence,

$$
\tilde{F}_k^{(n)}(A_l) \log \frac{G_k(A_l \mid A)}{\mu(A_l \mid A)} + \tilde{F}_k^{(n)}(A_r) \log \frac{G_k(A_r \mid A)}{\mu(A_r \mid A)} \\
\geq c\tilde{F}_k^{(n)}(A) \left[ \tilde{F}_k^{(n)}(A_l \mid A) \log \frac{\tilde{F}_k^{(n)}(A_l \mid A)}{\mu(A_l \mid A)} + \tilde{F}_k^{(n)}(A_r \mid A) \log \frac{\tilde{F}_k^{(n)}(A_r \mid A)}{\mu(A_r \mid A)} \right],
$$

where the sum inside of the brackets is the KL divergence for the two Bernoulli distributions and thus non-negative. Therefore, the improvement $D_k^{(n)}(G_k)$ is non-negative. Additionally, the last inequality is strict if and only if $\tilde{F}_k^{(n)}(A_l \mid A) = \mu(A_r \mid A)$ and so $D_k^{(n)}(G_k)$ is positive.

### 3.5.2 Expressive Power of the Tree Ensemble

In this section, we provide theoretical results on the expressive power of the tree ensemble with the final goal of proving Theorem 3.2.3.

#### Preparations

We introduce the following notations:
1. Let $\mathcal{T}^L$ be a collection of dyadic trees with axis-aligned boundaries with at most $L$ maximum resolution. When $L = d$, $\mathcal{T}^d$ is a set of trees that can be formed under Assumption 3. We note that as implied in the following proofs, $\mathcal{T}^L$ can a set of trees that have at least one node reach the $L$th while the other leaf nodes belong to the shallower levels.

2. For a tree $T \in \mathcal{T}^L$, a set $\mathcal{P}_T$ denotes a collection of probability measures conditionally uniform on $T$ such that

$$G(\cdot \mid A) = \mu(\cdot \mid A) \text{ and } G(A) > 0 \quad (3.20)$$

for every terminal node $A \in T$. A collection of such tree measures are denoted by $\mathcal{G}_0^L$, that is,

$$\mathcal{G}_0^L = \{G : G \in \mathcal{P}_T \text{ for some } T \in \mathcal{T}^L\}.$$ 

For a measure $G \in \mathcal{G}_0^L$ defined on a tree $T \in \mathcal{T}^L$, we can define a tree-CDF as in Section 3.2 which is denoted by $\mathcal{G}$. We define a set $\mathcal{G}_0^L$ as a collection of such tree CDFs, namely,

$$\mathcal{G}_0^L = \{G : G \text{ is a tree CDF of } G \in \mathcal{G}_0^L\}.$$ 

3. Let $\mathcal{G}^L$ denote a set of finite composition of tree CDFs, that is,

$$\mathcal{G}^L = \{G_K \circ \cdots \circ G_1 : K \in \mathbb{N} \text{ and for } k = 1, \ldots, K, \ G_k \in \mathcal{G}_0^L\}$$ 

and define $\mathcal{G}^L$ as a collection of probability measures defined by such finite compositions, that is,

$$\mathcal{G}^L = \{\mu(G(\cdot)) : G \in \mathcal{G}^L\}.$$ 

Hence $\mathcal{G}^L$ includes all measures that can be expressed in the form of ensemble $G_1 \oplus \cdots \oplus G_K$. 

114
We also need to review the definition of push-forward measures because this notation is closely related to the operation of residualization. Let $\varphi$ be a mapping $\Omega \mapsto \Omega$ and $H$ be a probability measure. Then the push-forward of $H$ is defined in the following form:

$$\varphi \# H(B) = H(\varphi^{-1}(B)) \text{ for } B \in \mathcal{B}(\Omega).$$

The following lemma establishes a connection between the ensemble measure and the push-forward measures.

**Lemma 3.5.2.** For a probability measure $F$, $F \in G^L$ holds if and only if there exists a mapping $G \in G^L$ such that $G \# F = \mu$.

(Proof) Suppose $F \in G^L$. Then there exists a mapping $G \in G^L$ such that

$$F(B) = \mu(G(B)) \text{ for } B \in \mathcal{B}(\Omega).$$

From Proposition 3.2.2, $G$ is bijective. Hence for $B \in \mathcal{B}(\Omega)$, we have

$$\mu(B) = \mu(G \circ G^{-1}(B)) = F(G^{-1}(B)),$$

so $G \# F = \mu$. The necessity can be shown in the same way. 

In the rest of the section, we first discuss the expressive power of the tree ensemble for the uni-variate cases and next generalize the result for the multi-variate cases. After that, this result is used to prove Theorem 3.2.3.

**3.5.3 Uni-variate Cases**

The following proposition shows that any distribution with piece-wise constant and positive densities can be represented in the form of tree ensemble.
Proposition 3.5.1. Let $F$ be a probability measure that admits the piece-wise constant density $f$ with the following form:

$$f(x) = \sum_{i=1}^{I} \beta_i 1_{(c_{i-1}, c_i]} ,$$

where $\beta_i > 0$ for $i = 1, \ldots, I$ and

$$0 = c_0 < c_1 < \cdots < c_I = 1.$$

Then, if $L \geq 2$, $F \in G^L$ holds.

(Proof) We first show the existence of a tree CDF $G_1 \in G_2^0$ such that the push-forward measure $G_1 \# F$ has a density $f_1$ with the following form

$$f_1(x) = \sum_{i=1}^{I-1} \tilde{\beta}_i 1_{(\tilde{c}_{i-1}, \tilde{c}_i]} ,$$

(3.21)

where $\tilde{\beta}_i > 0$ for $i = 1, \ldots, I - 1$ and $0 = \tilde{c}_0 < \tilde{c}_1 < \cdots < \tilde{c}_{I-1} = 1$.

Let $\alpha \in (0, 1)$ be a constant that satisfies

$$\frac{1 - \alpha}{\alpha} = \frac{\beta_2 1 - c_1}{\beta_1 c_1}.$$

Then define a measure $G_1 \in G_0^2$ such that

$$G_1((0, c_1]) = \alpha, \quad G_1((c_1, 1]) = 1 - \alpha$$

and $G_1$ is conditionally uniform on $(0, c_1]$ and $(c_1, 1]$. Let $G_1$ be $G_1$'s tree CDF and $F_1 = G_1 \# F$ be a probability measure with the density $f_1$. For $x \in (0, \alpha]$, we have

$$F_1((0, x]) = F(G_1^{-1}((0, x])) = F((0, G_1^{-1}(x)]) = \int_0^{G_1^{-1}(x)} f \, d\mu.$$

Hence, by the chain rule, the density at this $x$ is written as

$$f_1(x) = \frac{c_1}{\alpha} f(G_1^{-1}(x)) = \frac{c_1}{\alpha} \beta_1.$$
Similarly, the density at \( x \in (\alpha, 1] \) is written as
\[
f_1(x) = \frac{1 - c_1}{1 - \alpha} f \left( G_1^{-1}(x) \right).
\]

Let \( \tilde{c}_1 = G_1(c_{i+1}) \) for \( i = 1, \ldots, I - 1 \). By this definition, \( \alpha < \tilde{c}_1 \), and the density of \( f_1 \) at \( x \in (\alpha, \tilde{c}_1] \) satisfies
\[
f_1(x) = \frac{1 - c_1}{1 - \alpha} \beta = \frac{c_1}{\alpha} \beta_1,
\]
where the second equation follows the definition of \( \alpha \). Hence \( f_1 \) is constant on \((0, \tilde{c}_1]\).

Moreover, the density on \((\tilde{c}_{i-1}, \tilde{c}_i] \) for \( i = 2, \ldots, I - 1 \) is \((1 - c_1)/(1 - \alpha) \beta_{i-1}\) so constant. Therefore the density \( f_1 \) is written in the form of Eq (3.21).

By using the same logic for the rest of the \( I - 2 \) discontinuous points, we can define tree CDFs \( G_2, \ldots, G_{I-1} \) that connect the densities at these points one by one. Hence the measure \((G_{I-1} \circ \cdots \circ G_1)\#F \) has a constant density and thus is the uniform measure \( \mu \).

\[\Box\]

3.5.4 Multi-variate Cases

In this section, we prove the following proposition that is a generalization of Proposition 3.5.1.

**Proposition 3.5.2.** For \( j = 1, \ldots, d \), let \( \{c_{j,i}\}_{i=1}^{I_j} \) be a sequence such that
\[
0 = c_{j,0} < c_{j,1} < \cdots < c_{j,I_j} = 1,
\]
and \( \mathcal{L} = \{A_{i_1,\ldots,i_d}\}_{i_1,\ldots,i_d} \) be a partition of the sample space \((0, 1]^d\) that consists of rectangles written as
\[
A_{i_1,\ldots,i_d} = (c_{1,i_1-1}, c_{1,i_1}] \times \cdots \times (c_{d,i_d-1}, c_{d,i_d}].
\]

If a probability measure \( F \) is piecewise uniform on \( \mathcal{L} \) and written as
\[
F(B) = \sum_{i_1,\ldots,i_d} a_{i_1,\ldots,i_d} \frac{\mu(B \cap A_{i_1,\ldots,i_d})}{\mu(A_{i_1,\ldots,i_d})}, \text{ for } B \in B((0, 1]^d),
\]

117
where $a_{i_1,\ldots,i_d} > 0$, then for $L \geq d + 1$, there is a mapping $\mathcal{G} \in \mathcal{G}^L$ such that $\mathcal{G} \# F = \mu$ and thus $F \in \mathcal{G}^L$. In addition, we can choose $\mathcal{G}$ so that for every pair of indices $(i_1, \ldots, i_d)$, the image $\mathcal{G}(A_{i_1,\ldots,i_d})$ is a rectangle written as

$$(\mathcal{G}(c_{1,i_1-1}), \mathcal{G}(c_{1,i_1})) \times \cdots \times (\mathcal{G}(c_{d,i_d-1}), \mathcal{G}(c_{d,i_d})).$$

(Proof) We use induction: We assume that the statement of Proposition 3.5.2 is valid for the 1, 2, \ldots, (d-1)-dimensional cases. Because in this proof we handle measures and transformation defined in different dimensional spaces, the sets $\mathcal{G}^L$ and $\mathcal{G}^{L,d}$ defined for the $j$-dimensional space are denoted by $\mathcal{G}^{L,d}$ and $\mathcal{G}^{L,d}$, respectively.

Inside of the induction, we also assume that for some $l \in \{1, \ldots, I_d-1\}$, there are mappings $\mathcal{G}_1, \ldots, \mathcal{G}_l \in \mathcal{G}^{L,d}$ such that a probability measure $F_l := (\mathcal{G}_l \circ \cdots \circ \mathcal{G}_1) \# F$ is a piecewise uniform probability measure written as, for $B \in \mathcal{B}((0, 1]^d)$,

$$F_l(B) = \sum_{i=1}^{l} C_i \frac{\mu(B \cap (0, 1]^{d-1} \times (c_{d,i-1}, c_{d,i}))}{\mu((0, 1]^{d-1} \times (c_{d,i-1}, c_{d,i}))} + \sum_{i_d=l+1}^{I_d} \sum_{i_{i_d}=1}^{I_{i_d}} a_{i_1,\ldots,i_d}^{(l)} \frac{\mu(B \cap A_{i_1,\ldots,i_d}^{(l)})}{\mu(A_{i_1,\ldots,i_d}^{(l)})},$$

where $C_i > 0$ and $a_{i_1,\ldots,i_d}^{(l)} > 0$ for all indices. Also, for the second term, $A_{i_1,\ldots,i_d}^{(l)}$ is a rectangular written as

$$A_{i_1,\ldots,i_d}^{(l)} = \left( c_{1,i_1-1}^{(l)}, c_{1,i_1}^{(l)} \right] \times \cdots \times \left( c_{d-1,i_{i_d}-1}^{(l)}, c_{d-1,i_{i_d}}^{(l)} \right],$$

$$\times \left( c_{d,I_d-1}, c_{d,d} \right].$$

where for $j = 1, \ldots, d-1$, $\{c_j^{(l)}\}_{i=1}^{I_j}$ is a sequence such that

$$0 = c_{j,1}^{(l)} < c_{j,2}^{(l)} < \cdots < c_{j,I_j}^{(l)} = 1.$$  

(We note that this sequence’s length can be different from “$I_j$” provided in Proposition 3.5.2 but to avoid an excessive number of indices, we use $I_j$ here because its size does not affect the logic provided in this proof.) Under this assumption, we show that there is a measure (“$F_{l+1}$”) that has the same form for $l + 1$.  

118
Define a $d - 1$-dimensional probability measure $\hat{F}_{l+1}$

\[
\hat{F}_{l+1} = \sum_{i_1, \ldots, i_d} \frac{a^{(l)}_{i_1, \ldots, i_d, l+1} \mu_{d-1}(B \cap \hat{A}^{(l)}_{i_1, \ldots, i_d})}{C_{l+1} \mu_{d-1}(\hat{A}^{(l)}_{i_1, \ldots, i_d})} \quad \text{for } B \in \mathcal{B}((0, 1]^{d-1}),
\]

where $C_{l+1}$ is the normalizing constant, $\mu_{d-1}$ is the Lebesgue measure defined for the $d - 1$-dimensional sample space, and $\hat{A}^{(l)}_{i_1, \ldots, i_d}$ is a set written as

\[
\hat{A}^{(l)}_{i_1, \ldots, i_d} = \left[ c^{(l)}_{1, i_1-1} \right] \times \cdots \times \left[ c^{(l)}_{d-1, i_{d-1}-1}, c^{(l)}_{d-1, i_{d-1}} \right].
\]

Because $\hat{F}$ is a piecewise uniform measure defined on the partition that consists of hyper-rectangles, by the assumption we set for the induction, there is a mapping $\hat{G}_{l+1} \in \mathcal{G}^{L-1,d-1}$ such that $\hat{G}_{l+1} \# \hat{F}_{l+1} = \mu_{d-1}$. With this mapping, we define a mapping $\hat{G}_{l+1} : (0, 1]^d \mapsto (0, 1]^d$ such that for $x = (x_1, \ldots, x_d) \in (0, 1]^d$,

\[
\hat{G}_{l+1}(x) = \left( \hat{G}_{l+1}(x_1, \ldots, x_{d-1}), x_d \right)
\]

if $x_d \in (c_{d,l}, 1]$ and otherwise $\hat{G}_{l+1}(x) = x$. The mapping $\hat{G}_{l+1}$ moves points only in $(0, 1]^{d-1} \times (c_{l,d}, 1]$, which is a node one can obtain by dividing the sample space only once, according to $\hat{G}_{l+1}$, which is a mapping that is a composition of tree CDFs based on trees with $L - 1$ leaf nodes. Hence $\hat{G}_{l+1}$ is a composition of tree CDFs defined on trees with $(L - 1) + 1 = L$ leaf nodes, so we have $\hat{G}_{l+1} \in \mathcal{G}^{L,d}$. With this mapping, we define a measure $F_{l+1} = \hat{G}_{l+1} \# F_l$.

Fix a pair of indices $(i_1, \ldots, i_d)$ and let $B_{d-1} \in \mathcal{B}((0, 1]^{d-1})$ and $B_1 \in \mathcal{B}((0, 1])$ be measurable sets such that

\[
B_{d-1} \times B_1 \in \hat{G}_{l+1}(\hat{A}^{(l)}_{i_1, \ldots, i_d}) \times (c_{d,i_d-1}, c_{d,i_d}].
\]

If $i_d \leq l$, by the definition of $F_{l+1}$ and $\hat{G}_{l+1}$,

\[
F_{l+1}(B_{d-1} \times B_1) = F_l(\hat{G}_{l+1}^{-1}(B_{d-1} \times B_1)) = F_l(B_{d-1} \times B_1).
\]

119
On the other hand, if \( i_d \geq l + 1 \), since \( F_l \) is conditionally uniform on \( A_{i_1, \ldots, i_d}^{(l)} \),

\[
F_{l+1}(B_{d-1} \times B_1) = F_l(G_{l+1}^{-1}(B_{d-1} \times B_1)) = F_l(\hat{G}_{l+1}^{-1}(B_{d-1}) \times B_1)
\]

\[
= a^{(l)}_{i_1, \ldots, i_d} \frac{\mu(\hat{G}_{l+1}^{-1}(B_{d-1}) \times B_1)}{\mu(A_{i_1, \ldots, i_d}^{(l)})}
\]

\[
= a^{(l)}_{i_1, \ldots, i_d} \frac{\mu_{d-1}(\hat{G}_{l+1}^{-1}(B_{d-1})) \mu_1(B_1)}{\mu_{d-1}(A_{i_1, \ldots, i_{d-1}}^{(l)}) \mu_1([c_{d,i_{d-1}}, c_{d,i_d}])),
\]

where \( \mu_1 \) is the Lebesgue measure defined for the 1-dimensional sample space. For such \( i_d \), by the definition of \( F_{l+1} \) and \( \hat{G}_{l+1} \),

\[
\mu_{d-1}(B_{d-1}) = \hat{F}_{l+1}(\hat{G}_{l+1}^{-1}(B_{d-1})) = \frac{a^{(l)}_{i_1, \ldots, i_{d-1}, i_d} \mu_{d-1}(\hat{G}_{l+1}^{-1}(B_{d-1}))}{C_{l+1}} \mu_{d-1}(A_{i_1, \ldots, i_{d-1}}^{(l)}),
\]

from which we obtain

\[
F_{l+1}(B_{d-1} \times B_1) = C_{l+1} a^{(l)}_{i_1, \ldots, i_{d-1}, i_d} \frac{\mu_{d-1}(B_{d-1}) \mu_1(B_1)}{\mu_{d-1}(A_{i_1, \ldots, i_{d}}^{(l+1)})} \mu_1([c_{d,i_{d-1}}, c_{d,i_d}])),
\]

\[
= \begin{cases} 
C_{l+1} \frac{\mu(B_{d-1} \times B_1)}{\mu([0,1]^{d-1} \times [c_{d,i_{d-1}}, c_{d,i_d}]])} & (i_d = l + 1), \\
C_{l+1} \frac{\mu(B_{d-1} \times B_1)}{\mu_{d-1}(A_{i_1, \ldots, i_d}^{(l+1)})} (i_d > l + 1), 
\end{cases}
\]

(3.23)

where, for \( i_d > l + 1 \),

\[
\hat{A}_{i_1, \ldots, i_d}^{(l+1)} = \hat{G}_{l+1}(\hat{A}_{i_1, \ldots, i_d}^{(l)}) \times (c_{d,i_{d-1}}, c_{d,i_d}].
\]

and

\[
a^{(l+1)}_{i_1, \ldots, i_d} = C_{l+1} a^{(l)}_{i_1, \ldots, i_{d-1}, i_d} \mu_{d-1}(\hat{G}_{l+1}(\hat{A}_{i_1, \ldots, i_d}^{(l)}), (i_d \leq l + 1)
\]

Because \( B((0, 1]^{d-1}) \times B((0, 1]) \) generates \( B((0, 1]^{d-1}) \), from the discussion provided above, \( F_{l+1} \) is piecewise-uniform on a partition that consists of

\[
(0, 1]^{d-1} \times (c_{d,i_{d-1}}, c_{d,i_d}] (i_d \leq l + 1)
\]
and \( \hat{A}_{i_1,\ldots,i_d}^{(l+1)} \) \((i_d > l + 1)\), and this partition is denoted by \( \mathcal{P}^{(l+1)} \). Note that by the definition of \( \hat{G}_{l+1} \), and Proposition 3.5.2, which we assume holds for \((d-1)\)-dimensional cases, the image of the hyper-rectangle of \( \hat{A}_{i_1,\ldots,i_d}^{(l)} \) under \( \hat{G}_{l+1} \) and \( \hat{A}_{i_1,\ldots,i_d}^{(l+1)} \) are a hyper-rectangle in the \((d-1)\)-dimensional space and the \(d\)-dimensional space, respectively.

The following lemma states that the partition structure \( \mathcal{P}^{(l+1)} \) has a finer partition that has the “checker-board” form, as shown in the next lemma.

**Lemma 3.5.3.** Let \( \{D_i\}_{i=1}^L \) is a partition of the sample space \((0,1]^d\) such that every \( D_i \) is a hyper-rectangle. Then, there are sequences \( \{e_{j,i_j}\}_{i_j} \) \((j = 1, 2, \ldots, d)\) such that

\[
0 = e_{j,0} < e_{j,1} < \cdots < e_{j,l_j} = 1
\]

and a partition \( \{E_{i_1,\ldots,i_d}\}_{i_1,\ldots,i_d} \) defined as

\[
E_{i_1,\ldots,i_d} = (e_{1,i_1-1}, e_{1,i_1}] \times \cdots \times (e_{d,i_d-1}, e_{d,i_d}]
\]

such that every \( D_i \) is a finite union of elements of \( \{E_{i_1,\ldots,i_d}\}_{i_1,\ldots,i_d} \).

Its proof is straightforward because we only need to “extend” the boundaries between the rectangles \( \{D_i\}_{i=1}^L \). By applying this lemma to the partition \( \mathcal{P}^{(l+1)} \), it follows that there are finite sequences \( \{c_{i_j}\}_{i_j} \) \((j = 1, \ldots, d - 1)\) such that a checkerboard-like partition consisting of the following type of rectangles

\[
A_{i_1,\ldots,i_d}^{(l+1)} := (c_{1,i_1-1}, c_{1,i_1}] \times \cdots \times (c_{d-1,i_d-1}, c_{d-1,i_d-1} - 1] \times (c_{d,i_d-1}, c_{d,i_d}]
\]

is finer than \( \mathcal{P}^{(l+1)} \). With this partition, the measure \( F_{l+1} \) is written as for \( B \in \mathcal{B}((0,1]^d) \)

\[
F_{l+1}(B) = \sum_{i=1}^{l+1} C_i \mu(B \cap (0,1]^{d-1} \times (c_{d,i-1}, c_{d,i}]) + \sum_{i=l+2}^{l+1} \sum_{i_1,\ldots,i_{d-1}} a_{i_1,\ldots,i_d}^{(l)} \mu(B \cap A_{i_1,\ldots,i_d}^{(l)}).
\]
Because this result holds for $l = 1, \ldots, I_d - 1$, there exists a sequence of mappings $\mathcal{G}_1, \ldots, \mathcal{G}_{I_d} \in \mathcal{G}^{L, d}$ such that a push-forward measure $H := (\mathcal{G}_{I_d} \circ \cdots \circ \mathcal{G}_1) \# F$ has a form

$$H(B) = \sum_{i=1}^{I_d} C_i \frac{\mu(B \cap (0, 1]^{d-1} \times (c_{d,i-1}, c_{d,i})]}{\mu((0, 1]^{d-1} \times (c_{d,i-1}, c_{d,i}))}.$$

Define an one-dimensional probability measure $\hat{H}$

$$\hat{H}(B_1) = \sum_{i=1}^{I_d} C_i \frac{\mu_1(B_1 \cap (c_{d,i-1}, c_{d,i})]}{\mu_1((c_{d,i-1}, c_{d,i})]} \text{ for } B_1 \in \mathcal{B}((0, 1])$$

Then, by the assumption (or Proposition 3.5.1), there exists a mapping $\hat{G}_0 \in \mathcal{G}^{2,1}$ such that $\hat{G}_0 \# \hat{H} = \mu_1$. With this mapping, we define another mapping $\mathcal{G}_0 : (0, 1] \mapsto (0, 1]$ such that for $x = (x_1, \ldots, x_d)$,

$$\mathcal{G}_0(x) = (x_1, \ldots, x_{d-1}, \hat{G}_0(x_d)).$$

This mapping moves input points only in the $d$th dimension according to $\hat{G}_0$ so it is written as a composition of tree CDFs defined on trees with 2 terminal node and thus an element of $\mathcal{G}^{L, d}$. Hence $\mathcal{G}_0 \in \mathcal{G}^{L, d}$. Fix $i \in \{1, \ldots, I_d\}$. For a measurable set $B_{d-1} \times B_1$ such that

$$B_{d-1} \times B_1 \in \mathcal{B}((0, 1]^{d-1}) \times \hat{G}_0((c_{d,i-1}, c_{d,i}]),$$

because $H$ is piecewise uniform, we have

$$\mathcal{G}_0 \# H(B_{d-1} \times B_1) = H(\mathcal{G}_0^{-1}(B_{d-1} \times B_1)) = H(B_{d-1} \times \hat{G}_0^{-1}(B_1))$$

$$= C_i \frac{\mu(B_{d-1} \times \hat{G}_0^{-1}(B_1))}{\mu((0, 1]^{d-1} \times (c_{d,i-1}, c_{d,i}))}$$

$$= C_i \frac{\mu_{d-1}(B_{d-1})\mu_1(\hat{G}_0^{-1}(B_1))}{\mu_1((c_{d,i-1}, c_{d,i})]}.$$
On the other hand,

\[ \mu_1(B_1) = \hat{H}(\hat{G}_0^{-1}(B_1)) = C_1 \frac{\mu_1(\hat{G}_0^{-1}(B_1))}{\mu_1((c_{d,i-1}, c_{d,i}))}. \]

Hence, we obtain

\[ G_0 \# H(B_{d-1} \times B_1) = \mu_{d-1}(B_{d-1})\mu_1(B_1) = \mu(B_{d-1} \times B_1). \]

Therefore, we conclude that

\[ G_0 \# H = (G_0 \circ G_{1d} \circ \cdots \circ G_1) \# H = \mu. \]

The result of Proposition 3.5.2 can be described in a simplified form as in the next corollary. This proof immediately follows Proposition 3.5.2 and Lemma 3.5.3.

**Corollary 3.5.1.** Let \( \{E_i\}_{i=1}^L \) is a partition of the sample space \( (0, 1]^d \) such that \( E_i \) is a rectangle with a form

\[ E_i = (a_{i,1}, b_{i,1}] \times \cdots \times (a_{i,d}, b_{i,d}], \]

and \( F \) be a piecewise uniform probability measure defined on the partition:

\[ F(B) = \sum_{i=1}^L \beta_i \frac{\mu(B \cap E_i)}{\mu(E_i)} \text{ for } B \in \mathcal{B}((0, 1]^d), \]

where \( \beta_i > 0 \). Then \( F \in \mathcal{G}^L \) for \( L \geq d + 1 \).

**3.5.5 Proof of Theorem 3.2.3**

We finally provide the proof of Theorem 3.2.3, which can be obtained by adding minor modifications to the proof of Theorem 4 in Wong and Ma (2010).

Let \( f^* \) denote \( F^* \)'s density function, and we first assume that \( f^* \) is uniformly continuous. For \( \epsilon > 0 \), there exists \( \epsilon' > 0 \) such that \( \log(1 + \epsilon') < \epsilon \). Since the function \( f^* \) is uniformly continuous, there exists \( \delta > 0 \) such that

\[ |x - y| < \delta \Rightarrow |f^*(x) - f^*(y)| < \epsilon'. \]
Let \( \{E_i\}_{i=1}^I \) is a partition of the sample space \((0, 1]^d\) such that \(E_i\) has a rectangle shape and \(\text{diam}(E_i) < \delta\). Define a function \(\tilde{g}\) as

\[
\tilde{g} = \sum_{i=1}^I \left\{ \sup_{x \in E_i} f^*(x) \right\} 1_{E_i}(x) \text{ for } x \in (0, 1]^d.
\]

Let \(C = \int \tilde{g}d\mu\). Because \(\tilde{g}(x) \geq f^*(x)\) for \(x \in (0, 1]^d\), we have \(C \geq 1\) and

\[
0 \leq C - 1 = \int (\tilde{g} - f^*)d\mu = \sum_{i=1}^I \int_{E_i} (\tilde{g}(x) - f^*(x))d\mu
\]

\[
\leq \sum_{i=1}^I \int_{E_i} \epsilon'd\mu = \epsilon'.
\]

Define a density function \(g := \tilde{g}/C\). The corresponding probability measure \(G\) is an element of \(\mathcal{G}^L\) by Corollary 3.5.1. Hence, for the two measures \(F^*\) and \(G\), we can bound the KL divergence as follows:

\[
KL(F || G) = \int f^* \log \frac{f^*}{g} d\mu = \int f^* \log \frac{f^*}{\tilde{g}} d\mu + \int f^* \log C d\mu
\]

\[
\leq \log C \leq \log(1 + \epsilon') < \epsilon.
\]

We next consider the general case, where we assume \(f^* \leq M\) for some \(M > 0\). By Lusin’s theorem, for any \(\tilde{\epsilon} > 0\), there exits a closed set \(B\) such that \(\mu(B^c) < \tilde{\epsilon}\) and \(f^*\) is uniformly continuous on \(B\). Using this fact, we modify the first discussion as follows. The definition of \(\tilde{g}\) is modified as follows: If \(E_i \cap B \neq \emptyset\), for \(x \in E_i\), we let

\[
\tilde{g}(x) = \sup_{x \in E_i \cap B} f^*(x).
\]

Otherwise, \(g(x) = M\). With this modification, we obtain

\[
0 \leq C - 1 = \int (\tilde{g} - f^*)d\mu = \int_B (\tilde{g} - f^*)d\mu + \int_{B^c} (\tilde{g} - f^*)d\mu
\]

\[
\leq \epsilon' + M\tilde{\epsilon},
\]

which can be arbitrarily small, so the same result follows.
3.5.6 Details on Learning Probability Measures with the Pólya Tree Process

This section provides details on the weak learner we use to fit tree measures to the residuals in the estimation. The algorithm is based on the PT-based method proposed in Awaya and Ma (2021), and interested readers may refer to this paper.

3.5.7 Theoretical Justification of Using the PT-based Model

As shown in Section Proposition 3.2.5, the improvement in the entropy loss is maximized when a fitted tree is a solution of the problem

\[
\arg \max_{T \in T} \sum_{A \in \mathcal{L}(T)} \tilde{F}^{(n)}_k(A) \log \frac{\tilde{F}^{(n)}_k(A)}{\mu(A)},
\]

where \( \tilde{F}^{(n)}_k \) is the empirical measure defined by the residuals \( \{r_i^{(k-1)}\}_{i=1}^n \). As \( n \to \infty \), the empirical measure \( \tilde{F}^{(n)}_k(B) \) converges to \( \tilde{F}_k(B) \) for \( B \subset (0,1]^d \), where \( \tilde{F}_k \) is the true distribution of the residuals defined by the previous tree-CDFs \( G_1, \ldots, G_{k-1} \). At this population level, the maximization problem is written as

\[
\arg \max_{T \in T} \sum_{A \in \mathcal{L}(T)} \tilde{F}_k(A) \log \frac{\tilde{F}_k(A)}{\mu(A)},
\]

and we can show that this maximization is equivalent to minimizing the KL divergence \( KL(\tilde{F}_k||\tilde{F}_k|_T) \), where \( \tilde{F}_k|_T \) is “a tree-approximation of \( \tilde{F}_k \) under \( T \)”, namely,

\[
\tilde{F}_k|_T(B) = \sum_{A \in \mathcal{L}(T)} \tilde{F}_k(A) \frac{\mu(B \cap A)}{\mu(A)} \text{ for } B \in (0,1]^d.
\]

Theorem 4.1 in Awaya and Ma (2021) shows that the posterior of trees also concentrates on the minimizer of \( KL(\tilde{F}_k||\tilde{F}_k|_T) \), and this result implies that at the population level, or when \( n \) is large, we can find the tree that maximizes the improvement in the entropy loss or similar ones by checking the posterior of trees.
3.5.8 Details on the Sampling Algorithm

Suppose we have obtained the residuals at the beginning of the boosting algorithm. Since the task of fitting a new measure to the residuals is essentially the same for all steps, we drop the $k$, the index of the trees and measures consisting of the ensemble, from the notations for simplicity. Then the residuals are denoted by $r = (r_1, \ldots, r_n)$, and our task at each step is to capture their distributional structure by fitting a dyadic tree. In the section, we provide details on the prior distributions introduced for the tree $T$ and the stochastic top-down algorithm we use to find a tree with good fitting.

Prior Distribution of $T$

As in Awaya and Ma (2021), we construct a prior of $T$ by introducing the random splitting rule for each node $A$. First, we introduce the stopping variable $S(A)$ that takes 0 or 1, and if $S(A) = 1$, we stop splitting $A$ and otherwise split $A$. Here we set $P(S(A) = 1)$ to 0.5. In the latter case, we next define the dimension variable $D(A)$ and the location variable $L(A)$. If $D(A) = j$ ($j = 1, \ldots, d$), the node $A$ is split in the $j$th dimension, and the location of the boundary is determined by $L(A) \in (0, 1)$, in which 0 (or 1) corresponds to the left (or right) end point. Their prior distributions are as follows:

$$P(D(A) = j) = 1/d, \ (j = 1, \ldots, d),$$

$$P(L(A) = l/N_L) = \frac{1}{N_L - 1} \ (l = 1, \ldots, N_L - 1),$$

where $N_L - 1$ is the number of grid points, which is 127 in the estimation.

On the tree $T$, we also define a random measure $\tilde{G}$, with which we can define the likelihood of the residuals $r$. The prior of the measure $\tilde{G}$ is defined by introducing the parameters $\theta(A) = \tilde{G}(A_l \mid A)$, where $A_l$ is the left children node, for every
non-terminal node $A$. They follow the prior distribution specified as

$$
\theta(A) \sim \text{Beta}(\theta_0(A), 1 - \theta_0(A)), \quad \theta_0(A) = \frac{\mu(A)}{\mu}. 
$$

We note that this random measure $\tilde{G}$ is introduced just to define the marginal posterior of $T$, namely, $P(T \mid r)$, since our main goal is to find a tree that fits the distribution of the residuals. In the estimation, we first choose one tree according to this posterior and construct the measure to output, which is denoted by $G_k$ in the paper, as in Eq (3.12). The method to select one tree is described in more detail in the next section.

The joint model of the tree $T$ and the measure $\tilde{G}$ can be seen as a special case of the density estimation model that is referred to as the adaptive Pólya tree (Ma, 2017) model in Awaya and Ma (2021) with the number of the latent states being 2.

**Top-down Stochastic Algorithm**

The particle filter proposed in Awaya and Ma (2021) is shown to be effective to sample from the posterior of trees. This original algorithm, however, has drawbacks when seen as a component of the boosting from a viewpoint of computational cost:

1. In the particle filter, we construct thousands of candidate trees, but this strategy may make the whole boosting algorithm too time-consuming since in the boosting algorithm we need to repeat fitting trees to the residuals many times.

2. In the original algorithm, we do not stop splitting nodes until we reach the bottom nodes unless the number of included observations is too small. (Technically speaking, this is because the stopping variables, or the latent variables in general, are integrated out in the sampling.) The number of nodes generated in a tree, however, tends to be large especially when the sample size is large, and constructing such large trees repeatedly in the boosting algorithm is also
too-time consuming. The computation cost would become reasonable if we “give up non-promising nodes”, that is, stop dividing nodes if no interesting structures are found there.

From these reasons, we modify the original algorithm as follows: (i) Instead of generating many candidate trees, we set the number of particles to one, that is to say, construct a tree by randomly splitting nodes on the tree in a top-down manner. Hence the algorithm is similar to the top-down greedy method, but in our algorithm one selects splitting rules stochastically. (ii) For each active node, we compare possible splitting rules and the decision of stopping the splitting, where the latter option is added to the algorithm. This comparison is based on their posterior probabilities, and the splitting tends to be stopped if the conditional distribution is close to uniform.

For an active node $A$, the possible decisions are compared based on the following quantities that are seen as “prior × marginal likelihood”. (A conceptually very similar algorithm for supervised learning is proposed in He and Hahn (2021).) For the decision of stopping, we compute

$$L_\emptyset = P(S(A) = 1)\mu(A)^{-n(A)},$$

where $n(A)$ is the number of residuals included in $A$. On the other hand, for the
splitting rule $D(A) = j$ and $L(A) = l/N_L$ that divides $A$ into $A_l$ and $A_r$, we compute

$$L_{j,l} = P(S(A) = 0, D(A) = j, L(A) = l/N_L)$$

$$\times \int Beta(\theta | \theta_0(A), 1 - \theta_0(A)) \theta^{n(A_l)}(1 - \theta)^{n(A_r)} d\theta$$

$$\times \mu(A_l)^{-n(A_l)} \mu(A_r)^{-n(A_r)}$$

$$= P(S(A) = 0, D(A) = j, L(A) = l/N_L)$$

$$\times \frac{Be(\theta_0(A) + n(A_l), 1 - \theta_0(A) + n(A_r))}{Be(\theta_0(A), 1 - \theta_0(A))}$$

$$\times \mu(A_l)^{-n(A_l)} \mu(A_r)^{-n(A_r)},$$

where $Be(\cdot)$ is the beta function. Based on these quantities, we choose to stop the splitting with probability

$$\frac{L_\emptyset}{L_\emptyset + \sum_{j=1}^d \sum_{l=1}^{N_L - 1} L_{j,l}}.$$ 

Otherwise, we choose the splitting rule $D(A) = j$ and $L(A) = l/N_L$ with probability

$$\frac{L_{j,l}}{\sum_{j'=1}^d \sum_{l'=1}^{N_L - 1} L_{j',l'}}.$$ 

### 3.5.9 Details of the Two-dimensional Experiments

In the experiment, we used the following three scenarios.

**Scenario A:** $n = 1,000$, and

$\text{Normal}(\mu, \Sigma)$,

where

$$\mu = \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} 1/8^2 & 0.95/8^2 \\ 0.95/8^2 & 1/8^2 \end{bmatrix}.$$
**Scenario B:** \( n = 5,000 \), and
\[
\frac{1}{10} \beta(x_1 \mid 1, 1) \times \beta(x_2 \mid 1, 1) + \frac{3}{10} \beta(x_1 \mid 15, 45) \times \beta(x_2 \mid 15, 45)
+ \frac{3}{10} \beta(x_1 \mid 45, 15) \times \beta(x_2 \mid 22.5, 37.5)
+ \frac{3}{10} \beta(x_1 \mid 37.5, 22.5) \times \beta(x_2 \mid 45, 15).
\]

**Scenario C:** \( n = 2,000 \), and
\[
\frac{1}{3} 1_{[0.1,0.45] \times [0.35,0.9]}(x_1, x_2) + \frac{1}{3} 1_{[0.2,0.8] \times [0.45,0.5]}(x_1, x_2)
+ \frac{1}{3} 1_{[0.7,0.9] \times [0.05,0.6]}(x_1, x_2).
\]

### 3.5.10 Additional Tables and Figures

**Table 3.4:** The values of \( c_0 \) and \( \gamma \) (the parameters of the learning rate) selected for the data sets.

<table>
<thead>
<tr>
<th>POWER</th>
<th>GAS</th>
<th>HEPMASS</th>
<th>MINIBOONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.1, 0.0)</td>
<td>(0.3, 0.0)</td>
<td>(0.1, 0.7)</td>
<td>(0.1, 0.7)</td>
</tr>
<tr>
<td>AReM</td>
<td>CASP</td>
<td>BANK</td>
<td></td>
</tr>
<tr>
<td>(0.3, 0.0)</td>
<td>(0.1, 0.6)</td>
<td>(0.1, 0.2)</td>
<td></td>
</tr>
</tbody>
</table>
Figure 3.7: The training set of the POWER data (a subset of size 10,000 is visualized) and 10,000 observations simulated from the learned probability measure.

Figure 3.8: The training set of the GAS data (a subset of size 10,000 is visualized) and 10,000 observations simulated from the learned probability measure.
Figure 3.9: The training set of the AReM data (a subset of size 10,000 is visualized) and 10,000 observations simulated from the learned probability measure.
Concluding Remarks

This thesis presented the two new tree-based methods for learning probability distributions. In Chapter 2, we introduce the new single tree method by constructing the general Pólya tree-based model with the new flexible tree prior and the novel sequential Monte Carlo algorithm. The new method is shown to have better empirical performance compared to the existing parametric/non-parametric Bayesian methods in density estimation and two-sample comparison. In Chapter 3, we define the new ensemble model combining multiple probability measures under the new addition rule and propose the new boosting algorithm to construct the ensemble. In the numerical experiments using the real-world data sets, the new ensemble method have a predictive performance competitive with the state-of-the-art density estimators despite its cheap computational cost.

A few things need to be noted about the two works.

Chapter 2: *Hidden Markov Pólya trees for high-dimensional distributions*

- Though the proposed sequential Monte Carlo algorithm is shown to be effective to search trees with good fitting efficiently, the algorithm is not free from the
so-called particle degeneracy problem. In the case of the new coarse-to-fine algorithm, since the particles are resampled repeatedly, the sampled trees tend to have similar structures, and their difference often exists only in the deep layers, which adds only minor diversity in the partition structures. In the particle filter algorithms in general, the degeneracy problem is known to be solved with the backward smoothing (Doucet et al., 2009), but introducing the smoothing for the tree samplers is difficult because modifying nodes keeping their descendant nodes unchanged is not straightforward. As such keeping the diversity of the particles is still an important but very challenging task at this point.

- The application of our new HMPT method is demonstrated in density estimation and two-sample comparison in Chapter 2, but as mentioned in Section 2.1 it can be applied for other various inference tasks such as density regression (Jara and Hanson, 2011) and hierarchical modeling (Christensen and Ma, 2020). Using our proposed method would improve the empirical performance and also enables analyzing high-dimensional data sets that cannot not be handled with the existing Pólya-tree based methods.

- The program code to run the sequential Monte Carlo algorithm is available in [https://github.com/MaStatLab/boostPM](https://github.com/MaStatLab/boostPM) and the codes used in the experiments are stored in [https://github.com/MaStatLab/HMPT_experiments](https://github.com/MaStatLab/HMPT_experiments).

Chapter 3: *Unsupervised tree boosting for learning probability measures*

- The proposed boosting algorithm is general in that users can use their preferred weak-learners as long as they output tree-measures. As implied by the theoretical property on the relationship between the improvement in the KL divergence and the choice of trees, however, the high predictive performance of
the boosting can be achieved only when the weak learners accurately capture the distributional structures of the residuals. Using the tree learner proposed in Chapter 2 (strictly speaking, its simplified version) is a good option as shown in the empirical studies. Nevertheless, we should note that we can integrate other density estimators as weak learners, which might improve the empirical performance.

- The research focuses on building the boosting algorithm using tree measures and the new class of functions called tree-CDFs as its integrate components. Technically speaking, however, a similar boosting built on the basic notions such as “residuals” can be constructed for other types of probability measures as long as we can define functions that correspond to the tree-CDFs. For example, as demonstrated in Inouye and Ravikumar (2018), it is possible to define transformations that flatten distributions for the Gaussian mixtures and build the boosting-like algorithm in which one repeats fitting the Gaussian mixtures to the residuals.

- It should be clarified that the proposed boosting method is not a Bayesian method though the Bayesian tree learner can be used as a component. This fact makes us naturally interested in introducing a Bayesian approach based on the proposed ensemble model because it would be effective for uncertainty quantification and, more importantly, the Bayesian method would show better empirical performance as the Bayesian ensemble method for unsupervised learning tends to show better empirical performance than the other ensemble methods (Chipman et al., 2010). However, introducing a Bayesian method, especially a posterior sampling algorithm is very challenging since the addition rule defined for distributions is not communicative (symmetric). For example, if we want to introduce a back-fitting as commonly done for the Bayesian
ensemble methods (Chipman et al. 2010) and update $G_k$ in the ensemble

$$G_1 \oplus \cdots \oplus G_{k-1} \oplus G_k \oplus G_{k+1} \oplus \cdots \oplus G_K,$$

we need to compute “difference” between the observations and the other measures $\{G_i\}_{i \neq k}$. Subtracting the previous measures $G_1, \ldots, G_{k-1}$ is possible as we can calculate the residuals as in the boosting, but the problem is that how to subtract the subsequent measures $G_{k+1}, \ldots, G_K$ is unclear. Hence how to construct a feasible sampling algorithm is still an open problem.

- The program code to run the boosting algorithm is available in [https://github.com/MaStatLab/boostPM](https://github.com/MaStatLab/boostPM).


Biography

Naoki Awaya was born in Saitama, Japan. He completed a B.S. degree in Economics from the University of Tokyo in March 2015. He received an M.S. in Economics from the University of Tokyo in March 2017 and joined the Ph.D. program in the Department of Statistical Science, at Duke University in August 2018. He won a student/postdoc best paper award at the 2021 ISBA world meeting. He plans to graduate with his Ph.D. in Statistical Science from Duke University in September 2022.