The Dissertation Committee for Omar Demian Chavez
certifies that this is the approved version of the following dissertation:

Scalable and Causal Bayesian Inference

Committee:

Sinead Williamson, Supervisor

Michael J. Daniels

Antonio Linero

Tom Shively

Scalable and Causal Bayesian Inference

by

Omar Demian Chavez

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2021

Dedicated to my wife Jarinya, son Phoenix and parents, Sonny and Violet Chavez.

Acknowledgments

I thank my wife Jarinya for her continued support and encouragement through our journey together through life and in particular my graduate education years. I also want to say a word of appreciation for my son Phoenix for helping me to stay motivated to share with a young mind the value of the scientific method and the pursuit of knowledge. I thank my parents, Sonny and Violet Chavez, for supporting me through all my years of education, beginning with elementary pre-school at Colegio Casa Montessori and ending with the Doctor of Phillosophy in Statistics and Data Sciences and always encouraging me to achieve more than I think I am able to before I begin. I would also like to thank my grandmother, Gustavina Carrasco, my aunt Patricia Armendariz and the rest of my family for all their support through my formative years and beyond.

I would like to thank my advisor, Dr. Sinead Williamson, for her commitment, patience, guidance, and advice. Mike Daniels for his continued guidance and intuition contributing to the work in this dissertation and the other committee members, Antonio Linero and Thomas Shively.

I thank both my committee and the Department of Statistics and Data Science for their support during my graduate career. I thank Vicki Keller for being such a supportive coordinator for the doctoral program and always looking out for the best interests of the people in the department. I would also like to thank Maurice Diesendruck and Guy Cole for their collaborations and insights in the subject of statistics and life in general.

I would also like to thank my friends Aaron Von Flatern, Scott Dill, Patrick Jones and Dr. Fumiko Futamura for their continued support and advice in the pursuit of excellence.

Finally, I wish to thank the many other people who helped me to reach this point.

Scalable and Causal Bayesian Inference

Omar Demian Chavez, Ph.D.
The University of Texas at Austin, 2021

Supervisor: Sinead Williamson

This thesis will focus on two facets of Bayesian estimation. First, we propose methods that can improve parameter estimation in particle filtering when making use of a distributed computing environment by allowing for periodic communication between compute nodes. The periodic communication can improve the embarrassingly parallel version of our particle filter without dramatically increasing the computational costs. Our method is intended for use on data with large N or in streaming settings where latent parameters are changing over time. Secondly, we propose a method for estimating heterogeneous treatment effects in observational studies using transformed response variables via a modification to Bayesian additive regression trees that incorporates a mixture model in the regression error terms.

Table of Contents

List of Tables

List of Figures

Chapter 1

Overview

This thesis will focus on Bayesian methods in two areas. First, we introduce a new method based on almost embarrassingly parallel particle filtering. Second, we modify the Bayesian Additive Regression Trees algorithm in an application to estimating heterogeneous treatment effects.

The advent of High Performance (HPC) and parallel computing has opened the door to types of statistical analysis that were formerly unthinkable (Reumann et al., 2012), (Ko et al., 2021). This relatively new technology has always shown promise but with the acceptance of practitioners, the promise is coming into its own as an exciting new reality. Now that we are at or near the physical limitations for clock speeds of a computer's central processing units, new ideas of how to make use of computation will need to be explored to advance statistical and scientific computation. Now, increases in compute power comes from the use of multiple cores within a computer processor chip. Modern HPC clusters or machines are equipped with more than one CPU that work on the same problem, thus enabling parallelism. Technology has also enabled other frontiers that are poised to propel statistics into the future.

The continuous generation of data has led to a world where more and more the catch phrase "big data" is becoming more the norm than the exception. Organizations are having to make deliberate choices of what not to save as there is little ability to host in a data base, tempting as it may be to "record everything", all the observable events. Being as it is, there is tremendous value in understanding how to work with these new and now commonly available big data scenarios as they can deliver tremendous value to an organization. Models that that can process either large observational count data or large

parameter models that capture individual level effects have the capacity to significantly change an organizations bottom line.

In addition to advances in computation, there has also been an evolution in statistical thinking stemming from needs in sociology, economics, finance and practical industry needs to go beyond studying association to studying *causation*. Often, in observational settings, interventions are administered and need a *post-hoc* analysis to assess the effect of the intervention. Alternatively, a company may need to understand the effect of their product's price on demand. These are examples of problems that are designed to be addressed with *causal inference* (CI). In CI settings there is generally the ability to infer population level effects which can be quite useful in assessing population level decision. There is also often the need to assess individual level effects which can help in such clinical settings as assessing the quality of an intervention on a specific subset of a population.

Chapter 3 explores how advances in distributed computing can leverage the ability to send messages over the network while learning from data to improve parameter estimation and statistical inference. We show for two separate models (2 different data generating processes) how allowing communication between compute nodes and/or processes, each having its own data partition, allows us to often improve model fit.

Chapter 4 explores applications in Causal Inference for observational data using a transformation of the response outcome that is a function of the response, an indicator variable designating treatment assignment, and the probability of treatment assignment. Our proposed model also makes use of a newly devised reparameterization of the error distribution proposed in (Zaidi and Mukherjee, 2018) allowing the model to better reflect the mixture nature of the transformed response variable (TRV). We accomplish this by modifying Bayesian Additive Regression Trees and adapting them to the causal setting. The modification is accomplished with BART by jointly modeling the unobserved treatment effect function and a second function that is also unobserved comprised of the response under the control and treatment conditions as well as the probability of treatment assignment. We show how our model is a preferred alternative to other competing models also using the TRV with respect

to reducing the error in heterogeneous treatment effect estimates. Additionally, we show how our model is competitive with other state of the art methods not using the TRV. We also discuss a shortcoming of the parameterization related to modeling the response functions which serve as a limiting factor in inference quality of the proposed method.

Chapter 2

Background

2.0.1 Bayesian Inference

Bayesian inference is a method of statistical inference reliant of Bayes' Theorem. Bayes theorem is a mathematical formulation of updating the probability of a hypothesis given information or evidence. Bayes theorem is particularly useful in the dynamic analysis of sequential data and in applying regularization for probabalistic models. Formally, give two events A and B, Bayes' Theorem simply states

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

This formulation is very powerful and opens the door to interesting ways of interpreting both observed and unobserved phenomenon. Specifically, related to observed phenomenon, if we think of observed phenomenon as being represented by a set of data or records, X, θ as a set of parameters related to the mechanism by which the data is generated and $P(X)$ and $P(\theta)$ as probability distributions for the data and parameters, then

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)} \tag{2.1}$$

which reflects the a posterior belief about θ given the evidence, X. This quantity is proportional to the likelihood $P(X|\theta)$ and the prior belief about θ, reflected in a prior distribution, $P(\theta)$, and is inversely proportional to a normalizing constant $P(X)$ reflecting the data generating mechanism.

Many approaches have been proposed in Bayesian Inference to estimate the posterior distribution, $P(\theta|X)$. Chapters 3 and 4 will both rely on Monte Carlo Sampling methods to estimate target distribution by producing samples intended to approximate the posterior conditional on evidence. Due to the high dimensional nature of our parameter space, θ, it can be intractable to produce samples of our parameter space at once. Because of this, methods that partition the parameter space have been devised and used with great success. As an example, consider a partition of $\theta = \{\theta_1, \theta_2\}$, where we then write equation 2.1 as

$$P(\theta_1|X, \theta_2) = \frac{P(X|\theta_1, \theta_2)P(\theta_1, \theta_2)}{P(X, \theta_2)} \tag{2.2}$$

Under this formulation, we are able to sample from the conditional distribution of θ_1 conditional on the other set of parameters and the data. Chapter 3 makes use of this technique, as it allows us to estimate parameters that are evolving in time as well as model data in streaming settings. Chapter 4 on the other hand does not have parameters that are evolving in time or deal with streaming data but still has a high dimensional parameter space. We therefore employee techniques such as Gibbs Sampling (Geman and Geman, 1984), Metropolis-Hastings (Metropolis et al., 1953) and Bayesian Backfitting (Hastie and Tibshirani, 2000), all of which relay on the framing provided by equation 2.2.

This formulation using Bayes theorem is also particularly useful in applying regularization to model parameters as depending on the specification for $P(\theta)$, this can have the effect of shrinking our estimates of θ towards our prior belief about θ. Models such as CART (Breiman et al., 1984a), BART (Chipman et al., 1998), and LASSO (Tibshirani, 1996) all make use of shrinkage to varying degrees to estimate model parameters. We make use of this technique in Chapter 4.

2.0.2 Sequential Monte Carlo (Particle Filtering)

Sequential Monte Carlo methods are a set of simulation based methods that provide a convenient way to approximate posterior distributions. One ad-

vantage of SMC methods is their ease of implementation and their ability to be parallelized, making them ideally suited to scaleable solutions. To accomplish this, particle filtering uses a set of samples, also referred to as particles, to represent a posterior distribution of some parameter or set of parameters. Particle filters can be used to estimate the state of a signal that evolves through time by assigning to each particle a likelihood weight proportional to the probability of that particle being sample from a target distribution. As a simple example, consider a set of parameters θ and a set of observations $X = X_1, ..., X_T$. SMC methods can formulate the problem of estimating the target distribution using a modification of equation 2.1,

$$\begin{aligned} P(\theta|X) &\propto P(X|\theta)P(\theta) \\ &= P(\theta)P(X_1|\theta)...P(X_T|\theta) \end{aligned}$$

And in so doing, we can think of $P(\theta)P(X_1|\theta)...P(X_{t-1}|\theta)$ as a prior on $P(X_t|\theta)$. Iterating on this idea we can sequentially learn the parameter θ from the data. This is one of the the foundational ideas of SMC and particle filtering which we use to great extent in Chapter 3.

2.0.3 Causal Inference

Causal Reasoning, the desire to understand cause and effect relationships, is known to at least date back to the fourth century BC with Aristotle's Physics (Falcon, 2015) where we find the first known protoscientific study of cause and effect. One of the primary types and applications of causal reasoning involves causal inference, with a primary focus being treatment effect estimation. A treatment effect is the effect a binary variable has on a response outcome of scientific or policy interest. In particular, the treatment effect measures the difference in outcomes between a unit(s) assigned to treatment vs a unit(s) assigned to the control condition. There have been two primary and heavily influential approaches to estimating treatment effects. The first termed do calculus based on graphical models introduced by (Pearl, 2009) and the other based on the potential outcomes framework (POF) introduced by (Rubin, 1978). In the POF, we are provided a set of sufficient conditions needed to define a causal effect or more specifically, a numeric quantity known

as the causal estimand that can be estimated from the data and who's effect on the response can be attributed to some intervention. For a hypothetical intervention, the causal effect for a unit is the difference that would be observed for a particular unit administered the intervention versus the same unit not administered the intervention. This is the fundamental problem of causal inference. That is, we are only capable of observing a single state of the unit. The unit administered the intervention or the unit not administered the intervention but never both. Consequently, unit level causal effects are not identifiable but under some standard assumptions discussed in Section 4.4, causal effects are identifiable. In this work, we develop a methodology that extends from the POF defined in (Rubin, 1978).

2.0.3.1 Transformed Response Variable for Causal Inference

In some sense, we can think of a unit or individual level treatment effect as a difference of two regression functions:

$$\tau(x) = E[Y|X = x, W = 1] - E[Y|X = x, W = 1] \tag{2.3}$$

where $\tau(x)$ is the unit level treatment effect, Y is a continuous outcome response, X is a set of observable covariates, and W is a binary indicator equal to 1 for units administered treatment and 0 otherwise. A consequential property of the unit level treatment effect is that it equals the population level average treatment effect after marginalizing over X. An alternative approach that avoids formulating the problem via two regression functions was popularized by (Athey and Imbens, 2015a) for estimating the unit level treatment effects. We discuss the potential pros and cons in more detail in chapter 4.

The alternative formulation termed the transformed response variable (TRV) is constructed as follows. If we denote our observed response value Y_i, our indicator variable indicating treatment assignment as W_i and the probability of treatment assignment (propensity score) as $P(W_i = 1|X) = p(x_i)$. For ease of notation, we denote $p(x_i)$ as p_i. The transformed response variable is defined as:

$$Y_i^* = \frac{Y_i(W - p_i)}{p_i(1 - p_i)} \tag{2.4}$$

The expectation of Y_i^* is the individual causal effect

$$E[Y_i^*|X_i = x] = \tau(x)$$

We provide the proof in Appendix B.1 and in depth discussion and analysis in chapter 4.

2.0.3.2 Bayesian Additive Regression Trees

Decision trees have found vast popularity since their broad introduction with (Breiman et al., 1984a) which made their dissemination possible due to computers and available software. (Chipman et al., 1998) introduced a Bayesian version to probabilistically model a response variable conditional on the covariates by recursively partitioning the predictor space into subsets where the distribution of y is successively more homogeneous. Taking this idea further, (Chipman et al., 2010) extended CART into a boosting framework known as Bayesian Additive Regression Trees where the method fits a sequence of single trees, using each tree to fit residual data variation not explained by earlier trees in the sequence. BART has a attained wide popularity due to its extensive flexibility to model response surfaces and its ablity to produces full posterior inference including point and interval estimates of the unknown regression function. Owing to the flexibility of the method and its ease of modification, many have contributed to its expanded use. Modifications for BART for risk and survival analysis were proposed in (Sparapani et al., 2018) and (Sparapani et al., 2016). There are also adaptations to address function surface smoothness in (Linero and Yang, 2018) and (Starling et al., 2020a), estimating functions with heteroskedastic error (Pratola et al., 2020), adaptations that deal with sparsity and variable selection (Linero, 2018) and (Liu et al., 2021), categorical and count response outcomes (Murray, 2021), as well as adaptations for causal inference (Hahn et al., 2019); (Hill, 2011); (Starling et al., 2020b). For a full scale review of modeling developments regarding BART as of 2020, see (Hill et al., 2020).

At its core, BART is a semi-parametirc regression model that estimates the conditional expectation of a response variable Y_i with a sum of weak learners, i.e., the trees. If one wishes to flexibly model a response function with

BART, we can do so with:

$$f(\mathbf{X}) = \sum_{j=1}^{m} g_j(\mathbf{X}, (T_j, \mathcal{M}_j)) \tag{2.5}$$

where m is the total number of trees in the model; the pair $(T_j, \mathcal{M}_j)$ defines the structure of the j_{th} tree. Specifically, T_j defines the split rules and $\mathcal{M}_j = \eta_{j,1}, ..., \eta_{j,b}$, the set of parameters associated with the b terminal nodes in tree j. The function $g(.)$ is a single "weak" tree learner mapping the covariates X to a set of leaf nodes, $\mathcal{M}_j$ which are determined by the set of splitting rules governed by T_j. The conditional mean function $f(\mathbf{X}) = E(Y_i|\mathbf{X})$ is computed by summing all the terminal nodes $\eta_{j,b}$ assigned to X by the functions $g_j(.)$. There is extensive discussion about useful prior specifications for BART in (Chipman et al., 1998) and (Chipman et al., 2010). We go into further detail regarding prior specifications for our method in section 4.5.2

Chapter 3

Communal Monte Carlo Particle Filtering

3.1 Overview

In this chapter we explore how in a setting where we have streaming data and a parallel computing environment typically used for embarrassingly parallel model fitting, it is possible to significantly improve model fitting characteristics. We accomplish this by employing a relatively simple communication strategy between computational processes that typically would run to completion before sharing any parameter information. We validate our hypothesis with three separate trials. The first is with a dynamic linear regression model and the second is with a dynamic linear logistic regression model.

3.2 Introduction

The recent methodologies for big data can be loosely grouped into three categories. They are resampling-based methods, divide and conquer methods, and methods that perform online updating. In this chapter we introduce Communal Monte Carlo which makes use of techniques typical of divide and concur and online updating approaches. To summarize, we employ at a base level multiple sequential Monte Carlo (SMC) samplers running in parallel that can be characterized as the on-line updating portion but with an added global resampling step, that encourages particle diversity and shares information between the samplers. The parallel sequential Monte Carlo samplers allow us to implement a divide-and-conquer strategy, partitioning our data across multiple shards. This allows us to improve computation time and potentially reduce

local memory requirements.

3.3 Background

The method presented in this work is a SMC method that makes use of both approximate and non-approximate particle resampling steps to preserve particle diversity and also runs on a distributed computing system where each computer process or node has it's own shard of data where a batch model is trained. Our approach is unique in that it allows for intermittent communication between the processes/nodes allowing for potential improvement in inference quality.

The chapter is broken as follows: In Section 3.3 we discuss relevant background including Sequential Monte Carlo (SMC). In section 3.4 we briefly discuss previous parallel approaches relevant to our method. In section 3.5 we introduce Communal Monte Carlo and the necessary algorithms for the method. Finally we demonstrate the effectiveness of the approaches in section 3.6 and make concluding remarks and propose future work in Section 3.7.

3.3.1 Sequential Monte Carlo

Sequential Monte Carlo (SMC) methods (Gordon et al., 1993), also known as Particle Filtering, are simulation-based Monte Carlo sampling methods used to solve filtering problems arising in Bayesian inference by represent a posterior distribution with a set of particles (samples). In the general setting, we assume that we have a sequence $y_1, y_2, \ldots$ of observations, each of which is associated with some (set of) latent parameters $\theta_0, \theta_1, \theta_2, \ldots$. We assume the underlying stochastic process can be described in terms of some initial distribution $p(\theta_0)$, a Markovian transition distribution $p(\theta_t|\theta_{t-1})$ and some emission distribution $p(y_t|\theta_t)$ such that the observations y_t are conditionally independent given the θ_t. SMC methods use a swarm of P sequentially constructed "particles" $\{\theta_{0:T}^{(p)}, p = 1, \ldots, P\}$, each associated with a weight $w_t^{(p)}$ to represent the posterior distribution over the $\theta_1, \ldots, \theta_T$.

The simplest SMC algorithm, known as Sequential Importance Sam-

pling (SIS), is a natural extension of importance sampling to the sequential domain. At time t, each particle samples a value $\theta_t^{(p)}$ from some distribution $\pi(\theta_t|\theta_{0:t-1}, y_{0:t})$, and calculating the appropriate importance weights so that

$$w_t^{(p)} = w_{t-1}^{(p)} \frac{p(y_t|\theta_t^{(p)})p(\theta_t^{(p)}|\theta_{t-1}^{(p)})}{\pi(\theta_t^{(p)}|\theta_{0:t-1}^{(p)}, y_{0:t})}.$$

From the perspective of developing parallelizable algorithms, SIS is appealing, since each particle evolves independently from the rest of the swarm. However, as t increases, the particle weights become increasingly skewed, leading to a degenerate swarm that is dominated by a single particle. To avoid this, sequential importance resampling (SIR) periodically refreshes the samples, by sampling with replacement according to the normalized importance weights, resetting the importance weights to one. This allows us to propagate multiple copies of highly weighted particles, encouraging multiple particles to explore high-probability regions and reducing the degeneracy that plagues SIS. The Sequential Importance Resampling Algorithm (Kitagawa, 1996) and (Gordon et al., 1993) is detailed in Algorithm 1.

Algorithm 1 Sequential Importance Resampling (SIR)

Let $X = (X_1, \ldots, X_T)$, P be the number of particles, and $\theta_0 \sim P(\theta_0)$.

1. **for** $t = 1, ..., T$ **do:**
2. **for** $p = 1, ..., P$ **do:**
3. Sample $\theta_t^{(p)} \sim \pi(\theta_t|\theta_{0:t-1}, y_{1:t-1})$
4. Compute weight of particle $\xi_t^{(p)} = P(X_t|\theta_t)$
5. Normalize the weights $\xi_t^{(p)}$ for $p \in (1, ..., P)$
6. Sample particles with replacement $\propto \xi_t^{(p)}$
7. **return** $\theta_t^{(p)}$, the posterior estimate for $P(\theta|X)$

In practice π can be any proposal distribution such as a Gaussian.

While SIR offers improved estimation over SIS, it is no longer easily parallelizable. Resampling must be performed at frequent intervals, and requires access to the entire swarm of particles. While more sophisticated SMC

algorithms have been developed based on SIR (Whiteley et al., 2016), these inherit the global dependency of SIR. In our work, we provide an approximate, parallelizable variant of SIR; while we focus on extending the basic SIR algorithm, our approach could easily be extended to most modern SMC algorithms.

3.3.2 Parallel Monte Carlo Methods and Related Work

As computers have become cheaper, faster and new technologies have facilitated communication between processes or compute nodes, there is a greater potential to take advantage of such "clusters" in a way that allows them to work together in a coordinated fashion. This ability allows for "divide and conquer" strategies. Conceptually, suppose we have observations $X_1, ..., X_N$, the ability to run S separate compute processes in parallel allowing us to fit S separate models to the data. If we then divide the data into S sets, we can make use of all the available processes and fit $f_1, ..., f_s$ separate "mini" models to our S subsets of data. In the final step, we could combine our S models in an appropriate way so as to render an approximation representing a model fitting the full data set of N observations. Efforts to adapt and parallelize SMC algorithms to make use of more sophisticated computational resources and how the approaches differ from our own focus are detailed below.

MCMC Methods where the full data is split into a larger number of smaller datasets which may or may not overlap have been introduced by various researchers (Neiswanger et al., 2014), (Wang and Dunson, 2013), (Scott et al., 2016), (Minsker et al., 2014). Inference is then performed on the data subsets and the model parameters are then combined in some reasonable manner. The Bag of Little Bootstraps (BLB) was proposed by Kleiner et al. (2012) which incorporates features of both the bootstrap and sub-sampling to obtain a robust, computationally efficient means of assessing estimator quality. Divide and conquer ideas previously used to design MCMC algorithms for big data with a sequential MCMC strategy have also been proposed by Casarin et al. (2015a). A stochastic gradient MCMC method that exploits distributed computation by using mini-batches of data greatly decreasing inference time

was proposed by (Ahn et al., 2014) but is different in that it is a fully distributed MCMC algorithm based on stochastic gradients rather than Particle Filtering.

Developing theory and methods of doing SMC in parallel, (Lee and Whiteley, 2016) focusing on controlling Effective Sample Size (ESS) through a method called αSMC. The approach does so in a way that minimizes communication between nodes using tree based graph structures which themselves are based on the structure of the computer cluster/network. The method does not partition the observed data amongst the S compute nodes but rather all the compute nodes see all the data, where as our approach focuses on the scalability in part through data partitioning. In addition their method in practice limits what nodes have access to what particles, where as we allow all nodes access to all particles at global communication points. Methodology proposed by (Nam et al., 2014) in which the number of underlying states, H, in a Hiden Markov Model (HMM) framework can be determined by the use of parallel sequential Monte Carlo samplers. However, the approach does not make use of data partitioning and network communication to improve overall performance or fit quality making the focus of how to leverage distributed computing very different than our own proposed method.

A parallel version of SMC for optimization (PSMCO) by (Akyildiz et al., 2020) optimizes functions of the form $\min_{\theta \in \Theta} f(\theta) = \sum_{i=1}^{n} f_i(\theta)$, by using partitioned subsets of the data on each worker node but then only keeps the particles from the best performing worker and disregards estimates found on other workers. In addition, PSMCO at each iteration will only use a subset of the f_i's on each node; that is, each node will focus on optimizing only a subset of the global $f(\theta)$. Another key difference is that PSMCO transforms the function to be optimized from $\sum_{i=1}^{n} f_i(\theta)$ to one proportional to a probability measure of the form $\exp\left(-\sum_{i=1}^{n} f_i(\theta)\right)$. Our method in turn does nothing of the sort since we use the same probability distribution on all nodes performing inference and in no way requires distributions of the form $c_0 \exp(-\sum_{i=1}^{n} f_i(\theta))$, where c_0 is a constant. Lastly, since PSMCO is a method designed for optimization, the particle with the highest density on the worker that has the highest marginal likelihood is used as the estimate for the global minimum

of $f(\theta)$, where as our focus is on evaluating not only the MAP, but also all relevant moment's for any parameters of interest.

A parallel SMC algorithm proposed by (Verge and Moulines, 2015), named the island particle filter, allows for interaction between nodes, ie "particle islands", according to some resampling scheme. For example, an islands particle history can be replaced according to a draw from a multinomial distribution where each category represents another island's particle history and the likelihood of any one category being drawn is a function of the potential or likelihood function. The particles in each island are then jittered in a mutation step according to a Markov transition. In another proposal, (Verge and Moulines, 2015) also suggests saving the mutation step for situations where the effective sample size of the island falls below some threshold proportional to the number of particles in the island. And in a final proposal, the suggestion to perform the resampling of the islands only after the ESS falls below some threshold that is a function of the number of islands (ie. processors) and individuals (ie. particles) found on each island. These methods requires a global communication at every iteration to determine the ESS across the network, where as our approach relaxes the need for such frequent communication and also accomodates large datasets by partitioning the data across the compute nodes. There have also been contributions to the computational front, such as (Zhou, 2015) who provides a C++ template library that can be used to implement generic sequential Monte Carlo algorithms on parallel hardware using, most notably, MPI and (Casarin et al., 2015b) who developed a matlab package for efficient density combination making use of parallel GPU and multi-core CPU capabilities. (Lindsten et al., 2017) proposed a divide and conquer approach that accommodates paralleling but the the problem is broken up according to the graphical structure of the specified model with various compute nodes responsible for model inference specific to a portion of the graph. Again, our approach to leveraging distributed computing uses data partitioning and parallelization and particle swarms are kept local to each node until a global communication step. On the theoretical front, (Whiteley et al., 2016) were able to phrase sufficient conditions for time-uniform convergence in terms of algorithmic control of the ESS, in turn achievable by adaptive modulation of the interaction between particles. The results even

concern the behavior of PF's run in parallel but fall short of addressing convergence properties in a data partitioned regime. (Míguez and Vázquez, 2016) and (Read et al., 2014) propose using parallel architecture for particle filtering where the particles are partitioned amongst the compute nodes or elements but the observed data is not partitioned.

3.4 Parallel Methods and Consensus Monte Carlo

Increased computational resources has introduced some unique challenges in big data and there have been a variety of proposals in dealing with large datasets. Specifically, (Minsker et al., 2014), (Scott et al., 2016), (Srivastava et al., 2018) and (Neiswanger et al., 2014) to name a few have all suggested an approach reliant on partitioning a full data set into smaller subsets and distributing the data partitions into separate compute processes where sepereate MCMC chains can be run. The challenge comes with the final stage when the subset posteriors need to be combined in some sort of way so as to approximate the full posterior. Since our specific use case is focused on fast scalability, we found that of the aforementioned methods, the one most well suited to fast approximate inference was the Consensus Monte Carlo Method proposed by (Scott et al., 2016). Detailed derivation of the result for our specific use case can be found in Appendix A.1

3.5 Communal Monte Carlo

We now introduce our approximate, distributed online Sequential Monte Carlo algorithm, which we will refer to as Communal Monte Carlo (CMC). We assume that we have a model with time-evolving parameters θ_t, and that given N time-stamped data points (where n_t is the timestamp of the nth data point), we wish to approximate the posterior distribution $p(\theta_{t_n}|X_1, \ldots, X_n)$.

We partition our data across S subsets and assign each dataset to a separate shard. We write $X^{(s)}$ to denote the data in the sth subset. We then run S independent SIR algorithms on the partitioned data up to time t_1. The goal is to obtain S estimates of the posterior $p(\theta_{t_0} \ldots, \theta_{t_1}|\{X_n : n_t \leq$

$t_1\}$). Naively approximating this posterior with $p(\theta_{t_0} \ldots, \theta_{t_1} | \{X_n^{(s)} : n_t \leq t_1\})$ overweights the contribution of the prior (Minsker et al., 2014); instead we use the stochastic approximation

$$\hat{p}(\theta|X) \propto p(\theta|X^{(s)})^S p(\theta)$$

where Minsker et al. (2014) showed that while each sub posterior might be "unstable", in the aggragate, these random measures improve stability and yield smaller credible sets with good coverage properties.

At time t_1, each shard s contains P particles, each representing a parameter value $\theta^{(s,p)}$. We replace each local swarm of particles with P particles sampled, uniformly with replacement, from the set of *all* particles across all S shards. Unlike the SIR steps, this requires global communication. We then propagate our local swarm of particles on the local observations up to the next global communication time t_2. At the final communication time, we then approximate our target distribution with our set of particles using the Consensus Monte Carlo Algorithm proposed in (Scott et al., 2016). The algorithm is summarized in Algorithm 2.

Algorithm 2 Communal Monte Carlo (CMC)

Let $X_{t_1}, ..., X_{t_E}$ be a partition of the full data X, respecting the indexing t. Let $\theta_{t_0}^{(s,p)}$ be our initial set of particles of size SP defined by our prior $P(\theta_0)$.

1. **for** $e = 1, \ldots, E$ **do**
2. Sample w/o replacement from the set $\theta_{t_{e-1}}^{s,p}$ to S sub-sets of size P.
3. Distribute each partition of particles to to each of the S shards.
4. In parallel run SIR Alg. for each shard $s \in 1, ..., S$ with $X_{t_e}^{(s)}$.
5. Combine all S*P particles to from the set, $\theta_{t_e}^{(s,p)}$

return $\theta_{t_E}^{(s,p)}$, a sample from $\prod_{s=1}^{S} P(\theta_{s,E}|X_{s,E}) \approx P(\theta|X)$.

If we have a single global step, ie $t_1 = N_t$, then our algorithm can be seen as a variant of existing embarrassingly parallel algorithms such as those described in 3.3.2 but as we show in section 3.6 there is often advantage in allowing intermittent communication between the parallel processes.

We can think of the global steps as approximations to a global importance resampling operation. Since we have already locally resampled on each shard, the appropriate resampling weights within a shard are uniform. While the relative weights for each shard are not uniform since each shard has a different normalizing constant equal to $\frac{1}{P(X)}$, calculating the true weights would be expensive due to the necessity of evaluating the likelihood for each SP particles on every shard. Therefore, we approximate the weights as equal.

3.6 Real Data and Simulation Studies

We hypothesize that, in scenarios where full SMC-based posterior inference is infeasible due to time, CMC allows us to obtain high-quality estimates of the posterior via parallelization. Further, we argue that where model parameters evolve over time, periodic communication allows for better posterior estimates than would be achieved using an "embarrassingly parallel" approach. To this end, we use our algorithm to perform posterior inference in two Bayesian models, using simulated data. We compare various number of shards (including one shard, ie standard SIR SMC) to explore the trade-off between inference time and estimate quality, and compare various communication frequencies (including an embarrassingly parallel scenario) to explore the benefit of periodic communication. We evaluate the effectiveness of our algorithm using two models including linear and logistic regression for our simulation studies.

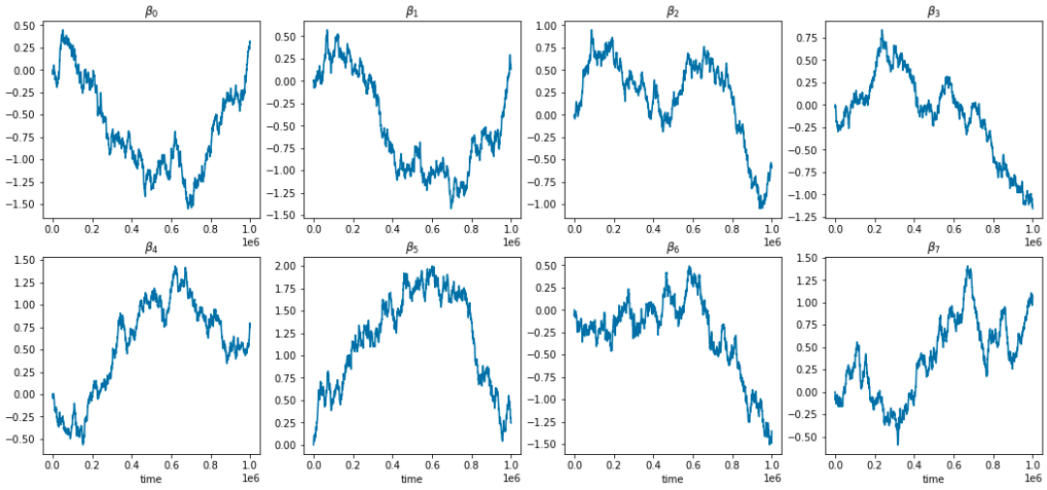
3.6.1 Models

We generate linear and logistic regression datasets where the true coefficients vary according to the Gaussian processes $\beta_k \sim GP(0, K)$, where K is a squared exponential kernel with unit bandwidth. Figrue 3.1 shows some examples of how the β_k vary in time.

3.6.2 Data Sets

Regression: We simulated 20 datasets, each associated with a different set of $K = 32$ dynamically varying parameters β_k. Each dataset contained $N = 1000000$ observations, with one observation per unit-spaced time step, according to equations 3.1:

Figure 3.1: Plot showing 8 of the 32 time varying regression coefficients generated according to $\beta_k \sim GP(0, K)$

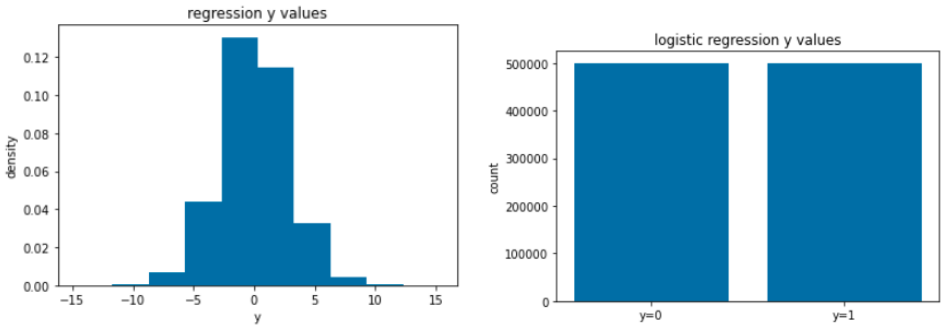


$$
\begin{aligned}
X_{k,t} &\sim \text{Uniform}(-1, 1); \quad k \in \{1, ..., 32\} \\
y_t &= \sum_{k=1}^{32} X_{k,t}\beta_{k,t} + \epsilon_t \\
\epsilon_t &\sim N(0, 1)
\end{aligned} \tag{3.1}
$$

Figure 3.2 shows the range of y values take over all values of t

Logistic Regression (Classification): Our simulated data used in our logistic regression model is generated similarly to the regression data in that the regression coefficients and X follow the same DGP. However, our response $y_t|X_t, \beta_t$ is drawn from a Bernoulli distribution, where we allowed for 10 observations per time stap. We noticed there is some instability in the particle

Figure 3.2: Histogram showing the distribution of y for an example regression data set on the left and distribution of y values for our logistic regression data on the right).



filters ability to learn the model when having only 1 observations per time stamp and sowe elected to allow 10 observations per time stamp. This allows us to batch observations onto the same shard with the same timestamp. We allow for $t \in \{1, ..., 100000\}$ where each time t has 10 corresponding observations $y_{i,t}$, where $i \in \{1, ..., 10\}$ allowing for a total of $N = 1,000,000$ total observations.

We simulated 20 datasets, each associated with a different set of $K = 32$ dynamically varying parameters β_k. Each dataset contained $N = 100000$ observations, with 10 observation per unit-spaced time step, according to equations 3.2:

$$\begin{aligned} X_{i,k,t} &\sim Uniform(-1, 1); \quad k \in \{1, ..., 32\}, \quad i \in \{1, ..., 10\} \\ \gamma_t &= \sum_{k=1}^{32} X_{k,t}\beta_{k,t} \\ p_t &= \frac{1}{1 + \exp(-\gamma_t)} \\ y_{i,t} &\sim Bernoulli(p_t); \quad i \in \{1, ..., 10\} \end{aligned} \tag{3.2}$$

Figure 3.2 shows the range of y values take over all values of t.

3.6.3 Evaluation Metrics

We define the following evaluations metrics for our data sets.

Regression: After fitting a given model, we perform an out-of-sample (OOS) evaluation of the Mean Squared Error (MSE) between the fitted values, $Y_t|X\hat{\beta}_{N_t}$ and the true $E(Y_t|X\beta_{N_t})$, where β_{N_t} is our ground truth value of the regression coefficient determined at the last time value the data is simulated from the Gaussian Process regression coefficients and $\hat{\beta}_{N_t}$ is our estimate of β_{N_t}. We performed the evaluation on 20 separate data set replications to compute the expected Mean Squared Error (MSE), MSE_R, accross the $R = 20$ replications. We generated $N_{oos} = 1000000$ out of sample observations for our test set. Our out-of-sample MSE posterior estimate is therefore,

$$MSE = \frac{1}{N_{oos}} \sum_{n=1}^{N_{oos}} \frac{1}{S} \sum_{s=1}^{S} \frac{1}{P} \sum_{p=1}^{P} (X_n^* \beta_{N_t}^{(s,p)} - X_n^* \beta_{N_t})^2 \tag{3.3}$$

where P is our total number of particles, S is the number of shards, X^* is our out of sample data and N_t is the last time state. We report the mean and standard error of the MSE accross the 20 replications with

$$MSE_R = \sum_{r=1}^{R} \frac{MSE_r}{R} \tag{3.4}$$

Logistic Regression (Classification): After fitting a given model, we performan an OOS evaluation of the mean posterior log likelihood, $E[ll(Y_t|X, \hat{\beta}_{N_t})]$, where $\hat{\beta}_{N_t}$ is our estimate of β_{N_t}. We performed the evaluation on 20 separate data set replications to compute the expected mean log likelihood ELL_R, accross the $R = 20$ replications. We generated $N_{oos} = 10000$ out of sample observations for our test set. Our out-of-sample $E[ll(Y_t|X, \hat{\beta}_{N_t})]$ posterior estimate is therefore,

$$\gamma^{(s,p)} = logit^{-1}(X_n^* \beta_{N_t}^{(s,p)})$$

$$E[ll(Y_t|X, \hat{\beta}_{N_t})] = \frac{1}{N_{oos}} \sum_{n=1}^{N_{oos}} \frac{1}{S} \sum_{s=1}^{S} \frac{1}{P} \sum_{p=1}^{P} y_n \log(\gamma^{(s,p)}) + (1-y) \log(1-\gamma^{(s,p)})$$

(3.5)

where P is our total number of particles and X^* is our out of sample data and N_t is the last time stamp. We report the mean and standard error of the expected mean log likelihood accross the 20 replications with

$$MLL_R = \sum_{r=1}^{R} \frac{E[ll(Y_t|X, \hat{\beta}_{N_t})]_r}{R} \tag{3.6}$$

3.6.4 Regression Comparison of CMC vs. EPSMC

In our evaluations we compare the performance of Communal Monte Carlo with various communication frequencies vs the embarrassingly parallel (one communication at final time stamp) version. For our regression data, our primary comparison metric is out-of-sample Mean Squared Error.

In Figure 3.3 we can see comparisons between our CMC method vs its embarrassingly parallel counterparts with communication frequencies every $100, 1000, 100000, 1000000$ and $S \in \{1, 10, 20, 30, 40, 50\}$ shards and $P = 1000$ particles. Note that communication at 1000000 corresponds to an embarrassingly parallel version of our algorithm. We can see there is the most to gain with frequent communication over the embarrassingly parallel counterpart in this dynamic linear model example. In particular, for a fixed number of shards, we see that more frequent communication produces better fits as is evident with the increasing MSE as communication frequency increases from 100 onward. However, this is not the case for all shard counts. There appears to be little advantage to increased communication for 10 shards as compared to 20,30,40, and 50 shards. In this case, we believe this is because although we lose inference quality with more partitions of the data, we are able to partially overcome this challenge by allowing the paralell processes to share parameter estimates. Additionally, for a fixed number of communication steps, we see that we obtain better performance with fewer shards. This makes sense because each compute node/process sees more data and therefore can produce a better fitting model. On the other hand, fewer shards means longer run times.

Figure 3.3: MSE_R (vertical axis) vs. communication frequency on the left and shards on the right (horizontal axis) . The non-solid lines are the estimated mean and solid bands are show ± 1 standard error. Our base line single machine comparisons are indicated in green

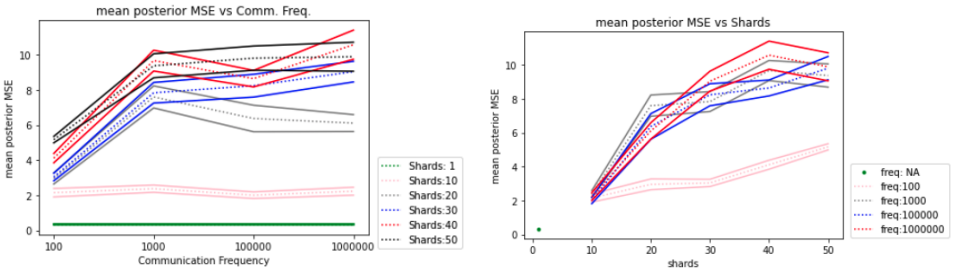
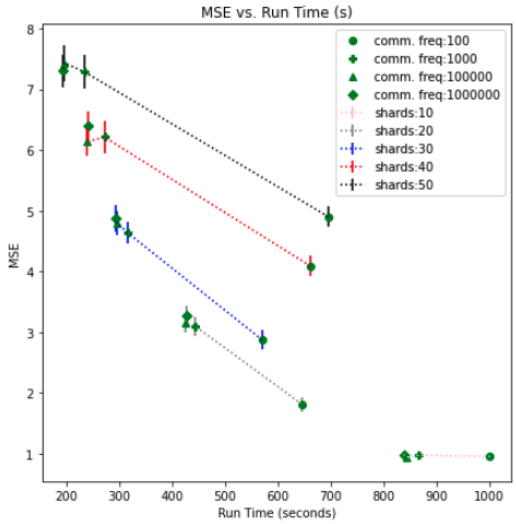


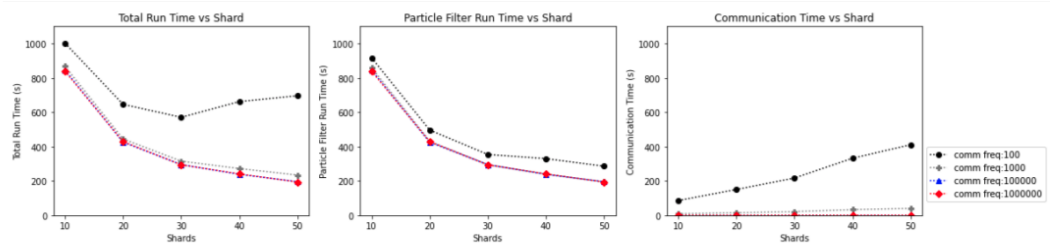
Figure 3.4: MSE_R (y-axis) vs. run time in seconds (x-axis). The ± 1 standard error is indicated by the error bars. Note: the 1 shard run time was approximately 8500 seconds.



Next we consider the computational time. In figure 3.4 we can see the plot of MSE_R (y-axis)vs. total run time in seconds(x-axis). When it comes to run time, it is easy to see that for a fixed number of shards, increasing the communication frequency increases the run time. On the hand, communication comes at a cost that needs to be balanced with the number of shards used. For example, in the 30 shard runs, we see a decrease in computation time as compared to the 40 and 50 shard runs when communication frequency is 100. So, in this specific case, we get a better fit and *faster run time* with the 30 shard runs vs when we use 40 and 50 shards! This is due to the increased cost of communication for increasing numbers of shards. One key takeaway, is it is possible to improve model fit at the expense of some computational time as would be expected but even with frequent communication, we can sometimes get the benefits of both decreased computational time and improved fit.

Figure 3.5 details the relationship between total run time, particle filter run time, and communication time for varying communication frequencies and shard counts. We can see in the plot of particle filter run time vs shars, the run time relationship is roughly proportional to $\frac{1}{s}$, where s is the shard count but in the plot of communication time vs shards, we see a relationship that is roughly linear. For a relatively frequent communication regime, a frequency of 100, in this case, the sum of particle filter run time and communication time will result in a smallest total run time that seems to occur around 30 shards which explains how we can get higher total run times for 40 and 50 shards even though each shard is processing less data than the 30 shard version.

Figure 3.5: time in seconds (y-axis) vs. shards (x-axis)



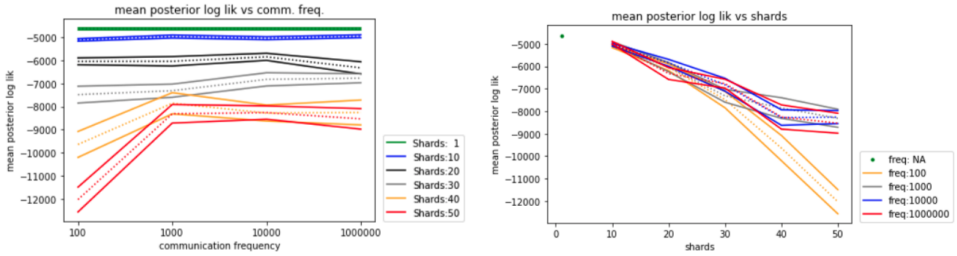
3.6.5 Logistic Regression Comparison of CMC vs. EPSMC

Next we show the results for our classification (logistic regression) model with 1000000 observations in total and where each time stamp has 10 observations for a total of $\frac{1000000}{10} = 100000$ unique time values. This allows us to batch our observations with one single machine getting all 10 observations with the same time stamp. It would be possible to distribute the observations with the same time stamp on separate machines but since our goal in part is to improve time efficiency, we elected to batch the observations with the same timestamp on the same machine.

In Figure 3.6 we can see comparisons between our CMC method vs its embarrassingly parallel counterparts with communication frequencies every $100, 1000, 10000, 1000000$ and $S \in \{1, 10, 20, 30, 40, 50\}$ shards and $P = 1000$ particles. Note that communication at 1000000 corresponds to an embarrassingly parallel version of our algorithm. We can see there appears to be no improvement in log likelihood for the runs with 10, 20, and 30 shards. this implies to us the processes running in the embarrassingly parallel setting have enough data to learn model parameters and do not need the "advantage" of sharing information amongst themselves. Conversely, we see a collapse in the model fit for the 40 and 50 shard runs when allowing frequent communication. We suspect this is in part due to particle impoverishment but also due to dominant particles that have become "lost" in their parameter space exploration and yet managed to remain relatively dominant long enough to "pollute" the other processes leading the entire fitting process to perform poorly. A second compounding possibility is we were not able to select appropriate tuning parameters that would allow the separate processes to adapt to the fact that they are getting data that is more separated in time which coupled with the relatively weak signal from y values that are either 0 or 1 could also lead to the poor performance.

Next we consider the computational time. In Figure 3.7 we can see the plot of mean posterior log likelihood (y-axis) vs. run time in seconds(x-axis). When it comes to run time we see a similar patter to what we observed with the regression data but with run times occurring at roughly one tenth the time as would be expected since we are batching our observations in groups of 10 on each machine due to there being 10 observations per time stamp.

Figure 3.6: MLL_R (vertical axis) vs. communication frequency on the left and shards on the right (horizontal axis) . The non-solid lines are the estimated mean and solid bands are show ± 1 standard error. Our base line single machine comparisons are indicated in green



Additionally, even though the fit quality decreases as we increase the number of shards, we can dramatically reduce run time roughly proportional to $\frac{1}{s}$, where s is the number of shards. The practitioner, this is something to consider where execution time is critical. Lastly, in Figure 3.8, we see the same general patterns observed in the regression data with total run time, particle filter run time, and communication time. We see the particle filter run times roughly proportional to $\frac{1}{s}$, where s is the shard count but in the plot of communication time vs shards, we see a relationship that is roughly linear. For a relatively frequent communication regime (frequency of 100, in this case), the sum of particle filter run time and communication time will result in a minimum total run time that seems to occur around 30 shards which explains how we can get higher total run times for 40 and 50 shards even though each shard is processing less data than the 30 shard version.

Figure 3.7: MLL_R (y-axis) vs. run time in seconds (x-axis). The ± 1 standard error is indicated by the error bars. Note: the 1 shard run time was approximately 1500 seconds.

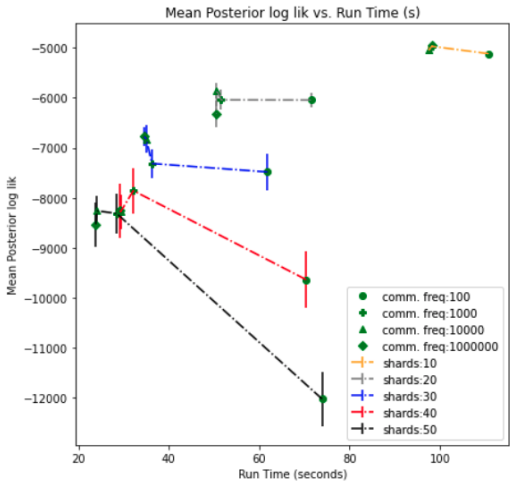
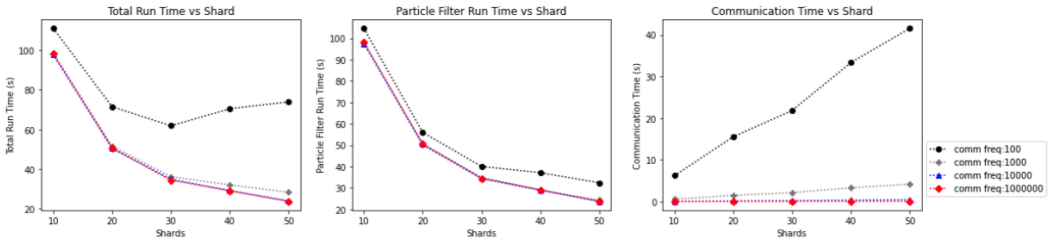


Figure 3.8: time in seconds (y-axis) vs. shards (x-axis)



3.6.6 Computational Resources

We fit our models to the simulated data using Stampede2, the flagship super computer of the National Science Foundation hosted by Texas Advanced Computing Center (TACC). It allows for massively parallel compute jobs and facilitates communication between compute nodes and up to 64 processes to run in parallel per node. In our experiments, we allowed for up to 50 parallel jobs.

3.7 Discussion and Conclusion

We have seen that Communal Monte Carlo has the capacity to improve the fit of models where pluralization is used. In particular where there are many parallel processes with partitioned data. We did not show in our analysis results where there was no benefit. We noticed that for small numbers of parallel processes consistent with what you have on a local laptop, such as 4 to 6 processes, there is no gain in fit performance. We also noticed that for the particular tuned settings we chose for our model fitting, above 50 shards for regression and 40 shards for classification, there also seemed to be a loss of advantageous performance with frequent communication doing as well as embarrassingly parallel. We have also seen in models that are less "hard" to learn, that increased communication does as well as the embarrassingly parallel counterpart.

The current work opens the possibility of improving model fitting with relatively simple communication strategies. We suspect that for more complex models such as ensemble tree methods where two functions are being fit (see Chapter 4 as an example), when the models are sufficiently hard to learn there is the possibility of improving tree ensemble methods by allowing communication between chains running in parallel. The objective obviously would be to "help" poor performing chains with trees that are stuck in a relatively low likelihood to find the part of the space with higher likelihood. With a theoretical backdrop, we would have an illuminated road map of where to find opportunities for increased parallelization. Rather, what we have shown is it is possible to significantly improve model performance with little time cost with

the use of very simple communication strategies in a parallel environment. Other possibilities include learning communication frequencies that improve model fit. Lastly, modifying CMC to behave like a gossip algorithm could also be useful in circomstances where there multiple processes are running in parallel on nodes running in parallel. In this setting, we could allow for local node communication with higher frequency than global communications between nodes.

Chapter 4

Bayesian Additive Regression Trees Mixture Model for Heterogeneous Causal Effects

4.1 Overview

In this section we propose a Bayesian semi-parametric model with a Gaussian mixture distribution in the residual term to model heterogeneous treatment effects. The model revolves around the transformed response variable (TRV) approach from Athey and Imbens (2015a), which is a reformulation of the response variable used to model heterogeneous treatment effects. We aim to address 2 objectives.

1. Explicitly model heterogeneous treatment effects with a distribution implied by the TRV, thereby reducing the variance of the estimators.
2. Model both the treatment and control conditions jointly, thereby improving inference over models that do not.

4.2 Introduction

Treatment effects are typically estimated as an average treatment effect (ATE) which is a measure of the average difference between units assigned to treatment versus units assigned to control. The ATE is particularly useful when there is a lack of heterogeneity in the treatment effect but work from Heckman et al. (2006), Green and Kern (2012), Xie et al. (2012) suggest this is less common than a population exhibiting heterogeneous treatment

effects. An obvious downside to the ATE in the face of heterogeneity is the ATE will fail to estimate treatment effects in subgroups of the population should they exhibit heterogeneity. Importantly, we can deal with heterogeneity with a different approach and estimate the Conditional Average Treatment Effect (CATE) which is the average treatment effect conditional on the unit's covariates. There have been both parametric and non-parametric methods proposed to estimate the CATE.

Parametric models of note used to estimate the conditional mean outcomes include linear and polynomial regression (Pearl, 2009) and penalized regression approaches (Tibshirani, 1996). Alternative non-parametric approaches have also been proposed to estimate heterogeneous treatment effects such as boosting (Powers et al., 2018), modified versions of Bayesian Additive Regression Trees (BART) (Hill, 2011), (Hahn et al., 2019), (Chipman et al., 2010) as well as more machine learning oriented regression tree methods (Athey and Imbens, 2015a), (Breiman et al., 1984b) and ensemble methods (Foster et al., 2011), (Wager and Athey, 2018), (Breiman, 2001), (Athey and Imbens, 2015b). The above methods have their strengths but are not without limitation which we discuss below. Estimating the CATE as proposed by Wager and Athey (2018) using a random forest has sound theoretical justification due to the valid probabilistic statistical inference properties and tends to outperform other classical methods, particularly in the presence of irrelevant predictors. This approach, however has been outperformed by competing methods such as Bayesian Causal Forest by (Hahn et al., 2019). A second downfall of the random forest approach is due to the lack of a procedure to perform some regularization, random forests are a challenge to use for heterogeneous treatment effect estimation (Wendling et al., 2018). On the other hand BART based methods modified for estimating heterogeneous causal effects have been shown to perform quite well (Hahn et al., 2019); (Hill, 2011). The down side of the modified BART methods is the limited work related to their theoretical inferential properties related to CATE estimation when compared to (Wager and Athey, 2018).

A second option for modeling the difference in conditional mean functions is to use the transformed response variables (TRV) (Dudík et al., 2014); (Beygelzimer and Langford, 2016) which has a similar flavor to inverse probability (propensity score) weighting (IPW) (Hirano et al., 2003). The TRV

approach introduces a new response variable that is a function of the observed outcomes, Y, the binary treatment assignment variable, W, and the probability of treatment assignment (propensity score), p. The expected value of the TRV conditional on the covariates, X, is the CATE. This property in particular allows the TRV to be used with "off the shelf" machine learning and other regression related methods since it only requires a single regression. Recent work by Athey and Imbens (2015a) with the TRV showed how one could model the CATE directly using the TRV as a function of the observed covariates with regression trees. This approach comes with a serious cost in that the estimation quality is poor due to the high variance resulting the TRV without propper adjustmet accounting for treatment level. Zaidi and Mukherjee (2018) introduced a reparameterization that explicitly accounts for the implied mixture of the TRV with a correctly specified model significantly reducing the variance previously associated with the TRV. **?** use a Gaussian processes mixture to model both the source of the heterogeneity, and the response functions under both treatment conditions. This approach serves to produce better estimates of the CATE by accounting for the mixture nature implied by the TRV as we discuss in section 4.4.2. While Gaussian processes are quite robust and performant, they come with the need for considerable tuning and computational resources as compared to BART. Thus, we use the same reparameterization introduced by Zaidi and Mukherjee (2018) and a modified version of BART to introduce a new method, Causal BART Mixture Model (CBARTMM), to estimate the CATE.

4.3 Framework and Assumptions

Our model is designed to addresses the scenario where we have data from an experiment where a binary variable, $W \in \{0, 1\}$, is assigned randomly, or in an observational setting that satisfies the assumptions laid out in (Imbens and Rubin, 2015) which are required to make valid statistical inference of the CATE. We state these assumptions next.

- **Assumption 1:** Individualistic assignment : *The probability a unit is assigned to the active treatment does not depend on the covariates or*

potential outcomes of the other units.

- **Assumption 2:** Probabalistic assignment:

$$0 < P(W = 1) < 1 \tag{4.1}$$

- **Assumption 3:** Unconfounded assignment:

$$Y(1), Y(0) \perp\!\!\!\perp W|X \tag{4.2}$$

- **Assumption 3:** The Stable Unit Treatment Value Assumption (SUTVA): *The potential outcomes for any unit do not vary with the treatments assigned to other units, and, for each unit, there are no different forms or versions of each treatment level, which lead to different potential outcomes.*

Let N be the number of units in the data, indexed by $i \in \{1, .., N\}$. Let $w_i \in \{0, 1\}$, be the binary indicator for the treatment, with $w_i = 0$ for units assigned to the control arm and $w_i = 1$ for units assigned to the treatment arm, and let X_i be a p-component vector of features, covariates or pre-treatment variables that are unaffected by treatment. Let $p = E[W_i] = P(W = 1)$ be the marginal probability of treatment assignment and $p_i = E[W_i|X_i] = P(W = 1|X_i)$ be the conditional probability of treatment assignment, also called the "propensity score".

We postulate that there exists a pair of potential outcomes for each unit $(Y_i(0), Y_i(1))$, following the Potential Outcomes Framework (POF) described in (Imbens and Rubin, 2015). We therefore define the unit-level causal effect τ_i as the difference in potential outcomes,

$$\tau_i = Y_i(1) - Y_i(0) \tag{4.3}$$

The observable outcome for unit i is the potential outcome linked to the outcome conditional on the treatment, W_i. That is,

$$Y_i = Y_i(W_i) = \begin{cases} Y_i(0), \text{ if } W_i = 0 \\ Y_i(1), \text{ if } W_i = 1 \end{cases} \tag{4.4}$$

The data consists of the triplet (Y_i, W_i, X_i), for $i \in \{1, ..., N\}$. We define the conditional average treatment effect (CATE), as

$$\tau(x) = E[Y_i(1) - Y_i(0)|X_i = x] \tag{4.5}$$

and the population average treatment effect as

$$\tau^{ATE} = E_X\left[E_Y[Y(1) - Y(0)|X]\right] = E\left[\tau(x)\right]. \tag{4.6}$$

4.4 Transformed Response Variable for Causal Inference

If we denote our observed response value Y_i, our indicator variable indicating treatment assignment as W_i and the probability of treatment assignment (propensity score) as $P(W_i = 1|X_i) = p_i$ then the transformed response variable is defined as:

$$Y_i^* = \frac{Y_i(W_i - p_i)}{p_i(1 - p_i))} \tag{4.7}$$

This definition has the property that $E[Y_i^*|X_i = x] = \tau(x)$. We provide the proof in Appendix B.1.

4.4.1 Advantages of the TRV

The TRV comes with a set of two main advantages. First, modeling heterogeneous treatment effects with TRV (Athey and Imbens, 2015a),(Wager and Athey, 2018),(Powers et al., 2018) can be done directly with machine learning (ML) techniques. Specifically, the TRV allows us to model the unobserved value, τ with a single regression, $E[Y_i^*|X_i = x]$. Second, unlike most causal modeling approaches that focus on estimating $E[Y(1) - Y(0)|X]$ by modeling $E[Y(0)|X]$ and $E[Y(1)|X]$ the TRV uses the propensity score as defined in equation (4.7).

4.4.2 Disadvantages of the TRV

As appealing as the TRV is, it does not come without downsides. First, the causal estimand has high variance ((Athey and Imbens, 2015a); (Powers et al., 2018)). This is due to the fact that

$$Var(Y_i^*) = \frac{\sigma_1^2}{p_i} + \frac{\sigma_0^2}{1-p_i} + \frac{(1-p_i)\mu_1^2}{p_i} + \frac{p_i\mu_0^2}{1-p_i} - 2\mu_1\mu_0 \tag{4.8}$$

where $\mu_0 = E[Y(0)|X = x_i]$, $\mu_1 = E[Y(1)|X = x_i]$, $\sigma_0^2 = Var[Y(0)|X = x_i]$ and $\sigma_1 = Var[Y(1)|X = x_i]$. See B.1 for proof. We can clearly see that $Var(Y_i^*)$ varies according to an inverse relationship with the propensity score and directly with the square of the conditional means of μ_0 and μ_1. This is a problem because if the propensity score is near 0 or 1 or if μ_0 or μ_1 is large in magnitude, $Var(Y_i^*)$ blows up. Figure 4.1 illustrates the relationships with a few examples. In addition to the variance having the issue of blowing up as previously mentioned, in Figure 4.2, we can also see two distinct distributions. This implies modeling Y_i^* as a mixture of two distributions makes sense, since we would otherwise be modeling the TRV with a single regression function.

A second downside, until recently, has been the lack of valid uncertainty associated with the estimators. As can be seen in Figure 4.1, there is a dramatic increase in variability for propensity scores that are large or small. This limitation adds to the concern that the large variance can lead to poor interval estimation for the CATE as well as the ATE. Monte Carlo methods to estimate the uncertainty in the CATE have been demonstrated for use in Knaus et al. (2020) but Wager et al. (2014) has also shown that in some situations the Monte Carlo error can dominate the uncertainty quantification estimation. Because of this, in situations where the treatment effect is small, the conflation of the Monte Carlo error with the sampling error can lead to overstating the variance and hence lower the quality of the analysis. Additionally, Powers et al. (2018) showed that when estimating the ATE, the variance versus absolute main effect (σ^2 vs. $\frac{\mu_0+\mu_1}{2}$) of the TRV (as compared to modeling the ATE with the difference of two conditional means) grows with increasing sample size when all the observations have propensity score $p_i = \frac{1}{2}$;

Figure 4.1: An example of total variance,

$$Var(Y_i^*) = \frac{\sigma_1^2}{p_i} + \frac{\sigma_0^2}{1-p_i} + \frac{(1-p_i)\mu_1^2}{p_i} + \frac{p_i\mu_0^2}{1-p_i} - 2\mu_1\mu_0$$

where $\sigma_0 = \sigma_1 = 1$, $\mu_0 = -1$ and $\mu_1 \in (-3, -2, -1, 0, 1, 2, 3)$

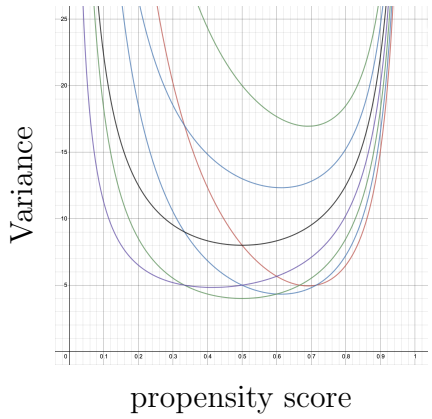
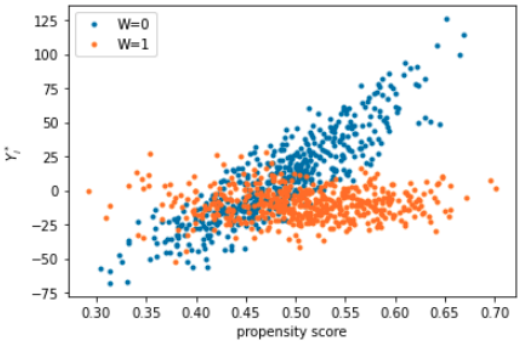


Figure 4.2: An example of the Y_i^* for two treatment levels. Notice the disting behavior when $W = 0$ verses when $W = 1$, implying the mixture nature of the TRV.



hence, working against methods like bootstrapping which performs better with increasing sample size.

It is because of the above limitations of the TRV Zaidi and Mukherjee (2018) proposed a reparameteraziation of the model to estimate heterogeneous treatment effects by accounting for the implied mixture distribution of the TRV. We also used this same representation to introduce a modified version of BART which comes with the benefits of estimated posterior distributions for the CATE at the unit level. Additionally, the model representation allows us to reduce the variance of the CATE estimates while still modeling the complex heterogeneous relationship of the covariates with the unit level treatment effect.

4.5 The Causal Bayesian Additive Regression Trees Mixture Model (CBARTMM)

4.5.1 The Model

Let us consider two regression functions for a response under two different conditions, treatment and control

$$\begin{aligned} \mathrm{Y}_i(1) &= \mu_1(x_i) + \epsilon_i(1), \text{ where } \epsilon_i(1) \sim N(0, \sigma^2) \\ \mathrm{Y}_i(0) &= \mu_0(x_i) + \epsilon_i(0), \text{ where } \epsilon_i(0) \sim N(0, \sigma^2). \end{aligned} \tag{4.9}$$

Our estimand of interest is

$$\tau(x_i) = \mu_1(x_i) - \mu_0(x_i) \tag{4.10}$$

If we substitute the regression functions under the treatment and control conditions into the definition of the TRV, we get the following mixture distribution,

$$\begin{aligned} Y_i^* &= \tau(x_i) + \epsilon_i^* \\ \epsilon_i^* \sim p_i N\left((1-p_i)h(x_i), \frac{1}{p_i^2}\sigma^2\right) &+ (1-p_i) N\left(-p_i h(x_i), \frac{1}{(1-p_i)^2}\sigma^2\right) \\ \tau(x_i) &= \mu_1(x_i) - \mu_0(x_i) \\ h(x_i) &= \frac{\mu_1(x_i)}{p_i} + \frac{\mu_0(x_i)}{1-p_i} \end{aligned} \tag{4.11}$$

In equation 4.11, $\tau(x_i)$ can be interpreted as the conditional average treatment effect, and $h(x_i)$ forms the mean of the heterogeneous residuals for the i^{th} observation. A detailed derivation of the model in 4.11 is available in appendix B.2

According to (Zaidi and Mukherjee, 2018) the reason this formulation is superior to directly modeling μ_1 and μ_0 in equation 4.9, is because the conditional distributions of $[Y_i|X_i, W = 0]$ and $[Y_i|X_i, W = 1]$ may be hard to estimate due to lack of overlap in X for the two treatment levels. Additionally, (Powers et al., 2018) also argued that because μ_0 and μ_1 are never estimated perfectly, ignoring the propensity score is a potential source of bias when estimating the CATE. In addition, (Hahn et al., 2019) points out that placing individual vague priors on the conditionals for Y_i make it a challenge for a model to capture the heterogeneity present in the data due to the implied priors on $(Y_i|X_i, W = 0) - (Y_i|X_i, W = 1)$ being potentially too vague.

4.5.2 Model Specification

We define our model as follows:

$$\begin{aligned} Y_i^* &= \tau(x_i) + \epsilon_i^* \\ \epsilon_i^* &\sim p_i N((1-p_i)h(x_i), \frac{1}{p_i^2}\sigma^2) + (1-p_i)N(-p_i h(x_i), \frac{1}{(1-p_i)^2}\sigma^2) \\ \mu_{j,b}^{(\tau)} &\sim N(\mu_0^{(\tau)}, \sigma_\tau^2) \\ \mu_{j,b}^{(h)} &\sim N(\mu_0^{(h)}, \sigma_h^2) \\ \sigma^2 &\sim \text{IG}(a, c) \end{aligned} \tag{4.12}$$

We model $\tau(x)$ and $h(x)$ using two BART specifications,

$$\begin{aligned} \tau(x) &= \sum_{m=1}^{M_\tau} T_{\tau,m}(x) \\ h(x) &= \sum_{m=1}^{M_h} T_{h,m}(x) \end{aligned} \tag{4.13}$$

where $T_{.,m}$ is a single binary tree in the sum of trees, M_τ and M_h are the number of trees used to model $\tau(x)$ and $h(x)$, respectively. For each binary

tree, $T_{.,m}$ there are an associated set of leaf nodes, $\mathcal{M}_m = \{\mu_{m,1}^{(.)}, ..., \mu_{m,b}^{(.)}\}$ which denote a set of parameter values associated with each of the b terminal nodes of $T_{.,m}$. Following Chipman et al. (2010), we define our set of regularization priors in order to maintain the rich structure of 4.13 and prevent our trees from over fitting. We place a BART prior on the tree structures and the leaf nodes as follows:

$$\begin{aligned} p((T_1^{(\tau)}, \mathcal{M}_1^{(\tau)}), ..., (T_{M_\tau}^{(\tau)}, \mathcal{M}_{M_\tau}^{(\tau)}), (T_1^{(h)}, \mathcal{M}_1^{(h)}), ..., (T_{M_h}^{(h)}, \mathcal{M}_{M_h}^{(h)}), \sigma) \\ &= \Big[\prod_j p(T_j^{(\tau)}, \mathcal{M}_j^{(\tau)})\Big]\Big[\prod_j p(T_j^{(h)}, \mathcal{M}_j^{(h)})\Big] p(\sigma) \\ &= \Big[\prod_j p(\mathcal{M}_j^{(\tau)}|T_j^{(\tau)}) p(T_j^{(\tau)})\Big]\Big[\prod_j p(\mathcal{M}_j^{(h)}|T_j^{(h)}) p(T_j^{(h)})\Big] p(\sigma) \end{aligned} \tag{4.14}$$

where

$$\begin{aligned} p(\mathcal{M}_j^{(\tau)}|T_j^{(\tau)}) &= \prod_i p(\mu_{i,j}^{(\tau)}|T_j^{(\tau)}) \\ p(\mathcal{M}_j^{(h)}|T_j^{(h)}) &= \prod_i p(\mu_{i,j}^{(h)}|T_j^{(h)}) \end{aligned} \tag{4.15}$$

where $\mu_{i,j}^{(\tau)} \in \mathcal{M}_j^{(\tau)}$ and $\mu_{i,j}^{(h)} \in \mathcal{M}_j^{(h)}$.

4.5.2.1 Priors on the leaf node parameters

Unlike the rescaling of the response variable in Chipman et al. (2010), we do not scale the response variable Y_i^* to the interval $[-\frac{1}{2}, \frac{1}{2}]$ since doing so makes τ and h unidentifiable. We make use of the data to select the hyperparameters. For a full discussion on selecting hyperparmeters for the leaf node and variance parameters, see (Chipman et al., 2010). We specify the hyperparameters $\mu_{i,j}^{(\tau)}$, $\mu_{i,j}^{(h)}$, σ_τ^2, σ_h^2, a and b in Equation 4.12 in a manner analogous to Chipman et al. (2010).

$$
\begin{aligned}
\mu_0^{(\tau)} &= \frac{\overline{Y_i^*}}{M_\tau} \\
\mu_0^{(h)} &= \frac{1}{M_h}\left(\frac{\sum_{i:W_i=1}\frac{Y_i}{p_i}}{\sum_i W_i} + \frac{\sum_{i:W_i=0}\frac{Y_i}{1-p_i}}{\sum_i(1-W_i)}\right) \\
\sigma_\tau &= \frac{0.5\left(\max_i(Y_i^*) - \min_i(Y_i^*)\right)}{k\sqrt{M_\tau}} \\
\sigma_h &= \frac{0.5\left(H_{upper} - H_{lower}\right)}{k\sqrt{M_h}} \\
a &= 3
\end{aligned}
\tag{4.16}
$$

and b is chosen such that $P(\sigma < \hat{\sigma}) = 0.99$. where $\hat{\sigma}$ is the sample standard deviation of Y_i^*,

$$
H_{upper} = \max_{i:W_i=1}\left(\frac{Y_i}{p_i}\right) + \max_{i:W_i=0}\left(\frac{Y_i}{1-p_i}\right)
$$

and

$$
H_{lower} = \min_{i:W_i=1}\left(\frac{Y_i}{p_i}\right) - \min_{i:W_i=0}\left(\frac{Y_i}{1-p_i}\right).
$$

The intuition is that we want to shrink the mean functions, τ and h toward the sample average means of their respective quantities, in the way Chipman et al. (2010) shrinks the mean function toward zero after rescaling to the interval $[-\frac{1}{2}, \frac{1}{2}]$. Similarly, the prior standard deviations are chosen such that when $k = 2$, we place 0.95 prior probability that $E(\tau|x)$ is in the interval $(\min(Y_i^*), \max(Y_i^*))$ and that $E(h|x)$ is in the interval (H_{lower}, H_{upper}), with 0.95 prior probability.

4.5.2.2 Priors on tree structure

We place a standard BART prior on our trees $T_j^{(\tau)}$ and $T_j^{(h)}$. The BART prior involves three components:

1. The probability that a node at depth $d \in (0, 1, 2, ...)$ is non-terminal given by

$$
\frac{\alpha}{(1+d)^\beta} \tag{4.17}
$$

where $\alpha \in (0, 1)$, $\beta \in [0, \infty)$.

2. The distribution on the splitting variable assignments at each interior node. Specifically, the uniform prior on available variables.

3. The distribution on the splitting rule assignment in each interior node, conditional on the splitting variable. Specifically, the uniform prior on the discrete set of available splitting values.

We set our $\alpha = 0.95$ and $\beta = 2$ in our examples in order to maintain a non overly restrictive shallow tree depth. Some applications of BART have used different prior settings; for example, Hahn et al. (2019) set $\alpha = 0.25$ and $\beta = 3$ for the function estimating $\tau(x)$, arguing that the resulting shallower tree depth leads to better estimation of the treatment effect, since it is typically slow-varying. However, since we did not find any consistent improvement using alternative parameter settings, we choose to use the default BART priors.

4.5.2.3 Selecting the number of trees

Choosing M_τ and M_h, the number of trees in the model for $\tau(x)$ and $h(x)$ respectively can be determined in several ways. In principal, we could place priors on M_τ and M_h and infer them in a fully Bayesian manner; however we discount this option as it would lead to much higher computational costs. Having too few trees can lead to under fitting, but does come with the advantage of lower computational costs and therefore faster model fitting. On the other hand, allowing too many trees can lead to over fitting. Further, convergence of the MCMC is slower with more trees, due to the more complex state space; while the MCMC algorithm is asymptotically exact, this can mean that the algorithm does not converge in a reasonable time frame. We found, for the example data sets we used to validate the model, $M_\tau = M_h = 25$ worked well consistently and did not require overly long periods of time to fit the model.

4.6 Posterior Sampling

Inference for our model requires sampling from the posterior distribution using Bayesian back-fitting similar to that of Chipman et al. (2010). This is an iterative procedure for fitting additive models such as BART. We must modify the BART Bayesian back fitting algorithm since we are working with two functions, and since the distribution at each leaf node is no longer a straightforward, homogeneous Gaussian. We summarize the resulting algorithm below. See Appendix B.3 for all relevant derivations related to posterior sampling.

4.6.1 Basic outline of sampler

Since BART is an additive model and is fit via Bayesian back fitting, we begin by defining the residuals for the j^{th} tree of the function $\tau(x)$. Combining equations 4.12 and 4.13 we can solve for the residuals on the j^{th} tree. Let

$$\begin{aligned} R^{(\tau)}_{-j,i} &= Y^*_i - \sum_{m \neq j}^{M_\tau} T^{(\tau)}_m(x_i) - (w_i(1-p_i)h(x_i) - (1-w_i)p_i h(x_i)) \\ \mathbf{R}^{(\tau)}_{-j} &= [R^{(\tau)}_{-j,1}, ..., R^{(\tau)}_{-j,N}]^T \\ R^{(\tau)}_{-j,i} &\sim N\left(\mu^{(\tau)}_{j,b}, w_i \frac{\sigma^2}{p_i^2} + (1-w_i)\frac{\sigma^2}{(1-p_i)^2}\right). \end{aligned} \tag{4.18}$$

Similarly, we define the the residuals for the j^{th} tree of the function $h(x)$. Let

$$\begin{aligned} R^{(h)}_{-j,i} &= Y^*_i - \tau(x_i) - \left(w_i(1-p_i) - (1-w_i)p_i\right) \sum_{m \neq j}^{M_h} T^{(h)}_m(x_i) \\ \mathbf{R}^{(h)}_{-j} &= [R^{(h)}_{-j,1}, ..., R^{(h)}_{-j,N}]^T \\ R^{(h)}_{-j,i} &\sim N\left(\left(w_i(1-p_i) - (1-w_i)p_i\right)\mu^{(h)}_{j,b}, w_i \frac{\sigma^2}{p_i^2} + (1-w_i)\frac{\sigma^2}{(1-p_i)^2}\right). \end{aligned} \tag{4.19}$$

Similarly, the residuals of the model using all trees is

$$\begin{aligned}
\mathcal{E}_i &= Y_i^* - \tau(x_i) - \left(w_i(1-p_i) - (1-w_i)p_i \right) h(x_i) \\
\mathcal{E} &= [\mathcal{E}_1, ..., \mathcal{E}_N]^T \\
\mathcal{E}_i &\sim N\left(0, w_i \frac{\sigma^2}{p_i^2} + (1-w_i)\frac{\sigma^2}{(1-p_i)^2}\right).
\end{aligned} \tag{4.20}$$

The sampler
The sampler alternates between fitting the trees for τ, followed by fitting the trees for h, followed by the draw for σ.

Sampling the trees and nodes for $\tau(x)$, we iterate through all $j = 1, ..., M_\tau$ trees

- $T_j^{(\tau)} | \mathbf{R}_{-j}^{(\tau)}, \sigma^2, h(x)$
- $\mathcal{M}_j^{(\tau)} | T_j^{(\tau)}, \mathbf{R}_{-j}^{(\tau)}, \sigma^2, h(x)$.

Sampling the trees and nodes for $h(x)$, we iterate through all $j = 1, ..., M_h$ trees

- $T_j^{(h)} | \mathbf{R}_{-j}^{(h)}, \sigma^2, \tau(x)$
- $\mathcal{M}_j^{(h)} | T_j^{(h)}, \mathbf{R}_{-j}^{(h)}, \sigma^2, \tau(x)$.

Finally, we sample σ^2

- $\sigma^2 | \mathcal{E}$.

4.6.2 Sampling the leaf node parameters:

In this subsection we present the conditional distributions used in posterior sampling. Detailed derivations are available in Appendix B.3.

The conditional distribution of the leaf nodes is given by

$$\mu_{i,j}^{(\delta)}|\sigma^2, T_j^{(\delta)}, y, \sim N\left(\frac{\sum_{i\in n_{j,b}^{(\delta)}} \frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2} + \frac{\mu_0^{(\delta)}}{\sigma_\delta^2}}{\frac{1}{\sigma_\delta^2} + \sum_{i\in n_{j,b}^{(\delta)}} \frac{1}{\sigma_{\delta,i}^2}}, \Big[\frac{1}{\sigma_\delta^2} + \sum_{i\in n_{j,b}^{(\delta)}} \frac{1}{\sigma_{\delta,i}^2}\Big]^{-1}\right) \tag{4.21}$$

When sampling the leaf nodes for τ, we define $\delta = \tau$, $n_{j,b}^{(\delta)}$ is the set of indices in node b in $T_j^{(\tau)}$ and

$$\begin{aligned} z_i^{(\tau)} &= R_{-j,i}^{(\tau)} \\ \sigma_{\tau,i}^2 &= w_i \frac{\sigma^2}{p_i^2} + (1-w_i)\frac{\sigma^2}{(1-p_i)^2}. \end{aligned} \tag{4.22}$$

When sampling the leaf nodes for h, we define $\delta = h$, $n_{j,b}^{(\delta)}$ is the set of indices in node b in $T_j^{(h)}$ and

$$\begin{aligned} z_i^{(h)} &= \Big(w_i(1-p_i) - (1-w_i)p_i\Big)^{-1} R_{-j,i}^{(h)} \\ \sigma_{h,i}^2 &= \frac{\sigma^2}{p_i^2(1-p_i)^2}. \end{aligned} \tag{4.23}$$

Finally, the conditional distribution of σ^2 is given by

$$\sigma^2 \sim IG\left(\frac{n}{2} + a, c + \frac{1}{2}\sum_{i=1}^{n}\left(w_i p_i^2 + (1-w_i)(1-p_i)^2\right)\mathcal{E}^2\right) \tag{4.24}$$

where σ^2 has an IG(a, c) prior.

4.6.3 Sampling the Tree Structures

We adapt the "grow" and "prune" steps in Chipman et al. (2010) as described below. The portion of the likelihood associated with node b in tree $T_j^{(\delta)}$ is given as follows:

$$
\begin{aligned}
L(Y_i^*; \sigma^2, T_{j\notin M}^{(\delta)}, T^{(\gamma)}) &= \int_{-\infty}^{\infty} L(Y_i^* | \mu_{j,b}^{(\delta)}, \sigma^2, T_{j\notin M_\delta}^{(\delta)}, T^{(\gamma)}) \pi(\mu_{j,b}^{(\delta)}) d\mu_{j,b}^{(\delta)} \\
&= \left(\sigma_\delta^{-2} + \sum_{i \in n_{j,b}^{(\delta)}} \sigma_{\delta,i}^{-2} \right)^{-1/2} (2\pi)^{-|n_{j,b}^{(\delta)}|/2} \sigma_\delta^{-1} \prod_{i \in n_{j,b}^{(\delta)}} \sigma_{\delta,i}^{-1} \\
&\quad \exp\left(-\frac{1}{2} \left[\sum_{i \in n_{j,b}^{(\delta)}} \frac{z_i^{(\delta)2}}{\sigma_{\delta,i}^2} + \frac{\mu_0^{(\delta)2}}{\sigma_\delta^2} \right] \right. \\
&\qquad \left. + \frac{1}{2} \left(\sigma_\delta^{-2} + \sum_{i \in n_{j,b}^{(\delta)}} \sigma_{\delta,i}^{-2} \right)^{-1} \left(\sum_{i \in n_{j,b}^{(\delta)}} \frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2} + \frac{\mu_0^{(\delta)}}{\sigma_\delta^2} \right)^2 \right).
\end{aligned} \tag{4.25}
$$

When we are sampling trees for τ, we let $\delta = \tau$ and $\gamma = h$, $T_{j\notin M_\delta}^{(\delta)}$ denotes the set of trees comprising $\tau(x)$, not including the j^{th} tree, $|n_{j,b}^{(\delta)}|$ is the magnitude of the set $n_{j,b}^{(\tau)}$ and $z_i^{(\delta)} = z_i^{(\tau)}$ and $\sigma_{\delta,i}^2 = \sigma_{\tau,i}^2$ are as defined in equations 4.22.

Similarly, when we are sampling trees for h, we let $\delta = h$ and $\gamma = \tau$, $T_{j\notin M_\delta}^{(\delta)}$ denotes the set of trees comprising $h(x)$, not including the j^{th} tree, $|n_{j,b}^{(\delta)}|$ is the magnitude of the set $n_{j,b}^{(h)}$ and $z_i^{(\delta)} = z_i^{(h)}$ and $\sigma_{\delta,i}^2 = \sigma_{h,i}^2$ are as defined in equations 4.23.

With the marginal likelihood associated with the leaf node parameters, we can propose a growth move as follows.

1. Randomly select a terminal node $b \in \{1, ..., B\}$ with probability $\frac{1}{B}$ where B is the number of terminal nodes in tree j.

2. Introduce a new cut rule $v_b \sim \pi_v(v_b)$ and cut point $c_b \sim \pi_c(c_b)$ where π_v and π_c are discrete uniform on the available variables and cut points.

3. After generating a proposal T', we accept with probability,

$$
\alpha = \min \left[1, \frac{\pi(T'|\sigma^2, Y_i^*) q(T|T')}{\pi(T|\sigma^2, Y_i^*) q(T'|T)} \right] \tag{4.26}
$$

When sampling trees, let L_{left} denote the marginal likelihood associated with the proposed left child of node b in tree j, L_{right} denote the marginal

likelihood associated with the proposed right child of node b in tree j and $L_{combined}$ be the terminal node b from tree j prior to the proposal, then we have,

$$\begin{aligned}\pi(T'|\sigma^2, \mathbf{y}) = kL_{left}(Y_i^*; \sigma^2, T_{j\notin M_\delta}^{(\delta)}, T^{(\gamma)})L_{right}(Y_i^*; \sigma^2, T_{j\notin M_\delta}^{(\delta)}, T^{(\gamma)}) \\ \pi(\text{node } b \text{ in tree } j \text{ is internal}) \\ \pi(\text{proposed left child of node } b \text{ in tree } j \text{ is terminal}) \\ \pi(\text{proposed right child of node } b \text{ in tree } j \text{ is terminal}) \\ \pi_v(v_{j,b} = v_b)\pi_c(c_{j,b} = c_b)\end{aligned}$$

and

$$\pi(T|\sigma^2, \mathbf{y}) = kL_{combined}(Y_i^*; \sigma^2, T_{j\notin M_\delta}^{(\delta)}, T^{(\gamma)})\pi(\text{node } b \text{ in tree } j \text{ is terminal})$$

for $\delta \in \{\tau, h\}$ and $\gamma \in \{\tau, h\}$. Recall that from equation 4.17

$$\begin{aligned}\pi(\text{node is internal}) &= \frac{\alpha}{(1+d)^\beta} \\ \pi(\text{node is terminal}) &= 1 - \frac{\alpha}{(1+d)^\beta}\end{aligned}$$

where d is the dept of the node. When sampling trees for τ, we let $\delta = \tau$ and $\gamma = h$. When sampling trees for h, we let $\delta = h$ and $\gamma = \tau$. Finally, we have k represents terms associated with leaf nodes other than node b and tree j, since these are constant across $\pi(T|\sigma^2, \mathbf{y})$ and $\pi(T'|\sigma^2, \mathbf{y})$.

Therefore, the expression for the ratio of likelihoods is

$$\begin{aligned}
&\frac{L_{left}(Y_i^*;\sigma^2,T_{j\notin M}^{(\delta)},T^{(\gamma)})L_{right}(Y_i^*;\sigma^2,T_{j\notin M}^{(\delta)},T^{(\gamma)})}{L_{combined}(Y_i^*;\sigma^2,T_{j\notin M}^{(\delta)},T^{(\gamma)})} \\
&= \frac{\left(\sigma_\mu^{-2}+\sum_{i\in n_{j,b_{combined}}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{1/2}}{\sigma_\mu\left(\sigma_\mu^{-2}+\sum_{i\in n_{j,b_{left}}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{1/2}\left(\sigma_\mu^{-2}+\sum_{i\in n_{j,b_{right}}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{1/2}} \\
&\quad \exp\left(\frac{1}{2}\left(\sigma_\mu^{-2}+\sum_{i\in n_{j,b_{left}}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-1}\left(\sum_{i\in n_{j,b_{left}}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0}{\sigma_\mu^2}\right)^2\right) \\
&\quad \exp\left(\frac{1}{2}\left(\sigma_\mu^{-2}+\sum_{i\in n_{j,b_{right}}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-1}\left(\sum_{i\in n_{j,b_{right}}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0}{\sigma_\mu^2}\right)^2\right) \\
&\quad \exp\left(-\frac{1}{2}\left(\sigma_\mu^{-2}+\sum_{i\in n_{j,b_{combined}}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-1}\left(\sum_{i\in n_{j,b_{combined}}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0}{\sigma_\mu^2}\right)^2\right)
\end{aligned} \tag{4.27}$$

for $\delta \in \{\tau, h\}$ and $\gamma \in \{\tau, h\}$. When $\delta = \tau$ and $\gamma = h$, we have $\sigma_{\tau,i}$ and $z_i^{(\tau)}$ as defined in Equation 4.22. When $\delta = h$ and $\gamma = \tau$, we have $\sigma_{h,i}$ and $z_i^{(h)}$ as defined in Equation 4.23.

A death proposal is accomplished similarly to a birth proposal since a death move is the direct opposite of the birth move with an acceptance probability of α^{-1}. Detailed derivations of the Metropolis-Hastings step are available in Appendix B.4

4.6.4 Computational Complexity

GPMM vs Causal BART Mixture Model Improve on the computational complexity for the currently only known Causal Mixture Model based on the TRV of O(n3) Zaidi and Mukherjee (2018), since Gaussian Processes are known to be more computationally expensive than BART models. Pratola et al. (2013) showed the com- putational complexity of BART is O(mn + nb), where

m is the number of trees, n is the number of observations, and b is a random variable representing the number of leaf nodes in a tree. In practice due to the regularization on tree depth, Pratola et al. (2013) showed that b is rea- sonably small and that without the regularization, BART's Complexity is bounded by O(mn + n2) ¡ O(n3).

4.7 Experimental Evaluation

In this section we explore our TRV BART-based model on multiple simulated data sets each with varying characteristics. We discuss how the ATE estimation is affected by the unique properties of our model. We also compare against other Bayesian tree based ensemble methods that directly estimate μ_1 and μ_0 or estimate μ_0 and τ. Results are based on 200 independent replications for each data generating process (DGP).

4.7.1 Evaluation Metrics

In this section we will introduce the evaluation metrics we used. We also describe how we compute the relevant metrics for competing methods when there is not a direct counterpart as is the case for the non-Bayesian approaches. We consider performance for both CATE as well as ATE. We will evaluate performance with the following metrics. Root mean square error, which will inform on average how far estimates are from the true CATE or ATE. Bias, which will indicate if on average a particular methods is biased relative to the true CATE or ATE. Coverage which will indicate if a method is accurately capturing uncertainty when estimating the CATE or ATE. Finally, we will also evaluate the mean interval length which is an indication of how well a method is able to capture uncertainty in its CATE or ATE estimates.

CATE metrics: We first specify the method used to estimate the CATE. The model is trained on data $(X_1, ..., X_N)$ and the two estimated function produce estimates

$$\boldsymbol{\tau} = (\tau(X_1), ..., \tau(X_n))$$
$$\mathbf{h} = (h(X_1), ..., h(X_n))$$

For the Bayesian approaches, we have K posterior samples $(\boldsymbol{\tau}^j, \mathbf{h}^j)_{j=1}^K$. These samples are then used to compute our evaluation metrics for the CATE for each X_i for $i \in 1, ..., N$ as

$$\tau_i^{(k)}(X_i) = \tau^{(k)}(X_i)$$
$$\widehat{\tau_i} = \frac{1}{K}\sum_{k=1}^{K} \tau_i^{(k)}(X_i)$$

Given the posterior mean, $\widehat{\tau_i}$ along with the 95% credible intervals, $[\tau_i^{lwr}, \tau_i^{upr}]$, we can compute our CATE metrics as follows.

$$\begin{aligned}
RMSE &= \frac{1}{R}\sum_{r=1}^{R}\left(\frac{1}{N}\sum_{i=1}^{N}(\tau_i - \widehat{\tau_i})^2\right)^{1/2} \\
Bias &= \frac{1}{R}\sum_{r=1}^{R}\frac{1}{N}\sum_{i=1}^{N}(\tau_i - \widehat{\tau_i}) \\
Coverage &= \frac{1}{R}\sum_{r=1}^{R}\frac{1}{N}\sum_{i=1}^{N}\mathbb{1}(\tau_i \in [\tau_i^{lwr}, \tau_i^{upr}]) \\
MIL &= \frac{1}{R}\sum_{r=1}^{R}\frac{1}{N}\sum_{i=1}^{N}\tau_i^{upr} - \tau_i^{lwr}
\end{aligned} \tag{4.28}$$

where MIL is the mean interval length, $R = 200$ is the number of replications and $\mathbb{1}$ is the indicator function.

ATE metrics: Similarly, regarding the ATE, we can define similar metrics.

$$\tau^{ATE(k)} = \frac{1}{N}\sum_{i=1}^{N}\tau_i^{(k)}(X_i)$$
$$\widehat{\tau^{ATE}} = \frac{1}{K}\sum_{k=1}^{K}\tau^{ATE(k)}$$

$$
\begin{aligned}
RMSE_{ATE} &= \frac{1}{R}\sum_{r=1}^{R}\left(\frac{1}{K}\sum_{k=1}^{K}(\tau_r^{ATE} - \tau^{ATE(k)})^2\right)^{1/2} \\
Bias_{ATE} &= \frac{1}{R}\sum_{r=1}^{R}(\tau_r^{ATE} - \widehat{\tau_r^{ATE}}) \\
Coverage_{ATE} &= \frac{1}{R}\sum_{r=1}^{R}\mathbb{1}(\tau_r^{ATE} \in [\tau_r^{ATE,lwr}, \tau_r^{ATE,upr}]) \\
MIL_{ATE} &= \frac{1}{R}\sum_{r=1}^{R}\tau_r^{ATE,upr} - \tau_r^{ATE,lwr}
\end{aligned}
\tag{4.29}
$$

Summary of alternative methods used: We will compare our method to other tree based methods. We selected as comparison methods, non-TRV methods including Bayesian Causal Forests (Hahn et al., 2019), causalBART (Hill, 2011), due to their popularity and consistent good performance. We also compare directly against other TRV based methods in order to assess the impact of the parameterization using the mixture distribution. We selected BART as the "off the shelf" method for comparison. We also compare a second TRV tree based method, the "transformed outcome tree" (Athey and Imbens, 2015a). We were not able to to compare against the method in Zaidi and Mukherjee (2018) where the mixture distribution was first proposed as we were unable to get their computer code. In all our experimental runs, we use the true propensity when needed to assess the quality of the estimation of the CATE and ATE.

4.7.2 Data Sets

To evaluate the performance of our model, we consider 4 simulated data sets. The first two are the simulated data sets found in Zaidi and Mukherjee (2018); however we add gaussian noise to the system and set the variance parameter $\sigma^2 = 25$. One is a high dimensional case with 40 predictors, which we will call Case A; the other is a lower dimensional case with 5 predictors, which we will call Case B. We also consider two of the simulated data sets

from Hahn et al. (2019) each with 5 predictors. The first has a non-linear mean response function and heterogeneous treatment effect, which we will call Case C. Additionally, we consider the non-linear mean response function function and homogeneous treatment which we will refer to as Case D. All data sets have a total of 250 observations.

Case A: We begin with the data set with 40 predictors. The data generating process (DGP) is as follows:

$$
\begin{aligned}
&X_k \sim \text{Normal}(0,1); \quad k = 1, ..., 15, \\
&X_k \sim \text{Uniform}(0,1); \quad k = 16, ..., 30, \\
&X_k \sim \text{Bernoulli}(q_k); \quad q_k = \text{logit}^{-1}(X_{k-30} - X_{k-15}); \quad k = 31, ..., 35, \\
&X_k \sim \text{Poisson}(\lambda_k); \quad \lambda_k = 5 + 0.75 X_{k-35}(X_{k-20} + X_{k-5}); \quad k = 36, ..., 40.
\end{aligned}
$$

Next we simulate the propensity scores and the corresponding treatment assignments.

$$
\begin{aligned}
p_i &= \text{logit}^{-1}(0.3 \sum_{k=1}^{5} X_k - 0.5 \sum_{k=21}^{2} 5X_k - 0.0001 \sum_{k=26}^{3} 5X_k + 0.055 \sum_{k=36}^{4} 0X_k), \\
W &\sim \text{Bernoulli}(p_i).
\end{aligned}
$$

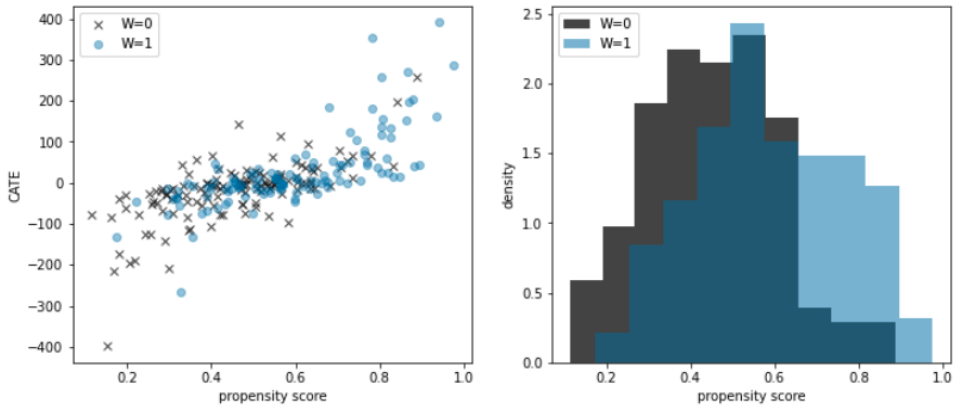
And here we will generate the potential outcomes along with the observed outcomes.

$$
\begin{aligned}
f(\mathbf{X}) &= \frac{\sum_{k=16}^{19} X_k \exp(X_{k+14})}{1 + \sum_{k=16}^{19} X_k \exp(X_{k+14})}, \\
Y(0) &= 0.15 \sum_{k=1}^{5} X_k + 1.5 \exp(1 + 1.5 f(\mathbf{X})) + \epsilon_i, \\
Y(1) &= \sum_{k=1}^{5} (2.15 X_k + 2.75 X_k^2 + 10 X_k^3) + 1.25 \left(0.5 + 1.5 \sum_{k=36}^{4} 0X_k \right)^{1/2} + \epsilon_i, \\
&= WY(1) + (1 - W)Y(0); \quad \epsilon_i \overset{i.i.d.}{\sim} \text{Normal}(0, 25)
\end{aligned}
$$

Figure 4.3 shows some summary distributions along with the relationship of the propensity score and the CATE, including the degree of overlap of the

two treatment conditions. In Case A there is a good amount of overlap of the propensity scores for each level of treatment but we do see in the larger and smaller values of the propensity score, overlap tends to decrease as compared to when the propensity scores are closer to 0.5.

Figure 4.3: Summary plots of Case A. On the left CATE vs propensity score broken down by treatment assignment (W). On the right histogram of the propensity scores broken down by W.



Case B: Next we define the DGP for Case B which is similar to the one proposed in Zaidi and Mukherjee (2018) and which is itself an adaptation of the nonlinear heterogeneous dataset proposed in Hahn et al. (2019) but again rather than have a near deterministic outcome we adapt the DGP to include a non-trivial amount of noise in the variance term, $\sigma^2 = 25$. The DGP is as follows:

$$X_k \sim \text{Normal}(0, 1); \quad k = 1, ..., 3,$$
$$X_4 \sim \text{Bernoulli}(p = 0.25),$$
$$X_5 \sim \text{Binomial}(n = 2, p = 0.5).$$

Next we generate the propensity scores.

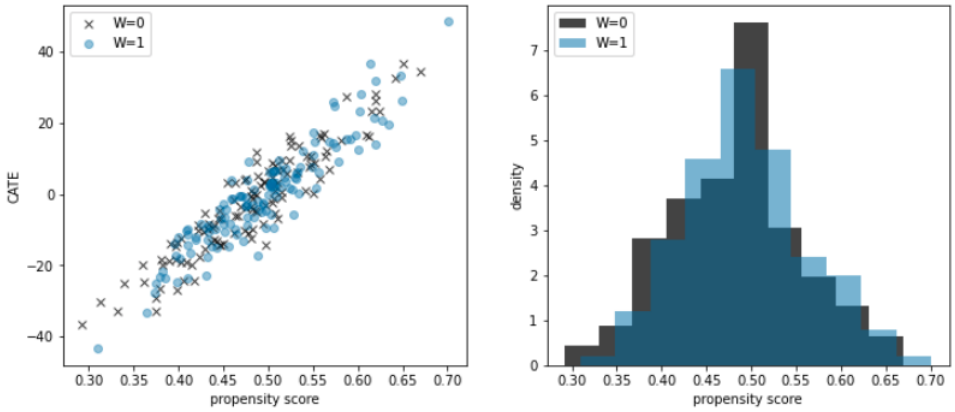
$$p_i = \text{logit}^{-1}(0.1X_1 0.001X_2 + .275X_3 0.03X_4),$$
$$W \sim \text{Bernoulli}(p_i)$$

Finally we generate the potential and observed outcomes.

$$
\begin{aligned}
f(X) &= -1 - 2X_5 + |X_3 - 1|, \\
Y(0) &= f(X) - 15X_3 + \epsilon_i, \\
Y(1) &= f(X) + (1 + 2X_2X_3) + \epsilon_i, \\
Y &= WY(1) + (1W)Y(0); \quad \epsilon_i \overset{i.i.d.}{\sim} Normal(0, 25).
\end{aligned}
$$

Figure 4.4 shows some summary distributions along with the relationship of the propensity score and the CATE. In Case B we appear to have greater overlap in the propensity scores as compared to Case A for the two treatment groups.

Figure 4.4: Summary plots of Case B. On the left CATE vs propensity score broken down by treatment level (W). On the right histogram of the propensity scores broken down by W.



Case C: Next we consider the DGP from Hahn et al. (2019) with a non-linear mean function, $\mu_0(x)$ and heterogeneous treatment effect, $\tau(x)$. The

data is generated as follows

$$
\begin{aligned}
X_1 &\sim \text{Normal}(0, 1), \\
X_2 &\sim \text{Normal}(0, 1), \\
X_3 &\sim \text{Normal}(0, 1), \\
X_4 &\sim \text{Discrete Uniform}(\{1, 2, 3\}) \\
X_5 &\sim \text{Binomial}(n = 1, p = .5).
\end{aligned}
$$

Next, we generate the non linear mean function and CATE as follows

$$
\begin{aligned}
\mu_0(X) &= -1 - 3(X_4) + 6|X_3 - 1|, \\
\tau(x) &= 1 + 2X_2X_5.
\end{aligned}
$$

Finally, we generate the propensity score.

$$
\pi(x_i) = 0.8\Phi\left(\frac{3\mu_0(x_i)}{s - 0.5X_1} + 0.05 + \frac{u_i}{10}\right),
$$

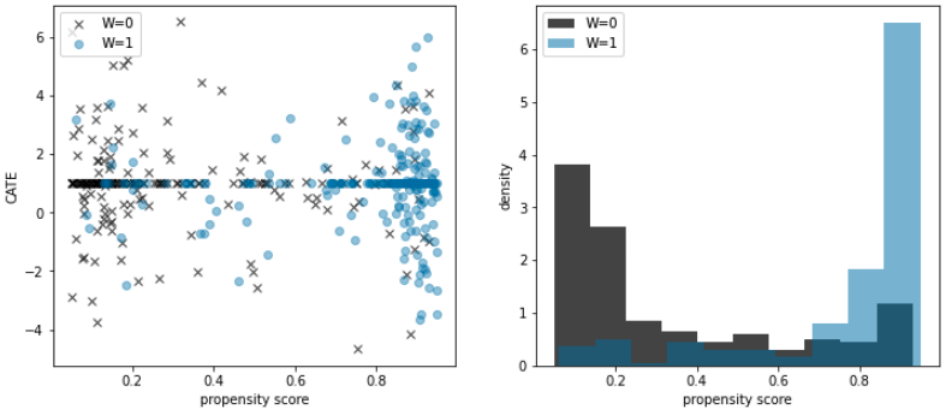
where s is the standard deviations of μ_0 computed from the sample and u_i has a Uniform$(0, 1)$.

Figure 4.5 shows some summary distributions along with the relationship of the propensity score and the CATE. We can see in this example relatively large numbers of the sample have a CATE of 1 as well as large numbers with spread around 1. We can also see that the overlap in the propensity scores is very poor with a large amount of observations from the control condition having propensity scores below 0.2 and similarly, large number of observations from the treatment condition with propensity scores above 0.8.

Case D: Finally, we consider the DGP from Hahn et al. (2019) with a non-linear mean function, $\mu_0(x)$ and homogeneous treatment effect, $\tau(x)$. The data is generated as follows.

$$
\begin{aligned}
X_1 &\sim \text{Normal}(0, 1), \\
X_2 &\sim \text{Normal}(0, 1), \\
X_3 &\sim \text{Normal}(0, 1), \\
X_4 &\sim \text{Discrete Uniform}(\{1, 2, 3\}) \\
X_5 &\sim \text{Binomial}(n = 1, p = .5)
\end{aligned}
$$

Figure 4.5: Summary plots of Case C (non-linear $\mu_0(x)$ and heterogeneous treatment effect). On the left CATE vs propensity score broken down by treatment level (W). On the right histogram of the propensity scores broken down by W.



Next we generate the non linear mean function and treatment effects,

$$\mu_0(X) = -1 - 2X_4 + 6|X_3 - 1|,$$
$$\tau(x) = 3.$$

Finally, we generate the propensity score.

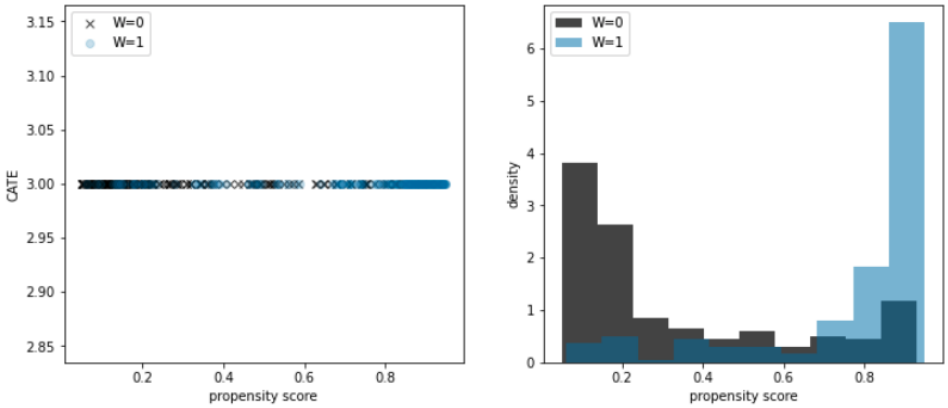
$$\pi(x_i) = 0.8\Phi\left(\frac{3\mu_0(x_i)}{s - 0.5X_1} + 0.05 + \frac{u_i}{10}\right),$$

where $s = std(\mu_0)$ is computed from the sample and $u_i \sim \text{Uniform}(0, 1)$. Figure 4.6 shows some summary distributions along with the relationship of the propensity score and the CATE. Cases C and D have the same mechanism for generating the propensity scores and therefore will have the same distribution.

4.7.3 Comparison Methods

In this section we discuss the results when comparing our method against competing models. For non-Bayesian methods, we relied on bootstrapping to generate uncertainty estimates. The Causal BART Mixture Model was

Figure 4.6: Summary plots of Case D (non-linear $\mu_0(x)$ and homogeneous treatment effect). On the left homogeneous effect vs propensity score broken down by treatment level (W). On the right histogram of the propensity scores broken down by W.



run with 2000 burn-in and 2000 sample iterations. Each method was run with 200 replications. Results of the analysis are summarized with a collection of histograms laid out in a tabular format allowing a qualitative evaluation of the summary distributional characteristics as well as the quantitative mean. We also report the means of our evaluation metrics in tables in Appendix B.5.

We use the following labels for the methods.

- **CBARTMM** - The Causal BART Mixture Model with 25 trees in τ and h, tree depth priors for τ and h with $\alpha = 0.95$ and $k = 2.0$, standard regularization priors for $\mu^*_{i,j}$ defined in equations 4.16, hyper parameters for σ had $a = 3.0$ and $q = 0.99$ which where were discussed in section 4.5.2.1. We explored tree count values between 2 and 250; Higher values increased computational cost with no significant increase in performance. We ran CBARTMM with with 2000 burn-in and 2000 sample iterations
- **BCF** - The Bayesian Causal Forest introduced in Hahn et al. (2019) and available as an R package (Hahn et al., 2017). We used the default

settings and the same covariates for the conditional mean function under the control condition, $\mu(x)$, and treatment effect function, $\tau(x)$. The propensity score is also used as a predictor in this example. This method uses two BART models, one for $\mu_0(x)$ and one for $\tau(x)$. For a full discussion of BCF functionality, see Hahn et al. (2017).

- **BART** - The method introduced in Hill (2011) which uses two BART models, one for the conditional mean under the control condition, μ_0 and one for conditional mean under the treatment condition, μ_1. This method does not use the propensity score. We used the default settings.
- **TOT** - The Adaptive Transformed Outcome Tree introduced in Athey and Imbens (2015a) and is available in the R package Athey et al. (2016). We use the default settings. This method is a single decision tree and makes implicit use of the propensity score through the TRV, and does not include the propensity score as a predictor.
- **BART (TRV)** - regular BART with the TRV used as the response. This is the particular comparison that allows us to gauge the "off-the-shelf" performance of BART modeling $\tau(x)$ with the TRV. In particular, this method models $\tau(x_i)$ as $E[Y_i^*|X = x_i]$ and assumes a constant variance.

4.7.4 CATE Results

Results for the CATE are given in Figures 4.7, 4.8, 4.9, 4.10. In each case, we see improved RMSE over the other TRV-based methods. This suggests that modeling the heterogeneity in the TRV allows us to better estimate the CATE. However, our coverage tends to be lower than the TRV methods. We hypothesize that this is because the interval length of the other TRV methods is too high to accommodate the heterogeneity.

Compared with non-TRV, BART-based methods, the story is less clear. Our method tends to perform comparably (in terms of RMSE) with the basic jhbart method, but slightly worse than the hbcf method. As we discuss in 4.7.6 we hypothesize that our method will tend to perform worse with data where the observations have propensity scores close to 0 or 1 causing some of our CATE estimates to be relatively large, and raising the overall RMSE.

Figure 4.7: Case A CATE

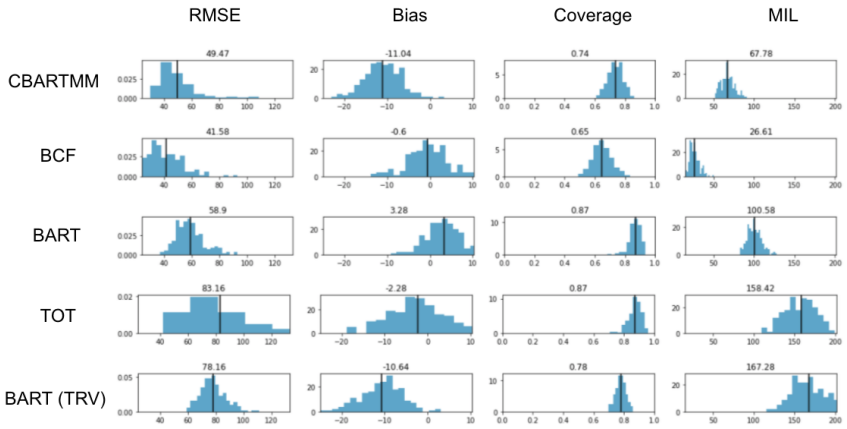


Figure 4.8: Case B CATE

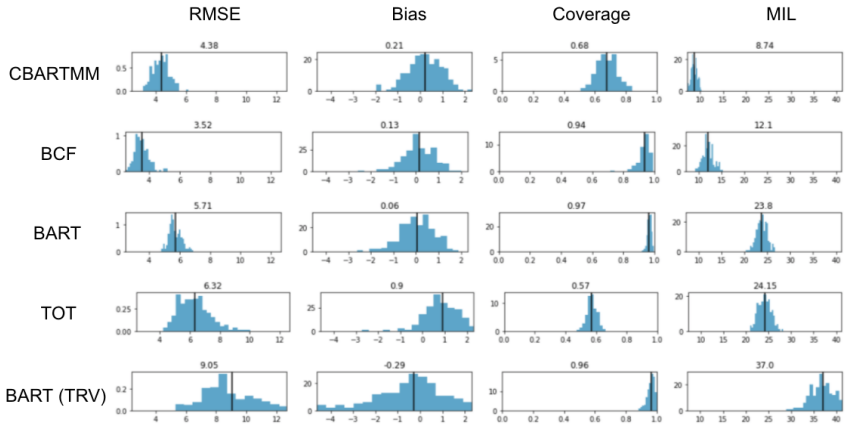


Figure 4.9: Case C CATE

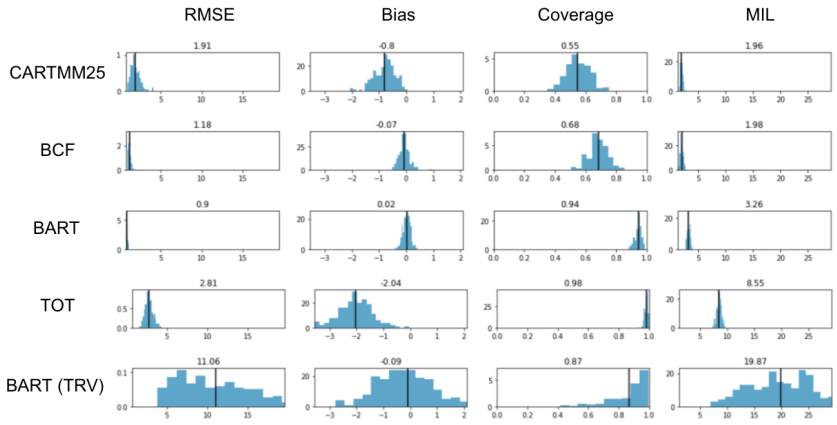
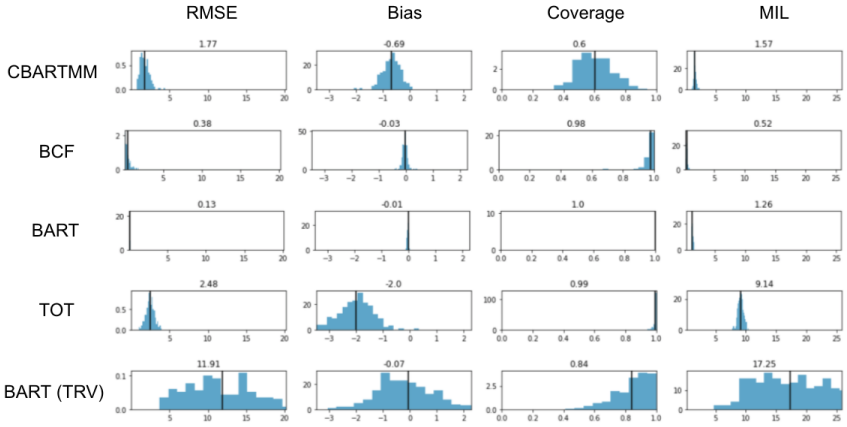


Figure 4.10: Case D CATE



4.7.5 ATE Results

Next we consider the results for the ATE in Figures 4.11, 4.12, 4.13, 4.14. Our method has the poor coverage. We will elaborate further on reasons why in section 4.7.6. However, although we have poor coverage of the ATE, our RMSE is still consistently below the RMSE of the other models using the TRV. The non-TRV methods however are the clear winners when it comes to

estimating the ATE.

Figure 4.11: Case A ATE

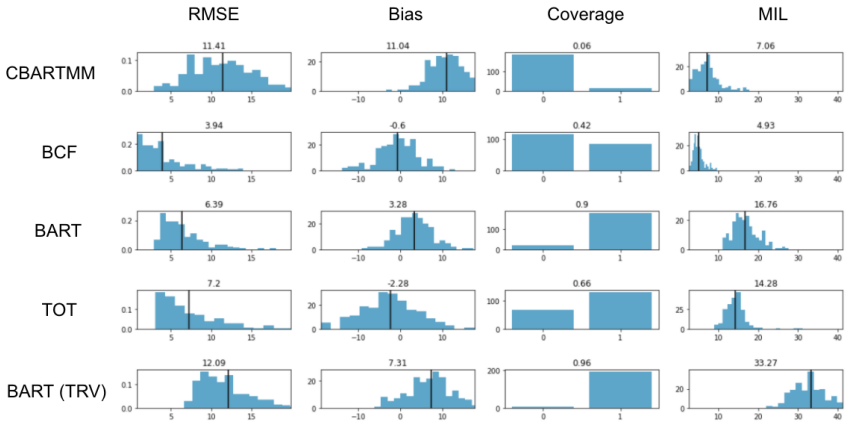


Figure 4.12: Case B ATE

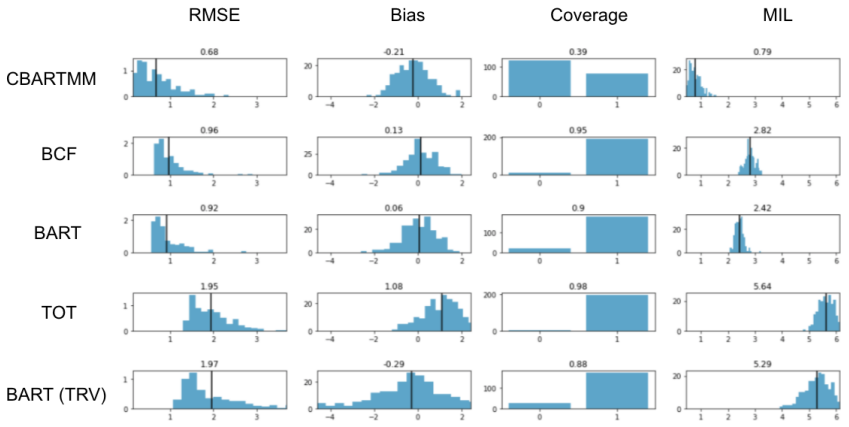


Figure 4.13: Case C ATE

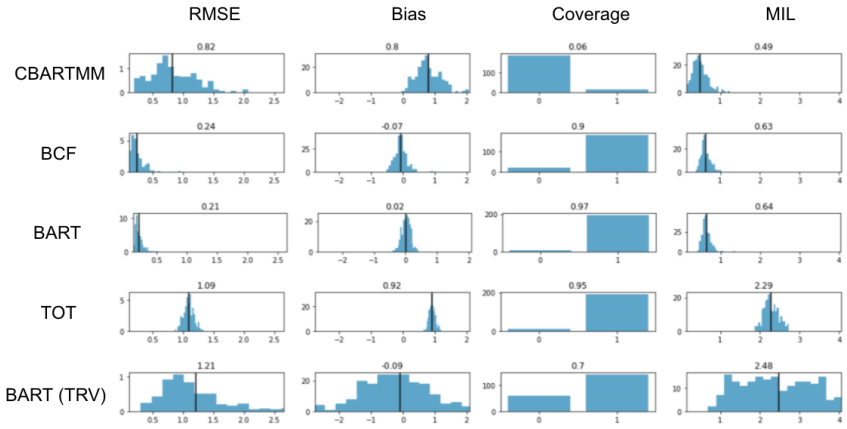
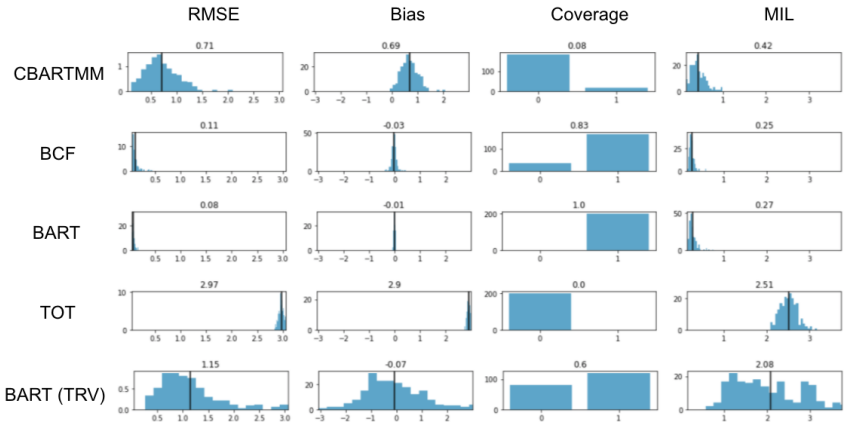


Figure 4.14: Case D ATE



4.7.6 Problems with estimating $h(x)$

In all the above experiments, while CBARTMM is competitive in terms of CATE, it tends to have higher RMSE and lower coverage when estimating the ATE. In this section, we provide some intuition for this.

We suspect the primary driver in the poor ATE estimation is related

to values of the propensity score p_i. Recall,

$$h(x) = \frac{\mu_1(x)}{p_i} + \frac{\mu_0(x)}{1 - p_i}. \tag{4.30}$$

It is easy to see that large and small values of pi can lead to exploding values of h and hence directly impact our ability to estimate $h(x)$. These poor estimates of h consequntly lead to poor esimates of the CATE producing an outlier effect when marginalizing over X. We believe this is due to two factors. First, large values of h tend to be associated with extreme propensity scores. Estimating causal effects in such scenarios is challenging due to data sparsity in regions where there is not good overlap between units of both treatment conditions. Additionally, BART or more generally, tree based models are not good at estimating steep functions which will be an issue when h is rapidly increasing.

To illustrate the issue with h and coverage of the CATE estimate, we modeled the coverage probability for the CATE via a Gaussian process classifier using the true values of $h(x)$ as the only predictor. The results are in Figures 4.15, 4.16, 4.17, 4.18 illustrate the issues with coverage and values of $h(x)$. It is easy to see in the tails of the distribution of h, we have a drop off in coverage of the CATE, providing evidence $h(x)$ is interfering with out ability to properly estimate $\tau(x)$. We also modeled the coverage probability for the CATE via a Gaussian process classifier using the true values of p_i as the only predictor. When comparing p_i to the coverage of CATE, we can see a clear drop off in coverage for values of the propensity score away from 0.5. The effect if most pronounced for values of the propensity score as they approach 0 or 1. We also included in the plots the residuals of τ vs p_i and h vs p_i. Again, we see a similar issue whith exploding values of the error in estimating τ or h for values of p_i that near 0 or 1.

Figure 4.15: Plots of Case A. On the top left is CATE coverage probability vs $h(x)$. On the bottom left is distribution of $h(x)$. Vertical marks '|' indicate which observation with specific value of h are covered (indicated by value of 1) or not covered (indicated with a value of 0). Coverage is calculated for the 95% credible interval. On the top center CATE coverage probability vs propensity score. On the bottom center is the distribution of the propensity score. Vertical marks '|' again indicate coverage. Top right plot shows the error $(\widehat{\tau}(x_i) - \tau(x_i))$ vs the propensity score. Bottom right plot shows the error $(\widehat{h}(x_i) - h(x_i))$ vs the propensity score.

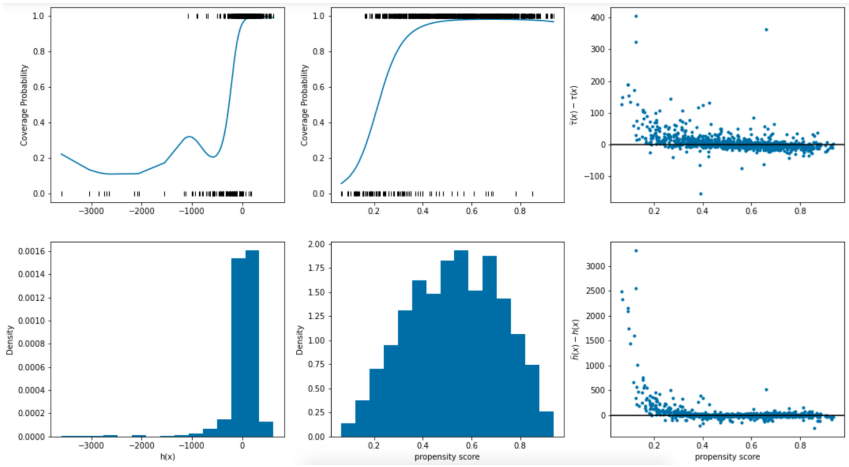


Figure 4.16: Plots of Case B. On the top left is CATE coverage probability vs $h(x)$. On the bottom left is distribution of $h(x)$. Vertical marks '|' indicate which observation with specific value of h are covered (indicated by value of 1) or not covered (indicated with a value of 0). Coverage is calculated for the 95% credible interval. On the top center CATE coverage probability vs propensity score. On the bottom center is the distribution of the propensity score. Vertical marks '|' again indicate coverage. Top right plot shows the error $(\widehat{\tau}(x_i) - \tau(x_i))$ vs the propensity score. Bottom right plot shows the error $(\widehat{h}(x_i) - h(x_i))$ vs the propensity score.

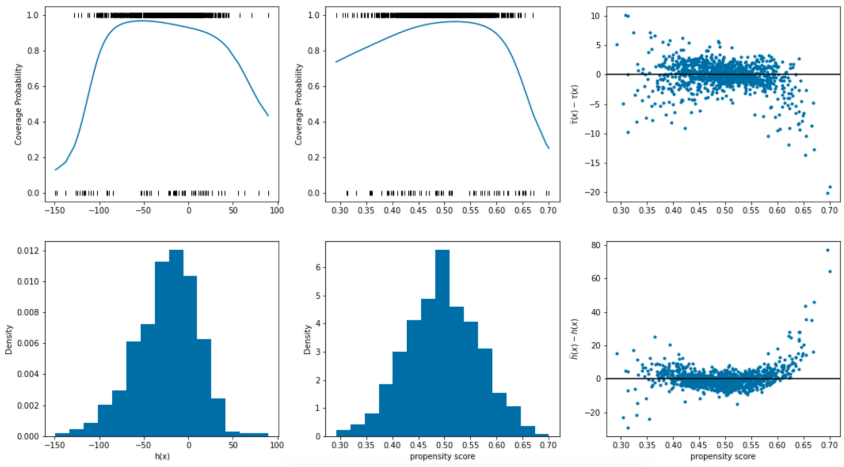


Figure 4.17: Plots of Case C. On the top left is CATE coverage probability vs $h(x)$. On the bottom left is distribution of $h(x)$. Vertical marks '|' indicate which observation with specific value of h are covered (indicated by value of 1) or not covered (indicated with a value of 0). Coverage is calculated for the 95% credible interval. On the top center CATE coverage probability vs propensity score. On the bottom center is the distribution of the propensity score. Vertical marks '|' again indicate coverage. Top right plot shows the error $(\widehat{\tau}(x_i) - \tau(x_i))$ vs the propensity score. Bottom right plot shows the error $(\widehat{h}(x_i) - h(x_i))$ vs the propensity score.

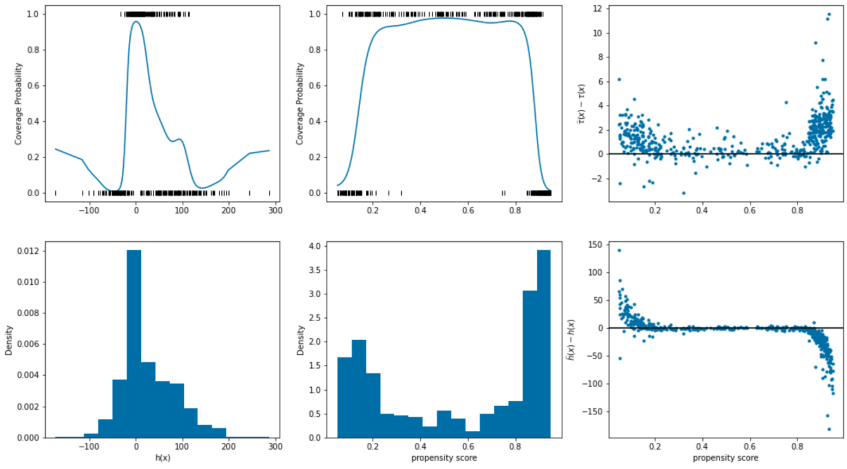
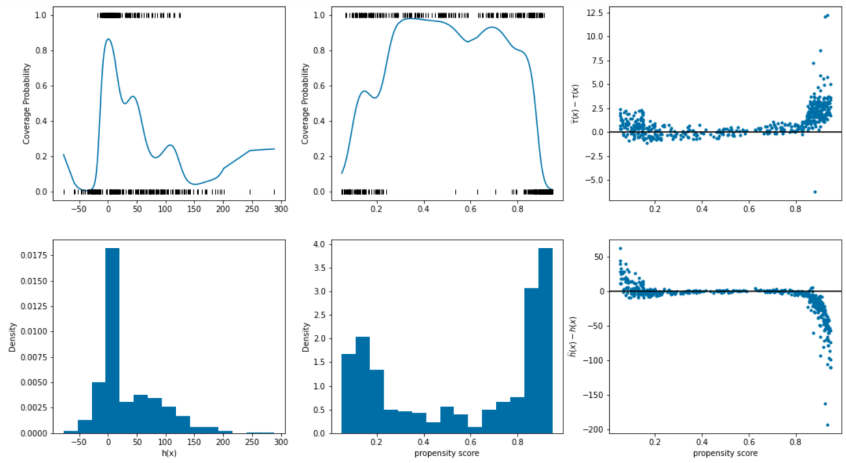


Figure 4.18: Plots of Case D. On the top left is CATE coverage probability vs $h(x)$. On the bottom left is distribution of $h(x)$. Vertical marks '|' indicate which observation with specific value of h are covered (indicated by value of 1) or not covered (indicated with a value of 0). Coverage is calculated for the 95% credible interval. On the top center CATE coverage probability vs propensity score. On the bottom center is the distribution of the propensity score. Vertical marks '|' again indicate coverage. Top right plot shows the error $(\widehat{\tau}(x_i) - \tau(x_i))$ vs the propensity score. Bottom right plot shows the error $(\widehat{h}(x_i) - h(x_i))$ vs the propensity score.



4.8 Discussion and Future Work

In this chapter we introduced the Causal BART Mixture Model and showed how accounting for the mixture implied by the TRV improves CATE estimates significantly over existing methods using the TRV. Additionally, we also showed that our method is often competitive with other state of the art methods in estimating CATEs. We also noted that our proposed model can struggle when estimating the ATE. We hypothesize that this is due to

challenges in estimating h which seems to be at the core of the poor estimation related to the ATE from the effect of marginalizing over X. To that end, moving forward, if TRV causal models are to be competitive against other state of the art methods, there is a need to dampen the effects of large values of $h(x)$. Another avenue in future work worth considering would be to incorporate variable selection sparsity inducing priors on the predictor variables (Linero, 2018).

Chapter 5

Conclusion

In this work, we have extended existing approaches to big data as well as causal inference.

In Chapter 3, we explored communication strategies in distributed computing environments for models that perform on-line learning via particle filtering genetic algorithms. We showed that allowing communication between the compute processes has the capacity to significantly improve model fit. We also showed from our experiments the potential downside is limited to increased computing time but does not adversely affect the model fit quality for some models, implying a possible avenue for improving model performance where on-line parameter learning is performed in a distributed computing environment.

In Chapter 4 we introduced a reparameterization for a causal inference model using the TRV. We applied a modification to BART which allowed us to expand the set of methods in estimating CATE's as well as outline what we believe are some inherit challenges in using the TRV for modeling CATE and ATE. Regardless, we were able to explicitly model heterogeneous treatment effects with a distribution implied by the TRV, thereby reducing the variance of the estimators. Finally, we also introduced a method that models both the treatment and control conditions in an observational setting jointly, thereby improving inference over some models that do not.

Appendices

Appendix A

Appendix: Sequential Monte Carlo Particle Filtering

A.1 Consensus Monte Carlo Stochastic Approximation Covariance Adjustment

The Consensus Monte Carlo method, originally proposed by (Scott et al., 2016) posits an approximation to a target distribution can be estimated with the simplifying assumption our posterior distribution is drawn from a MVN distribution when our sampels or particles come from a collection of sub-posteriors. Let $\tilde{P}(\theta)_s \propto P(\theta)P(y_s|\theta)^s$, and assume that this is well approximated by a Gaussian with mean $\widetilde{\mu}_s$, covariance $\widetilde{\Sigma}_s$.

$$\begin{aligned} P(\theta|y)^s &\propto \prod_i \tilde{P}(\theta)_i \\ &\propto \prod_{i=1}^{S} \exp\left(\frac{-1}{2}(\theta-\widetilde{\mu}_i)^T\widetilde{\Sigma}_i^{-1}(\theta-\widetilde{\mu}_i)\right) \end{aligned} \tag{A.1}$$

so,

$$P(\theta|y) \propto \left(\prod_{i=1}^{S} \exp\left(\frac{-1}{2}(\theta-\widetilde{\mu}_i)^T\widetilde{\Sigma}_i^{-1}(\theta-\widetilde{\mu}_i)\right)\right)^{1/s} \tag{A.2}$$

If

$$P(X) \propto \exp\left(-\frac{1}{2}(X-\mu)^T\Sigma^{-1}(X-\mu)\right)$$

then if $P'(X) \propto P(X)^{1/s}$, we have

$$
\begin{aligned}
p'(X) &\propto \left(\exp\left\{-\frac{1}{2}(X-\mu)^T\Sigma^{-1}(X-\mu)\right\}\right)^{1/s} \\
&= \left(\exp\left\{-\frac{1}{2s}(X-\mu)^T\Sigma^{-1}(X-\mu)\right\}\right) \\
&\propto N(X;\mu,s\Sigma)
\end{aligned}
$$

and

$$
P(\theta|y) \propto \prod_{i=1}^{S} \exp\left(-\frac{1}{2}(\theta-\widetilde{\mu}_i)^T(s\widetilde{\Sigma}_i)^{-1}(\theta-\widetilde{\mu}_i)\right)
$$

Which implies we have mean and covariance with

$$
\begin{aligned}
\widetilde{V}^{-1} &= \sum_{i=1}^{S} \widetilde{\Sigma}_i^{-1} S^{-1} \\
&= S^{-1} \sum_{i=1}^{S} \widetilde{\Sigma}_i^{-1}
\end{aligned}
\tag{A.3}
$$

$$
\begin{aligned}
\widetilde{\mu} &= \widetilde{V} \sum_{i=1}^{S} S^{-1}\widetilde{\Sigma}_i^{-1}\widetilde{\mu}_i \\
&= S\left(\sum_{i=1}^{S} \widetilde{\Sigma}_i^{-1}\right)^{-1} S^{-1} \sum_{i=1}^{S} \widetilde{\Sigma}_i^{-1}\widetilde{\mu}_i \\
&= \left(\sum_{i=1}^{S} \widetilde{\Sigma}_i^{-1}\right)^{-1} \sum_{i=1}^{S} \widetilde{\Sigma}_i^{-1}\widetilde{\mu}_i
\end{aligned}
\tag{A.4}
$$

From this we see that, $\mu = \widetilde{\mu}$. Or in other words, the mean of our stochastic approximated sub-posteriors are the same as the Consensus Monte Carlo Mean. Additionally, we can see $S\widetilde{V} = V$. Therefore,

$$
P(\theta|\mathbf{y}) \propto N(\widetilde{\mu}, S\widetilde{V}) \tag{A.5}
$$

Appendix B

Appendix: Causal BART Mixture Model

In this supplement, we provide proofs and derivations for the Causal BART Mixture Model.

B.1 Theorems

In this section we include proofs and results used in Chapter 4.

Multi-Variance Theorem: Let $y_1, .., y_n$ be a sequence of size n of independent random variables with $y_i \sim N(\mu, \sigma_i^2)$ and $\mu \sim N(\mu_0, \sigma_0^2)$ where $\mu_0, \sigma_0^2, ..., \sigma_n^2$ are known, then

$$\mu | y_1, .., y_n, \mu_0, \sigma_0^2, .., \sigma_n^2 \sim N\left(\frac{\frac{\mu_0}{\sigma_0^2} + \sum_{i=1}^n \frac{y_i}{\sigma_i^2}}{\frac{1}{\sigma_0^2} + \sum_{i=1}^n \frac{1}{\sigma_i^2}}, \left[\frac{1}{\sigma_0^2} + \sum_{i=1}^n \frac{1}{\sigma_i^2} \right]^{-1} \right) \tag{B.1}$$

Proof. The posterior for $\mu||y_1, .., y_n, \mu_0, \sigma_0^2, .., \sigma_n^2$ is

$$
\begin{aligned}
&L(\mu|y_1, .., y_n, \mu_0, \sigma_0^2, .., \sigma_n^2) \propto \\
&\quad \exp\left(-\frac{1}{2}\frac{(\mu-\mu_0)^2}{\sigma_0^2}\right)\prod_{i=1}^{n}\exp\left(-\frac{1}{2}\frac{(y_i-\mu)^2}{\sigma_i^2}\right) \\
&\quad = \exp\left(-\frac{1}{2}\frac{(\mu-\mu_0)^2}{\sigma_0^2} - \frac{1}{2}\sum_{i=1}^{n}\frac{(\mu-y_i)^2}{\sigma_i^2}\right) \\
&\quad = \exp\left(-\frac{1}{2}\frac{\mu^2-2\mu_0\mu+\mu_0^2}{\sigma_0^2} - \frac{1}{2}\sum_{i=1}^{n}\frac{\mu^2-2y_i\mu+y_i^2}{\sigma_i^2}\right) \\
&\quad = \exp\left(-\frac{1}{2}\left(\mu^2\left(\frac{1}{\sigma_0^2}+\sum_{i=1}^{n}\frac{1}{\sigma_i^2}\right) - 2\left(\frac{\mu_0}{\sigma_0^2}+\sum_{i=1}^{n}\frac{y_i}{\sigma_i^2}\right)\mu + ...\right)\right) \\
&\quad = \exp\left(-\frac{1}{2}\left(\frac{1}{\sigma_0^2}+\sum_{i=1}^{n}\frac{1}{\sigma_i^2}\right)\right. \\
&\qquad\qquad \left.\left(\mu^2 - 2\left(\frac{1}{\sigma_0^2}+\sum_{i=1}^{n}\frac{1}{\sigma_i^2}\right)^{-1}\left(\frac{\mu_0}{\sigma_0^2}+\sum_{i=1}^{n}\frac{y_i}{\sigma_i^2}\right)\mu + ...\right)\right)
\end{aligned}
$$

Completing the square yields

$$
L(\mu|y, \sigma_1^2, ..., \sigma_n^2) \propto \exp\left(-\frac{1}{2}\frac{\left(\mu - \left[\frac{1}{\sigma_0^2}+\sum_{i=1}^{n}\frac{1}{\sigma_i^2}\right]^{-1}\left[\frac{\mu_0}{\sigma_0^2}+\sum_{i=1}^{n}\frac{y_i}{\sigma_i^2}\right]\right)^2}{\left[\frac{1}{\sigma_0^2}+\sum_{i=1}^{n}\frac{1}{\sigma_i^2}\right]^{-1}}\right)
$$

Therefore,

$$
\begin{aligned}
&\mu|y_1, ..., y_n, \mu_0, \sigma_0^2, .., \sigma_n^2 \\
&\quad \sim N\left(\left[\frac{1}{\sigma_0^2}+\sum_{i=1}^{n}\frac{1}{\sigma_i^2}\right]^{-1}\left[\frac{\mu_0}{\sigma_0^2}+\sum_{i=1}^{n}\frac{y_i}{\sigma_i^2}\right], \left[\frac{1}{\sigma_0^2}+\sum_{i=1}^{n}\frac{1}{\sigma_i^2}\right]^{-1}\right).
\end{aligned}
$$

□

The conditional expectation of Y_i^*: $E[Y_i^*|X_i = x] = \tau(x)$

Proof.

$$\begin{aligned} E[Y_i^*|X_i = x] &= E[Y_i \frac{W_i - p_i}{p(1-p_i)}|X_i = x] \\ &= E[W_i Y_i \frac{W_i - p_i}{p_i(1-p_i)} + (1-W_i)Y_i \frac{W_i - p_i}{p_i(1-p_i)}|X_i = x] \end{aligned}$$

Because $W_i Y_i = W_i(1)$ and $(1-W_i)Y_i = (1-W_i)Y_i(0)$, we can re-write the expression as

$$\begin{aligned} E[Y_i^*|X_i = x] &= E[W_i Y_i(1) \frac{W_i - p_i}{p_i(1-p_i)} + (1-W_i)Y_i(0) \frac{W_i - p_i}{p_i(1-p_i)}|X_i = x] \\ &= E[Y_i(1) \frac{W_i(1-p_i)}{p_i(1-p_i)}] - E[Y_i(0) \frac{(1-W_i)p_i}{p_i(1-p_i)}|X_i = x] \end{aligned}$$

Under no unmeasured confounding, we have

$$\begin{aligned} E[Y_i^*|X_i = x] &= E[Y_i(1)|X_i = x]E[W_i|X_i = x]\frac{1}{p_i} \\ &\quad - E[Y_i(0)|X_i = x]E[1 - W_i|X_i = x]\frac{1}{1-p_i} \\ &= \mu_1(x) - \mu_0(x) = \tau(x) \end{aligned}$$

□

The conditional variance of Y_i^*:

Let

$$\begin{aligned} E[Y_i(1)|X_i = x] &= \mu_1(x) = \mu_1 \\ E[Y_i(0)|X_i = x] &= \mu_0(x) = \mu_0 \\ Var[Y_i(1)|X_i = x] &= \sigma_1^2 \\ Var[Y_i(0)|X_i = x] &= \sigma_0^2 \\ W_i|X_i = x &\sim Bernoulli(p_i) \\ Y_i^* &= \frac{Y_i(w_i - p_i)}{p_i(1-p_i)} \\ E[Y_i^*|X_i = x] &= \tau(x) \end{aligned}$$

then,

$$\begin{aligned}\mathrm{Var}[\mathrm{Y}_i^*|X_i = x] &= \frac{\sigma_1^2 + \mu_1^2}{p_i} + \frac{\sigma_0^2 + \mu_0^2}{1 - p_i} - \tau(x)^2 \\ &= \frac{\sigma_1^2}{p_i} + \frac{\sigma_0^2}{1 - p_i} + \frac{(1 - p_i)\mu_1^2}{p_i} + \frac{p_i\mu_0^2}{1 - p_i} - 2\mu_1\mu_0\end{aligned}$$

Proof. We will need the following expectation results to derive

$$Var[Y_i^*|X_i = x].$$

Conditioning on $X_i = x$,

$$\begin{aligned}E[W_i(W_i - p_i)^2|X_i = x] &= \sum_{w_i=0}^{1} w_i(w_i - p_i)^2 p_i^{w_i}(1 - p_i)^{1-w_i} \\ &= p_i(1 - p_i)^2\end{aligned}$$

$$\begin{aligned}E[(1 - W_i)(W_i - p_i)^2|X_i = x] &= \sum_{w_i=0}^{1} (1 - w_i)(w_i - p_i)^2 p_i^{w_i}(1 - p_i)^{1-w_i} \\ &= p_i^2(1 - p_i)\end{aligned}$$

$$\begin{aligned}E[W_i(W_i - p_i)|X_i = x] &= \sum_{w_i=0}^{1} w_i(w_i - p_i) p_i^{w_i}(1 - p_i)^{1-w_i} \\ &= p_i(1 - p_i)\end{aligned}$$

$$\begin{aligned}E[(1 - W_i)(W_i - p_i)|X_i = x] &= \sum_{w_i=0}^{1} (1 - w_i)(w_i - p_i) p_i^{w_i}(1 - p_i)^{1-w_i} \\ &= -p_i(1 - p_i)\end{aligned}$$

$$
\begin{aligned}
Var[Y_i^*|X_i = x] &= E\big[(Y_i^* - \tau(x))^2|X_i = x\big] \\
&= E\left[(\frac{Y_i(W_i - p_i)}{p_i(1-p_i)} - \tau(x))^2|X_i = x\right] \\
&= E\left[\frac{Y_i^2(W_i - p_i)^2}{p_i^2(1-p_i)^2} - 2\frac{Y_i(W_i - p_i)}{p_i(1-p_i)}\tau(x) + \tau(x)^2|X_i = x\right] \\
&= E\left[W_i\frac{Y_i(1)^2(W_i - p_i)^2}{p_i^2(1-p_i)^2} + (1-W_i)\frac{Y_i(0)^2(W_i - p_i)^2}{p_i^2(1-p_i)^2}\right. \\
&\qquad - 2W_i\frac{Y_i(1)(W_i - p_i)}{p_i(1-p_i)}\tau(x) - \\
&\qquad\qquad \left. 2(1-W_i)\frac{Y_i(0)(W_i - p_i)}{p_i(1-p_i)}\tau(x) + \tau(x)^2|X_i = x\right]
\end{aligned}
$$

Under no unmeasured confounding, we have

$$
\begin{aligned}
Var(Y_i^*) = &\frac{E[Y_i(1)^2|X_i = x]E[W_i(W_i - p_i)^2|X_i = x]}{p_i^2(1-p_i)^2} \\
&+ \frac{E[Y_i(0)^2|X_i = x]E[(1-W_i)(W_i - p_i)^2|X_i = x]}{p_i^2(1-p_i)^2} \\
&- 2\frac{E[Y_i(1)|X_i = x]E[W_i(W_i - p_i)|X_i = x]}{p_i(1-p_i)}\tau(x) \\
&- 2\frac{E[Y_i(0)|X_i = x]E[(1-W_i)(W_i - p_i)|X_i = x]}{p_i(1-p_i)}\tau(x) \\
&+ \tau(x)^2
\end{aligned}
$$

Using the expectations listed above and the fact that

$\tau(x) = E[Y_i(1)|X_i = x] - E[Y_i(0)|X_i = x]$, we have

$$\begin{aligned}
Var(Y_i^*) &= \frac{E\left[Y_i(1)^2|X_i = x\right] p_i(1-p_i)^2}{p_i^2(1-p_i)^2} \\
&\quad + \frac{E\left[Y_i(0)^2|X_i = x\right] p_i^2(1-p_i)}{p_i^2(1-p_i)^2} \\
&\quad - 2\frac{E[Y_i(1)|X_i = x]p_i(1-p_i)}{p_i(1-p_i)}\tau(x) \\
&\quad - 2\frac{E[Y_i(0)|X_i = x]p_i(1-p_i)}{p_i(1-p_i)}\tau(x) + \tau(x)^2 \\
&= \frac{E[Y_i(1)^2|X_i = x]}{p_i} + \frac{E[Y_i(0)^2|X_i = x]}{1-p_i} \\
&\quad - 2\tau(x)(E[Y_i(1)|X_i = x] - E[Y_i(0)|X_i = x]) + \tau(x)^2 \\
&= \frac{E[Y_i(1)^2|X_i = x]}{p_i} + \frac{E[Y_i(0)^2|X_i = x]}{1-p_i} - 2\tau(x)^2 + \tau(x)^2 \\
&= \frac{E[Y_i(1)^2|X_i = x]}{p_i} + \frac{E[Y_i(0)^2|X_i = x]}{1-p_i} - \tau(x)^2
\end{aligned}$$

Next, we use the fact that

$$\begin{aligned}
E[Y_i(0)^2|X_i = x] &= \sigma_0^2 + \mu_0^2 \\
E[Y_i(1)^2|X_i = x] &= \sigma_1^2 + \mu_1^2
\end{aligned}$$

so we have

$$Var(Y_i^*) = \frac{\sigma_1^2 + \mu_1^2}{p_i} + \frac{\sigma_0^2 + \mu_0^2}{1-p_i} - \tau(x)^2$$

At this stage using the fact that $\tau(x) = \mu_1 - \mu_0$, we have

$$\begin{aligned}
Var(Y_i^*) &= \frac{\sigma_1^2 + \mu_1^2}{p_i} + \frac{\sigma_0^2 + \mu_0^2}{1-p_i} - (\mu_1 - \mu_0)^2) \\
&= \frac{\sigma_1^2 + \mu_1^2}{p_i} + \frac{\sigma_0^2 + \mu_0^2}{1-p_i} + \mu_1^2 + \mu_0^2 - 2\mu_1\mu_0 \\
&= \frac{\sigma_1^2}{p_i} + \frac{\sigma_0^2}{1-p_i} + \frac{\mu_1^2}{p_i} - \frac{p_i\mu_1^2}{p_i} + \frac{\mu_0^2}{1-p_i} - \frac{(1-p_i)\mu_0^2}{1-p_i} - 2\mu_1\mu_0
\end{aligned}$$

So, finally we have

$$Var(Y_i^*) = \frac{\sigma_1^2}{p_i} + \frac{\sigma_0^2}{1-p_i} + \frac{(1-p_i)\mu_1^2}{p_i} + \frac{p_i\mu_0^2}{1-p_i} - 2\mu_1\mu_0.$$

□

B.2 Derivation of Mixture Model for Y_i^*

Since $Y_i^* = Y_i \frac{W_i - p_i}{p_i(1-p_i)}$, we can write the expression as a mixture over W_i with

$$\begin{aligned} E[Y_i^*] &= Y_i \frac{1-p_i}{p_i(1-p_i)} P(W_i = 1|X_i = x) + Y_i \frac{0-p_i}{p_i(1-p_i)} P(W = 0|X_i = x) \\ &= Y_i(1) \frac{1-p_i}{p_i(1-p_i)} p_i - Y_i(0) \frac{p_i}{p_i(1-p_i)} (1-p_i) \\ &= Y_i(1) - Y_i(0) \end{aligned}$$

We therefore define

$$\begin{aligned} Y_i(1) &= \mu_1(x_i) + \epsilon_i(1), \text{ where } \epsilon_i(1) \sim (0, \sigma^2) \\ Y_i(0) &= \mu_0(x_i) + \epsilon_i(0), \text{ where } \epsilon_i(0) \sim (0, \sigma^2) \\ h(x_i) &= \frac{\mu_1(x_i)}{p_i} + \frac{\mu_0(x_i)}{1-p_i} \\ \tau(x_i) &= \mu_1(x_i) - \mu_0(x_i) \end{aligned}$$

and assume equal variance under $W = 0$ or 1, i.e., $\sigma_0^2 = \sigma_1^2$.

Substituting $Y_i(1) = \mu_1(x_i) + \epsilon_i(1)$ and $Y_i(0) = \mu_0(x_i) + \epsilon_i(0)$ under the treatment and control cases in the definition of the Transformed Response yields the following mixture model,

$$\begin{aligned} &\mathrm{Y}_i^* = \tau(x_i) + \epsilon_i^*, \\ &\epsilon_i^* \sim p_i N((1-p_i)h(x_i), \frac{1}{p_i^2}\sigma^2) + (1-p_i) N(-p_i h(x_i), \frac{1}{(1-p_i)^2}\sigma^2) \end{aligned} \tag{B.2}$$

Proof. First consider, Y_i^* when $W_i = 1$:

$$\begin{aligned}
Y_i^* &= Y_i(1)\frac{1-p_i}{p_i(1-p_i)} \\
&= (\mu_1(x_i) + \epsilon_i(1))\frac{1}{p_i} \\
&= \frac{\mu_1(x_i)}{p_i} + \frac{\epsilon_i(1)}{p_i} + \mu_1(x_i) - \mu_1(x_i) + \mu_0(x_i) - \mu_0(x_i) \\
&= \mu_1(x_i) - \mu_0(x_i) + \frac{\mu_1(x_i)}{p_i} + \frac{\epsilon_i(1)}{p_i} - \mu_1(x_i) + \mu_0(x_i) \\
&= \tau(x_i) + \frac{1-p_i}{p_i}\mu_1(x_i) + \frac{1-p_i}{1-p_i}\mu_0(x_i) + \frac{\epsilon_i(1)}{p_i} \\
&= \tau(x_i) + (1-p_i)\left(\frac{1}{p_i}\mu_1(x_i) + \frac{1}{1-p_i}\mu_0(x_i)\right) + \frac{\epsilon_i(1)}{p_i} \\
&= \tau(x_i) + (1-p_i)h(x_i) + \frac{\epsilon_i(1)}{p_i}
\end{aligned}$$

Next, consider Y_i^* when $W_i = 0$,

$$\begin{aligned}
Y_i^* &= Y_i(0)\frac{0-p_i}{p_i(1-p_i)} \\
&= (\mu_0(x_i) + \epsilon_i(0))\frac{-1}{1-p_i} \\
&= \frac{-\mu_0(x_i)}{1-p_i} + \frac{-\epsilon_i(0)}{1-p_i} + \mu_1(x_i) - \mu_1(x_i) + \mu_0(x_i) - \mu_0(x_i) \\
&= \mu_1(x_i) - \mu_0(x_i) \\
&\qquad - \frac{p_i}{p_i}\mu_1(x_i) + \frac{1-p_i}{1-p_i}\mu_0(x_i) - \frac{1}{1-p_i}\mu_0(x_i) - \frac{\epsilon_i(0)}{1-p_i} \\
&= \tau(x_i) + \frac{-p_i}{p_i}\mu_1(x_i) + \frac{-p_i}{1-p_i}\mu_0(x_i) - \frac{\epsilon_i(0)}{1-p_i} \\
&= \tau(x_i) - p_i\left(\frac{1}{p_i}\mu_1(x_i) + \frac{1}{1-p_i}\mu_0(x_i)\right) + \frac{\epsilon_i(0)}{p_i - 1} \\
&= \tau(x_i) - p_i h(x_i) + \frac{\epsilon(0)}{p-1}.
\end{aligned}$$

Since the errors, $\epsilon_i(1)$ and $\epsilon_i(0) \sim N(0, \sigma^2)$, we have that Y_i^* is drawn from a

Gaussian mixture with

$$\begin{aligned}
&\mathrm{Y}_i^* = \tau(x_i) + \epsilon_i^*, \\
&\epsilon_i^* \sim p_i N((1-p_i)h(x_i), \frac{1}{p_i^2}\sigma^2) + (1-p_i)N(-p_i h(x_i), \frac{1}{(1-p_i)^2}\sigma^2).
\end{aligned}$$

□

B.3 Posterior Derivations

B.3.1 Derivation of $\tau|X, W, Y, p, h, (T^h, \mathcal{M}^h), (T^\tau, \mathcal{M}^\tau), \sigma^2$

Recall,

$$\begin{aligned}
Y_i^* &= \tau(x_i) + \epsilon_i^* \\
&= \sum_{m=1}^{M} T_{m_\tau}^{(\tau)}(x_i) + \epsilon_i^*
\end{aligned}$$

So, for Tree j, we are fitting

$$Y_i^* - \sum_{m \neq j}^{M_\tau} T_m^{(\tau)}(x_i) - (w_i(1-p_i)h(x_i) - (1-w_i)p_i h(x_i)) = T_j^{(\tau)}(x_i) + \epsilon_i$$

Where $\epsilon_i \sim p_i N(0, \frac{\sigma^2}{p_i^2}) + (1-p_i)N(0, \frac{\sigma^2}{(1-p_i)^2})$. Conditional on our stochastic tree process, our likelihood is

$$\begin{aligned}
\prod_{i=1}^{n} \Bigg[& p_i \frac{1}{\sqrt{2\pi}} \frac{1}{\frac{\sigma}{p_i}} \exp\left(-\frac{1}{2} \frac{\left(z_i^{(\tau)} - T_j^{(\tau)}(x_i) \right)^2}{\frac{\sigma^2}{p_i^2}} \right) \\
& + (1-p_i) \frac{1}{\sqrt{2\pi}} \frac{1}{\frac{\sigma}{1-p_i}} \exp\left(-\frac{1}{2} \frac{\left(z_i^{(\tau)} - T_j^{(\tau)}(x_i) \right)^2}{\frac{\sigma^2}{(1-p_i)^2}} \right) \Bigg]
\end{aligned}$$

where

$$z_i^{(\tau)} = Y_i^* - \sum_{m \neq j}^{M_\tau} T_m^{(\tau)}(x_i) - (w_i(1-p_i)h(x_i) - (1-w_i)p_i h(x_i)), \tag{B.3}$$

$T_j^{(\tau)}(x_i)$ is the value of the mean function for x_i for the j^{th} tree and $z_i^{(\tau)}$ is the value of Y_i^* after subtracting out the rest of the residuals explained by all other trees not including the j^{th} tree as well as the mixture means from equation B.2.

$$
\begin{aligned}
L(z_i^{(\tau)}|\sigma, T^h, \mathcal{M}^h, T^\tau, \mathcal{M}^\tau) = \\
\prod_{i=1}^{n} \Bigg(p_i \frac{1}{\sqrt{2\pi}} \frac{p_i}{\sigma} \exp\left(-\frac{1}{2} \frac{(z_i^{(\tau)} - T_j(x_i))^2}{\frac{\sigma^2}{p_i^2}} \right) \\
+ (1-p_i) \frac{1}{\sqrt{2\pi}} \frac{1-p_i}{\sigma} \exp\left(-\frac{1}{2} \frac{(z_i^{(\tau)} - T_j(x_i))^2}{\frac{\sigma^2}{(1-p_i)^2}} \right) \Bigg)
\end{aligned}
$$

Suppose there are B_j nodes in our j^{th} tree, then we can factor the above expression as

$$
\begin{aligned}
L(z_i^{(\tau)}|\sigma, T^h, \mathcal{M}^h, T^\tau, \mathcal{M}^\tau) = \\
\prod_{b=1}^{B_j} \prod_{i \in n_{j,b}^{(\tau)}} \Bigg(p_i \frac{1}{\sqrt{2\pi}} \frac{p_i}{\sigma} \exp\left(-\frac{1}{2} \frac{(z_i^{(\tau)} - \mu_{j,b}^{(\tau)})^2}{\frac{\sigma^2}{p_i^2}} \right) \\
+ (1-p_i) \frac{1}{\sqrt{2\pi}} \frac{1-p_i}{\sigma} \exp\left(-\frac{1}{2} \frac{(z_i^{(\tau)} - \mu_{j,b}^{(\tau)})^2}{\frac{\sigma^2}{(1-p_i)^2}} \right) \Bigg)
\end{aligned}
$$

where $\mu_{j,b}^{(\tau)}$ is the value of the mean function in node b and $n_{j,b}^{(\tau)}$ is the set of indices in node b. Under no unmeasured confounding, we have

$$
\begin{aligned}
L(z_i^{(\tau)}|\sigma, T^h, \mathcal{M}^h, T^\tau, \mathcal{M}^\tau) = \\
\prod_{b=1}^{B_j} \prod_{i \in n_{j,b}} \Bigg(w_i \frac{1}{\sqrt{2\pi}} \frac{p_i}{\sigma} \exp\left(-\frac{1}{2} \frac{(z_i^{(\tau)} - \mu_{j,b}^{(\tau)})^2}{\frac{\sigma^2}{p_i^2}} \right) \\
+ (1-w_i) \frac{1}{\sqrt{2\pi}} \frac{1-p_i}{\sigma} \exp\left(-\frac{1}{2} \frac{(z_i^{(\tau)} - \mu_{j,b}^{(\tau)})^2}{\frac{\sigma^2}{(1-p_i)^2}} \right) \Bigg)
\end{aligned}
$$

Here we introduce a change of notation to simplify the likelihood. Let

$$\sigma_{\tau,i}^2 = w_i \frac{\sigma^2}{p_i^2} + (1-w_i) \frac{\sigma^2}{(1-p_i)^2}. \tag{B.4}$$

Then we have,

$$L(z_i^{(\tau)}|\sigma, T^h, \mathcal{M}^h, T^\tau, \mathcal{M}^\tau) = \prod_{b=1}^{B_j} \prod_{i \in n_{j,b}} \left(\frac{1}{\sqrt{2\pi\sigma_{\tau,i}^2}} \exp\left(-\frac{1}{2} \frac{(z_i^{(\tau)} - \mu_{j,b}^{(\tau)})^2}{\sigma_{\tau,i}^2} \right) \right) \tag{B.5}$$

So, conditional on all the trees, including tree j and the scalar terminal node parameters, the leaf node parameters are independent. Below is the full conditional for each $\mu_{j,b}^{(\tau)}$ which can be drawn sequentially using a Gibbs step. Under the following prior,

$$\mu_{j,b}^{(\tau)} \sim N(\mu_0^{(\tau)}, \sigma_\tau^2)$$

by Theorem B.1,

$$\mu_{j,b}^{(\tau)}|\sigma^2, T, z_i^{(\tau)}, \sim N\left(\frac{\frac{\mu_0^{(\tau)}}{\sigma_\tau^2} + \sum_{i \in n_{j,b}} \frac{z_i^{(\tau)}}{\sigma_{\tau,i}^2}}{\frac{1}{\sigma_\tau^2} + \sum_{i \in n_{j,b}} \frac{1}{\sigma_{\tau,i}^2}}, \left[\frac{1}{\sigma_\tau^2} + \sum_{i \in n_{j,b}} \frac{1}{\sigma_{\tau,i}^2} \right]^{-1} \right).$$

B.3.2 Derivation of $h|X, W, Y, p, h, (T^h, \mathcal{M}^h), (T^{(\tau)}, \mathcal{M}^{(\tau)}), \sigma^2$

Recall,

$$\begin{aligned} Y_i^* &= \tau(x_i) + \epsilon_i^* \\ &= \tau(x_i) + w_i(1-p_i)h(x_i) - (1-w_i)p_i h(x_i) + \epsilon_i, \\ &= \tau(x_i) + w_i(1-p_i) \sum_{m=1}^{M_h} T_m^h(x_i) - (1-w_i)p_i \sum_{m=1}^{M_h} T_m^h(x_i) + \epsilon_i \end{aligned}$$

Where $\epsilon_i \sim p_i N(0, \frac{\sigma^2}{p_i^2}) + (1-p_i)N(0, \frac{\sigma^2}{(1-p_i)^2})$. So, for the $j^t h$ tree, we are fitting

$$Y_i^* - \tau(x_i) - \left(w_i(1-p_i) - (1-w_i)p_i \right) \sum_{m \neq j}^{M_h} T_m^h(x_i) = \left(w_i(1-p_i) - (1-w_i)p_i \right) T_j^h(x_i) + \epsilon_i$$

Multiplying the equation by $\left(w_i(1-p_i)-(1-w_i)p_i\right)^{-1}$, we have

$$\left(w_i(1-p_i)-(1-w_i)p_i\right)^{-1}$$
$$\left(Y_i^*-\tau(x_i)-\left(w_i(1-p_i)-(1-w_i)p_i\right)\sum_{m\neq j}^{M_h}T_m^h(x_i)\right)=$$
$$T_j^h(x_i)+\left(w_i(1-p_i)-(1-w_i)p_i\right)^{-1}\epsilon_i$$

Notice that,

$$\left(w_i(1-p_i)-(1-w_i)p_i\right)^{-1}\epsilon_i \sim p_i N(0,\frac{\sigma^2}{p_i^2(1-p_i)^2})+(1-p_i)N(0,\frac{\sigma^2}{p_i^2(1-p_i)^2})$$

We define

$$\sigma_{h,i}^2=\frac{\sigma^2}{p_i^2(1-p_i)^2}$$
and
$$z_i^{(h)}=\left(w_i(1-p_i)-(1-w_i)p_i\right)^{-1} \tag{B.6}$$
$$\left(Y_i^*-\tau(x_i)-\left(w_i(1-p_i)-(1-w_i)p_i\right)\sum_{m\neq j}^{M_h}T_m^h(x_i)\right)$$

Conditional on our stochastic tree process, our likelihood is

$$\prod_{i=1}^{n}\left[p_i\frac{1}{\sqrt{2\pi}}\frac{1}{\frac{\sigma}{p_i}}\exp\left(-\frac{1}{2}\frac{\left(z_i^{(h)}-T_j^{(h)}(x_i)\right)^2}{\sigma_{h,i}^2}\right)\right.$$
$$\left.+(1-p_i)\frac{1}{\sqrt{2\pi}}\frac{1}{\frac{\sigma}{1-p_i}}\exp\left(-\frac{1}{2}\frac{\left(z_i^{(h)}-T_j^{(h)}(x_i)\right)^2}{\sigma_{h,i}^2}\right)\right]$$

We can derive the conditional distribution for the leaf node mean for the j^{th} tree in node b for the function h, $\mu_{j,b}^{(h)}$, in a manner similar to the derivation for $\mu_{j,b}^{(\tau)}$ in section B.3.1. So, conditional on all the trees, including tree j and the scalar terminal node parameters, the leaf node parameters are independent. Below is the full conditional for each $\mu_{j,b}^{(h)}$ which can be drawn sequentially using a Gibbs step. Under the following prior,

$$\mu_{j,b}^{(h)} \sim N(\mu_0^{(h)}, \sigma_h^2)$$

by Theorem B.1,

$$\mu_{j,b}^{(h)}|\sigma^2, T, z_i^{(h)}, \sim N\left(\frac{\frac{\mu_0^{(\tau)}}{\sigma_\tau^2} + \sum_{i\in n_{j,b}} \frac{z_i^{(h)}}{\sigma_{h,i}^2}}{\frac{1}{\sigma_h^2} + \sum_{i\in n_{j,b}} \frac{1}{\sigma_{h,i}^2}}, \Big[\frac{1}{\sigma_h^2} + \sum_{i\in n_{j,b}} \frac{1}{\sigma_{h,i}^2}\Big]^{-1}\right).$$

B.3.3 Derivation of $\sigma^2|X, W, Y, p, h, g, (T^h, \mathcal{M}^h), (T^{(\tau)}, \mathcal{M}^{(\tau)})$

We start with the same likelihood as before. Let,

$$\mathcal{E}_i = Y_i^* - \tau(x_i) - \Big(w_i(1-p_i) - (1-w_i)p_i\Big)h(x_i),$$

then we can write our likelihood as

$$L(\mathcal{E}|\sigma, T^h, \mathcal{M}^h, T^\tau, \mathcal{M}^\tau) = \prod_{i=1}^{n}\left[p_i \frac{1}{\sqrt{2\pi}}\frac{1}{\frac{\sigma}{p_i}}\exp\left(-\frac{1}{2}\frac{\mathcal{E}_i^2}{\frac{\sigma^2}{p_i^2}}\right) + (1-p_i)\frac{1}{\sqrt{2\pi}}\frac{1}{\frac{\sigma}{1-p_i}}\exp\left(-\frac{1}{2}\frac{\mathcal{E}_i^2}{\frac{\sigma^2}{(1-p_i)^2}}\right)\right].$$

We assume a prior on σ^2 of the form

$$\sigma^2 \sim IG(a, c)$$

Applying the same approach as in the derivations for τ and h, we have

$$
\begin{aligned}
&L(\sigma^2|T^h, \mathcal{M}^h, T^\tau, \mathcal{M}^\tau, \mathcal{E}) \\
&\propto \prod_{i=1}^{n} \left[p_i \frac{1}{\sqrt{2\pi}} \frac{1}{\frac{\sigma}{p_i}} \exp\left(-\frac{1}{2} \frac{\mathcal{E}_i^2}{\frac{\sigma^2}{p_i^2}} \right) + (1-p_i) \frac{1}{\sqrt{2\pi}} \frac{1}{\frac{\sigma}{1-p_i}} \exp\left(-\frac{1}{2} \frac{\mathcal{E}_i^2}{\frac{\sigma^2}{(1-p_i)^2}} \right) \right] \\
&\qquad \left[\frac{c^a}{\Gamma(a)} (\sigma^{-2})^{-a-1} \exp\left(\frac{-c}{\sigma^2} \right) \right] \\
&\propto \prod_{i=1}^{n} \left[p_i \frac{p_i}{\sqrt{2\pi}} \frac{1}{\sigma} \exp\left(-\frac{1}{2} \frac{p_i^2 \mathcal{E}_i^2}{\sigma^2} \right) \right. \\
&\qquad \left. + (1-p_i) \frac{1-p_i}{\sqrt{2\pi}} \frac{1}{\sigma} \exp\left(-\frac{1}{2} \frac{(1-p_i)^2 \mathcal{E}_i^2}{\sigma^2} \right) \right] \\
&\qquad \left[\frac{c^a}{\Gamma(a)} (\sigma^{-2})^{-a-1} \exp\left(\frac{-c}{\sigma^2} \right) \right].
\end{aligned}
$$

Under no unmeasured counfounding, we have

$$
\begin{aligned}
&L(\sigma^2|T^h, \mathcal{M}^h, T^\tau, \mathcal{M}^\tau, \mathcal{E}) \\
&\propto \sigma^{-n} \exp\left(-\frac{\frac{1}{2} \sum_{i=1}^{n} (w_i p_i^2 + (1-w_i)(1-p_i)^2) \mathcal{E}_i^2}{\sigma^2} \right) \\
&\qquad \left[\frac{c^a}{\Gamma(a)} (\sigma^{-2})^{-a-1} \exp\left(\frac{-c}{\sigma^2} \right) \right].
\end{aligned}
$$

Therefor, σ^2 has full conditional of the form

$$
\sigma^2 \sim IG\left(\frac{n}{2} + a, c + \frac{1}{2} \sum_{i=1}^{n} (w_i p_i^2 + (1-w_i)(1-p_i)^2) \mathcal{E}_i^2 \right).
$$

B.4 Metropolis-Hastings Step Derivations

B.4.1 Marginalized leaf node parameters for $\tau(x)$

Recall,

$$z_i^{(\tau)} = \left(Y_i^* - \sum_{m \neq j}^{M_\tau} T_m^{(\tau)}(x_i) - (w_i(1-p_i)h(x_i) - (1-w_i)p_i h(x_i)) \right)$$

and

$$\sigma^2_{\tau,i} = w_i \frac{\sigma^2}{p_i^2} + (1-w_i)\frac{\sigma^2}{(1-p_i)^2}$$

from equations B.3 and B.4. We can use the derivation in B.4.3 to get the marginalized likelihood associated with the leaf node parameters for $\tau(x)$ if $\tau = \delta$ and $h = \gamma$.

B.4.2 Marginalized leaf node parameters for $h(x)$

Recall,

$$\sigma^2_{h,i} = \frac{\sigma^2}{p_i^2(1-p_i)^2}$$

and

$$z_i^{(h)} = \left(w_i(1-p_i) - (1-w_i)p_i \right)^{-1} \left(Y_i^* - \tau(x_i) - \left(w_i(1-p_i) - (1-w_i)p_i \right) \sum_{m \neq j}^{M_h} T_m^h(x_i) \right)$$

from equation B.6. We can use the derivation in B.4.3 to get the marginalized likelihood associated with the leaf node parameters for $h(x)$ if $h = \delta$ and $\tau = \gamma$.

B.4.3 Marginalizing the portion of the likelihood associated with the leaf nodes.

We begin with our factorized likelihood. Regarding the functions τ and h, we refer to the function who's tree, j, leaf node parameter, b, is being

marginalized as δ and the other function as γ. Let $n_{j,b}^{(\delta)}$ be the set of indices for our observed data in node b in $T_j^{(\delta)}$. Then we have

$$L(Y_i^*; \sigma^2, T_{j\notin M_\delta}^{(\delta)}, T^{(\gamma)}) = \int_{-\infty}^{\infty} L(Y_i^* | \mu_{j,b}^{(\delta)}, \sigma^2, T_{j\notin M_\delta}^{(\delta)}, T^{(\gamma)}) \pi(\mu_{j,b}^{(\delta)}) d\mu_{j,b}^{(\delta)}$$

$$=$$

$$\int_{-\infty}^{\infty} \prod_{i \in n_{j,b}^{(\delta)}} \left(w_i \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_{\delta,i}} \exp\left(-\frac{1}{2} \frac{(z_i^{(\delta)} - \mu_{j,b}^{(\delta)})^2}{\sigma_{\delta,i}^2} \right) + \right.$$

$$\left. (1 - w_i) \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_{\delta,i}} \exp\left(-\frac{1}{2} \frac{(z_i^{(\delta)} - \mu_{j,b}^{(\delta)})^2}{\sigma_{\delta,i}^2} \right) \right)$$

$$\frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_\delta} \exp\left(-\frac{1}{2} \frac{(\mu_{j,b}^{(\delta)} - \mu_0^{(\delta)})^2}{\sigma_\delta^2} \right) d\mu_{j,b}^{(\delta)}$$

$$=$$

$$\int_{-\infty}^{\infty} \prod_{i \in n_{j,b}^{(\delta)}} \left(\frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_{\delta,i}} \exp\left(-\frac{1}{2} \frac{(z_i^{(\delta)} - \mu_{j,b}^{(\delta)})^2}{\sigma_{\delta,i}^2} \right) \right)$$

$$\frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_\delta} \exp\left(-\frac{1}{2} \frac{(\mu_{j,b}^{(\delta)} - \mu_0^{(\delta)})^2}{\sigma_\delta^2} \right) d\mu_{j,b}^{(\delta)}$$

$$
=
$$

$$
\int_{-\infty}^{\infty} (2\pi)^{-n/2} \Big(\prod_{i \in n_{j,b}^{(\delta)}} \sigma_{\delta,i}^{-1} \Big) \exp\left(\frac{-1}{2} \sum_{i \in n_{j,b}^{(\delta)}} \frac{\left(z_i^{(\delta)} - \mu_{j,b}^{(\delta)} \right)^2}{\sigma_{\delta,i}^2} \right)
$$

$$
(2\pi)^{-1/2} \sigma_\delta^{-1} \exp\left(\frac{-1}{2} \frac{(\mu_{j,b}^{(\delta)} - \mu_0^{(\delta)})^2}{\sigma_\delta^2} \right) d\mu_{j,b}^{(\delta)}
$$

$$
=
$$

$$
\int_{-\infty}^{\infty} (2\pi)^{-(n+1)/2} \sigma_\delta^{-1} \Big(\prod_{i \in n_{j,b}^{(\delta)}} \sigma_{\delta,i}^{-1} \Big)
$$

$$
\exp\left(\frac{-1}{2} \Big(\sum_{i \in n_{j,b}^{(\delta)}} \frac{z_i^{(\delta)2}}{\sigma_{\delta,i}^2} - 2\mu_{j,b}^{(\delta)} \sum_{i \in n_{j,b}^{(\delta)}} \frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2} + \sum_{i \in n_{j,b}^{(\delta)}} \frac{\mu_{j,b}^{(\delta)2}}{\sigma_{\delta,i}^2} \right.
$$

$$
\left. \left. + \frac{\mu_{j,b}^{(\delta)2} - 2\mu_{j,b}^{(\delta)} \mu_0^{(\delta)} + \mu_0^{(\delta)2}}{\sigma_\delta^2} \Big) \right) d\mu_{j,b}^{(\delta)}
$$

$$
=
$$

$$
\int_{-\infty}^{\infty} (2\pi)^{-(n+1)/2} \sigma_\delta^{-1} \Big(\prod_{i \in n_{j,b}^{(\delta)}} \sigma_{\delta,i}^{-1} \Big) \exp\left(\frac{-1}{2} \Big[\sum_{i \in n_{j,b}^{(\delta)}} \frac{z_i^{(\delta)2}}{\sigma_{\delta,i}^2} + \frac{\mu_0^{(\delta)2}}{\sigma_\delta^2} \Big] \right)
$$

$$
\exp\left(\frac{-1}{2} \left[\left(\sigma_\delta^{-2} + \sum_{i \in n_{j,b}^{(\delta)}} \sigma_{\delta,i}^{-2} \right) \mu_{j,b}^{(\delta)2} - 2 \Big(\sum_{i \in n_{j,b}^{(\delta)}} \frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2} + \frac{\mu_0^{(\delta)}}{\sigma_\delta^2} \Big) \mu_{j,b}^{(\delta)} \right] \right) d\mu_{j,b}^{(\delta)}
$$

$$=$$

$$\int_{-\infty}^{\infty}(2\pi)^{-(n+1)/2}\sigma_{\delta}^{-1}(\prod_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-1})\exp\left(\frac{-1}{2}\Big[\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)2}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)2}}{\sigma_{\delta}^2}\Big]\right)$$

$$\exp\left(\frac{-1}{2}\Bigg[\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)$$

$$\left(\mu_{j,b}^{(\delta)2}-2\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-1}\left(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_{\delta}^2}\right)\mu_{j,b}^{(\delta)}\right.$$

$$+\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-2}\left(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_{\delta}^2}\right)^2$$

$$\left.-\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-2}\left(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_{\delta}^2}\right)^2\right)\Bigg]\Bigg)d\mu_{j,b}^{(\delta)}$$

$$=$$

$$\int_{-\infty}^{\infty}(2\pi)^{-(n+1)/2}\sigma_{\delta}^{-1}(\prod_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-1})\exp\left(\frac{-1}{2}\Big[\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)2}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)2}}{\sigma_{\delta}^2}\Big]\right)$$

$$\exp\left(\frac{1}{2}\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-2}\left(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_{\delta}^2}\right)^2\right)$$

$$\exp\left(\frac{-1}{2}\Bigg[\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)\right.$$

$$\left(\mu_{j,b}^{(\delta)2}-2\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-1}\left(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_{\delta}^2}\right)\mu_{j,b}^{(\delta)}+\right.$$

$$\left.\left.\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-2}\left(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_{\delta}^2}\right)^2\Bigg]\right)d\mu_{j,b}^{(\delta)}$$

$$=$$

$$(2\pi)^{-(n+1)/2}\sigma_\delta^{-1}(\prod_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-1})\exp\left(\frac{-1}{2}\Big[\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)2}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)2}}{\sigma_\delta^2}\Big]\right)$$

$$\exp\left(\frac{1}{2}\Big(\sigma_\delta^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\Big)^{-1}\Big(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_\delta^2}\Big)^2\right)$$

$$\int_{-\infty}^{\infty}\exp\left(\frac{-1}{2}\Bigg[\Big(\sigma_\delta^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\Big)\right.$$

$$\Big(\mu_{j,b}^{(\delta)2}-2\Big(\sigma_\delta^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\Big)^{-1}\Big(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_\delta^2}\Big)\mu_{j,b}^{(\delta)}+$$

$$\left.\Big(\sigma_\delta^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\Big)^{-2}\Big(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_\delta^2}\Big)^2\Bigg]\right)d\mu_{j,b}^{(\delta)}$$

$$=$$

$$(2\pi)^{-(n+1)/2}\sigma_\delta^{-1}(\prod_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-1})\exp\left(\frac{-1}{2}\Big[\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)2}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)2}}{\sigma_\delta^2}\Big]\right)$$

$$\exp\left(\frac{1}{2}\Big(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^2}+\frac{\mu_0^{(\delta)}}{\sigma_\delta^2}\Big)^2\right)$$

$$(2\pi)^{1/2}\Big(\sigma_\delta^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\Big)^{-1/2}$$

$$=$$

$$\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-1/2}(2\pi)^{-n_{j,b}^{*}/2}\sigma_{\delta}^{-1}\prod_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-1}$$

$$\exp\left(-\frac{1}{2}\left[\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)2}}{\sigma_{\delta,i}^{2}}+\frac{\mu_0^{(\delta)2}}{\sigma_{\delta}^{2}}\right]\right.$$

$$\left.+\frac{1}{2}\left(\sigma_{\delta}^{-2}+\sum_{i\in n_{j,b}^{(\delta)}}\sigma_{\delta,i}^{-2}\right)^{-1}\left(\sum_{i\in n_{j,b}^{(\delta)}}\frac{z_i^{(\delta)}}{\sigma_{\delta,i}^{2}}+\frac{\mu_0^{(\delta)}}{\sigma_{\delta}^{2}}\right)^{2}\right)$$

Leaving us with the marginalized portion of the likelihood associated with $\eta_{j,b}^{*}$

B.5 Summary Tables of Cases A-D

Case A								
	ATE				CATE			
Method	rmse	bias	cov	mil	rmse	bias	cov	mil
CBARTMM25	11.41	11.04	0.06	7.06	49.47	-11.04	0.74	67.78
BCF	3.94	-0.6	0.42	4.93	41.58	-0.6	0.65	26.61
BART	6.39	3.28	0.9	16.76	58.9	3.28	0.87	100.58
TOT	7.2	-2.28	0.66	14.28	83.16	-2.28	0.87	158.42
BART TRV	12.09	7.31	0.96	33.27	78.16	-10.64	0.78	167.28

Case B								
	ATE				CATE			
Method	rmse	bias	cov	mil	rmse	bias	cov	mil
CBARTMM25	0.68	-0.21	0.39	0.79	4.38	0.21	0.68	8.74
BCF	0.96	0.13	0.95	2.82	3.52	0.13	0.94	12.1
BART	0.92	0.06	0.9	2.42	5.71	0.06	0.97	23.8
TOT	1.95	1.08	0.98	5.64	6.32	0.9	0.57	24.15
BART TRV	1.97	-0.29	0.88	5.29	9.05	-0.29	0.96	37.0

Case C								
	ATE				CATE			
Method	rmse	bias	cov	mil	rmse	bias	cov	mil
CBARTMM25	0.82	0.8	0.06	0.49	1.91	-0.8	0.55	1.96
BCF	0.24	-0.07	0.9	0.63	1.18	-0.07	0.68	1.98
BART	0.21	0.02	0.97	0.64	0.9	0.02	0.94	3.26
TOT	1.09	0.92	0.95	2.29	2.81	-2.04	0.98	8.55
BART TRV	1.21	-0.09	0.7	2.48	11.06	-0.09	0.87	19.87

Case D								
	ATE				CATE			
Method	rmse	bias	cov	mil	rmse	bias	cov	mil
CBARTMM25	0.71	0.69	0.08	0.42	1.77	-0.69	0.6	1.57
BCF	0.11	-0.03	0.83	0.25	0.38	-0.03	0.98	0.52
BART	0.08	-0.01	1.0	0.27	0.13	-0.01	1.0	1.26
TOT	2.97	2.9	0.0	2.51	2.48	-2.0	0.99	9.14
BART TRV	1.15	-0.07	0.6	2.08	11.91	-0.07	0.84	17.25

Bibliography

Ahn, S., Shahbaba, B., and Welling, M. (2014). Distributed stochastic gradient mcmc. In Xing, E. P. and Jebara, T., editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1044–1052, Bejing, China. PMLR.

Akyildiz, D., Crisan, D., and Míguez, J. (2020). Parallel sequential monte carlo for stochastic gradient-free nonconvex optimization. *Statistics and computing*, 30(6):1645–1663.

Athey, S. and Imbens, G. (2015a). Machine learning for estimating heterogeneous causal effects. Research papers, Stanford University, Graduate School of Business.

Athey, S. and Imbens, G. (2015b). Recursive partitioning for heterogeneous causal effects.

Athey, S., Imbens, G., and Kong, Y. (2016). *causalTree: Recursive Partitioning Causal Trees*. R package version 0.0.

Beygelzimer, A. and Langford, J. (2016). The offset tree for learning with partial labels.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984a). *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA. new edition **?**?

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984b). *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA.

Casarin, R., Craiu, R., and Leisen, F. (2015a). Embarrassingly parallel sequential markov-chain monte carlo for large sets of time series. *Statistics and Its Interface*, 9.

Casarin, R., Grassi, S., Ravazzolo, F., and van Dijk, H. K. (2015b). Parallel sequential monte carlo for efficient density combination: The deco matlab toolbox. *Journal of Statistical Software, Articles*, 68(3):1–30.

Chipman, H. A., George, E. I., and McCulloch, R. E. (1998). Bayesian cart model search. *Journal of the American Statistical Association*, 93(443):935–948.

Chipman, H. A., George, E. I., and McCulloch, R. E. (2010). Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1).

Dudík, M., Erhan, D., Langford, J., and Li, L. (2014). Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4).

Falcon, A. (2015). The stanford encyclopedia of philosophy.

Foster, J. C., Taylor, J. M., and Ruberg, S. J. (2011). Subgroup identification from randomized clinical trial data. *Statistics in Medicine*, 30(24):2867–2880.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, (6):721–741.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F - Radar and Signal Processing*, 140(2):107–113.

Green, D. P. and Kern, H. L. (2012). Modeling heterogeneous treatment effects in survey experiments with bayesian additive regression trees. *The Public Opinion Quarterly*, 76(3):491–511.

Hahn, P. R., Murray, J. S., and Carvalho, C. (2019). Bayesian regression tree models for causal inference: regularization, confounding, and heterogeneous effects.

Hahn, P. R., Murray, J. S., and Carvalho, C. M. (2017). Bayesian regression tree models for causal inference: regularization, confounding, and heterogeneous effects.

Hastie, T. and Tibshirani, R. (2000). Bayesian backfitting. *Statistical Science*, 15(3):196–213.

Heckman, J. J., Urzua, S., and Vytlacil, E. (2006). Understanding Instrumental Variables in Models with Essential Heterogeneity. *The Review of Economics and Statistics*, 88(3):389–432.

Hill, J., Linero, A., and Murray, J. (2020). Bayesian additive regression trees: A review and look forward. *Annual Review of Statistics and Its Application*, 7(1):251–278.

Hill, J. L. (2011). Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics*, 20(1):217–240.

Hirano, K., Imbens, G. W., and Ridder, G. (2003). Efficient estimation of average treatment effects using the estimated propensity score. *Econometrica*, 71(4):1161–1189.

Imbens, G. W. and Rubin, D. B. (2015). *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. Cambridge University Press.

Kitagawa, G. (1996). Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*.

Kleiner, A., Talwalkar, A., Sarkar, P., and Jordan, M. I. (2012). The big data bootstrap. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 1787–1794, Madison, WI, USA. Omnipress.

Knaus, M. C., Lechner, M., and Strittmatter, A. (2020). Machine learning estimation of heterogeneous causal effects: Empirical monte carlo evidence. *The Econometrics Journal*, 24(1):134–161.

Ko, S., Zhou, H., Zhou, J. J., and Won, J.-H. (2021). High-performance statistical computing in the computing environments of the 2020s.

Lee, A. and Whiteley, N. (2016). Forest resampling for distributed sequential monte carlo. *Statistical analysis and data mining*, 9(4):230–248.

Lindsten, F., Johansen, A. M., Naesseth, C. A., Kirkpatrick, B., Schön, T. B., Aston, J. A. D., and Bouchard-Côté, A. (2017). Divide-and-conquer with sequential monte carlo. *Journal of Computational and Graphical Statistics*, 26(2):445–458.

Linero, A. R. (2018). Bayesian regression trees for high-dimensional prediction and variable selection. *Journal of the American Statistical Association*, 113(522):626–636.

Linero, A. R. and Yang, Y. (2018). Bayesian regression tree ensembles that adapt to smoothness and sparsity. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 80(5):1087–1110.

Liu, Y., Ročková, V., and Wang, Y. (2021). Variable selection with abc bayesian forests. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. B. (2014). Scalable and robust bayesian inference via the median posterior. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page II–1656–II–1664. JMLR.org.

Murray, J. S. (2021). Log-linear bayesian additive regression trees for multinomial logistic and count regression models. *Journal of the American Statistical Association*, 0(0):1–14.

Míguez, J. and Vázquez, M. A. (2016). A proof of uniform convergence over time for a distributed particle filter. *Signal Processing*, 122:152–163.

Nam, C. F. H., Aston, J. A. D., and Johansen, A. M. (2014). Parallel sequential monte carlo samplers and estimation of the number of states in a hidden markov model. *Annals of the Institute of Statistical Mathematics*, 66(3):553–575.

Neiswanger, W., Wang, C., and Xing, E. P. (2014). Asymptotically exact, embarrassingly parallel mcmc. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, UAI'14, page 623–632, Arlington, Virginia, USA. AUAI Press.

Pearl, J. (2009). *Causality: Models, Reasoning and Inference*. Cambridge University Press, USA, 2nd edition.

Powers, S., Qian, J., Jung, K., Schuler, A., Shah, N. H., Hastie, T., and Tibshirani, R. (2018). Some methods for heterogeneous treatment effect estimation in high dimensions. *Statistics in Medicine*, 37(11):1767–1787.

Pratola, M. T., Chipman, H. A., George, E. I., and McCulloch, R. E. (2020). Heteroscedastic bart via multiplicative regression trees. *Journal of Computational and Graphical Statistics*, 29(2):405–417.

Read, J., Achutegui, K., and Míguez, J. (2014). A distributed particle filter for nonlinear tracking in wireless sensor networks. *Signal Processing*, 98:121 – 134.

Reumann, M., Makalic, E., Goudey, B. W., Inouye, M., Bickerstaffe, A., Bui, M., Park, D. J., Kapuscinski, M. K., Schmidt, D. F., Zhou, Z., Qian, G., Zobel, J., Wagner, J., and Hopper, J. L. (2012). Supercomputing enabling exhaustive statistical analysis of genome wide association study data: Preliminary results. In *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1258–1261.

Rubin, D. B. (1978). Bayesian Inference for Causal Effects: The Role of Randomization. *The Annals of Statistics*, 6(1):34 – 58.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016). Bayes and big data: the consensus monte carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88.

Sparapani, R., Logan, B. R., McCulloch, R. E., and Laud, P. W. (2018). Nonparametric competing risks analysis using bayesian additive regression trees (bart).

Sparapani, R. A., Logan, B. R., McCulloch, R. E., and Laud, P. W. (2016). Nonparametric survival analysis using bayesian additive regression trees (bart). *Statistics in medicine*, 35(16):2741–2753. 26854022[pmid].

Srivastava, S., Li, C., and Dunson, D. B. (2018). Scalable bayes via barycenter in wasserstein space. *Journal of Machine Learning Research*, 19(8):1–35.

Starling, J. E., Murray, J. S., Carvalho, C. M., Bukowski, R. K., and Scott, J. G. (2020a). BART with targeted smoothing: An analysis of patient-specific stillbirth risk. *The Annals of Applied Statistics*, 14(1):28 – 50.

Starling, J. E., Murray, J. S., Lohr, P. A., Aiken, A. R. A., Carvalho, C. M., and Scott, J. G. (2020b). Targeted smooth bayesian causal forests: An analysis of heterogeneous treatment effects for simultaneous versus interval medical abortion regimens over gestation.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288.

Verge, Dubarry, D. M. and Moulines (2015). On parallel implementation of sequential monte carlo methods: the island particle model. *Statistics and computing*, 25(2):243–260.

Wager, S. and Athey, S. (2018). Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523):1228–1242.

Wager, S., Hastie, T., and Efron, B. (2014). Confidence intervals for random forests: The jackknife and the infinitesimal jackknife. *Journal of machine learning research : JMLR*, 15(1):1625–1651. 25580094[pmid].

Wang, X. and Dunson, D. B. (2013). Parallelizing MCMC via Weierstrass Sampler. *arXiv e-prints*, page arXiv:1312.4605.

Wendling, T., Jung, K., Callahan, A., Schuler, A., Shah, N., and Gallego, B. (2018). Comparing methods for estimation of heterogeneous treatment effects using observational data from health care databases. *Statistics in Medicine*, 37(23):3309–3324.

Whiteley, N., Lee, A., and Heine, K. (2016). On the role of interaction in sequential monte carlo algorithms. *Bernoulli*, 22(1):494–529.

Xie, Y., Brand, J. E., and Jann, B. (2012). Estimating heterogeneous treatment effects with observational data. *Sociological methodology*, 42(1):314–347. 23482633[pmid].

Zaidi, A. and Mukherjee, S. (2018). Gaussian process mixtures for estimating heterogeneous treatment effects.

Zhou, Y. (2015). vsmc: Parallel sequential monte carlo in c++. *Journal of Statistical Software, Articles*, 62(9):1–49.

The authors acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC, visualization, database, or grid resources that have contributed to the research results reported within this paper. URL: http://www.tacc.utexas.edu