Bayesian Analysis (2016) **11**, Number 3, pp. 885–911

Efficient Metropolis–Hastings Proposal Mechanisms for Bayesian Regression Tree Models*

Matthew T. Pratola†

Abstract. Bayesian regression trees are flexible non-parametric models that are well suited to many modern statistical regression problems. Many such tree models have been proposed, from the simple single-tree model to more complex tree ensembles. Their nonparametric formulation allows one to model datasets exhibiting complex non-linear relationships between the model predictors and observations. However, the mixing behavior of the Markov Chain Monte Carlo (MCMC) sampler is sometimes poor, frequently suffering from local mode stickiness and poor mixing. This is because existing Metropolis–Hastings proposals do not allow for efficient traversal of the model space. We develop novel Metropolis–Hastings proposals that account for the topological structure of regression trees. The first is a novel tree rotation proposal that only requires local changes to the regression tree structure, yet efficiently traverses disparate regions of the model space along contours of high likelihood. The second is a rule perturbation proposal which can be seen as an efficient variation of the change proposal found in existing literature. We implement these samplers and demonstrate their effectiveness on a prediction problem from computer experiments, a test function where structural tree variability is needed to fully explore the posterior and data from a heart rate study.

Keywords: Markov chain Monte Carlo, proposal distribution, computer experiments, uncertainty quantification, credible interval, coverage probability.

1 Introduction

Regression tree approaches to modeling complex nonlinear relationships have enjoyed increasing popularity in the statistical literature in recent years under the guise of Bayesian formulations (e.g. Chipman et al., 1998, 2002; Denison et al., 1998). A variety of such Bayesian formulations have been developed, from single-tree models (Chipman et al., 1998, 2002; Denison et al., 1998), treed Gaussian Process models (Gramacy and Lee, 2008), sequential regression trees (Taddy et al., 2011) and Bayesian Additive Regression Trees (BART) (Chipman et al., 2010). Recent work also indicates that efficient and scalable parallel versions of regression tree models are possible (Pratola et al., 2014), which is timely given the explosion in the size and complexity of modern datasets.

The benefits of regression tree models are well known; they allow for a flexible modeling approach that can handle a wide variety of nonlinear problems, have a simple

*Related articles: DOI: 10.1214/16-BA999A, DOI: 10.1214/16-BA999B, DOI: 10.1214/16-BA999H; rejoinder at DOI: 10.1214/16-BA999REJ.

†Department of Statistics, The Ohio State University, mpratola@stat.osu.edu

 DOI: 10.1214/16-BA999

and easy to understand structure, and offer fast predictive performance. At the same time, none of the drawbacks of conventional basis function approaches are present; for instance, no specification of a basis is required, and the computationally expensive matrix-algebra associated with basis function approaches is absent. The promise of Bayesian formulations of regression trees is to combine the benefits of regression tree models with the benefits of the Bayesian modeling paradigm, namely accounting for various sources of uncertainty which are then propagated through to the posterior predictive distribution.

In practice, the aspirations of Bayesian regression tree methodologies are realized in many applications, but there are some problems that arise in certain cases. Primarily, it is well known that the Metropolis–Hastings (MH) proposals in the Markov Chain Monte Carlo (MCMC) sampler of these models can suffer from poor mixing (Wu et al., 2007), resulting in overfitting the data and under-representing model uncertainty. A few approaches that go some way towards mitigating this issue have appeared in the literature. The method of Taddy et al. (2011) approaches the problem by using a particle-based representation of the unknown posterior distribution. The BART model of Chipman et al. (2010) forms a sum-of-trees representation of the data, where each tree is penalized to have shallow depth. The idea is that with shallow trees, the simple MH proposals that sometimes failed to explore the model space when used with deep trees will be adequate due to the vastly reduced search space.

Gramacy and Lee (2008) improve mixing of a Bayesian treed Gaussian Process model by applying the rotation algorithm from the Binary Search Tree literature (e.g. Sleator et al., 1988). They show that such a move has high probability of acceptance as its likelihood ratio is always 1. However, it requires that all 3 internal nodes involved in a rotation split on the same variable. This constraint in general will usually not be satisfied in Bayesian regression trees.

Another approach to improve mixing is the proposed "radical restructure" MH proposal developed in Wu et al. (2007). Their result suggested that mixing problems could be eliminated when their restructure proposal was combined with the usual proposal mechanisms previously developed in the literature. However, their proposal is computationally expensive and does not scale well with high-dimensional problems (i.e. large number of covariates, d) due to the curse of dimensionality. Additionally, in Section 5.1 we demonstrate a situation where the sampler of Wu et al. (2007) does not sample all trees consistent with the data. The limitation in this case appears to be the inherent restriction on tree dimensionality with their proposal. In contrast, the samplers developed in this article do sample all trees consistent with the data.

Alternatives to MH samplers have also been recently explored. Lakshminarayanan et al. (2013) propose a Sequential Monte Carlo approach for a single decision tree model and empirically demonstrate similar performance as MCMC methods. Their ideas are extended to BART in Lakshminarayanan et al. (2015) using a particle Gibbs algorithm.

Our work was motivated by examples from disparate regression tree models that exhibited poor mixing and severely underestimated posterior predictive uncertainty. The solutions we develop in this paper include a vastly more efficient version of the

classical "change" proposal (Chipman et al., 1998; Denison et al., 1998) and a novel proposal mechanism that enables efficient searching of the tree space, thereby allowing the Markov chain to mix adequately leading to accurate representation of model uncertainty. The proposed samplers are applied to the motivating examples to demonstrate the improvements realized. The samplers developed are applicable to Bayesian regression tree models in general and should yield improvements similar to what we demonstrate on a wide variety of applied problems.

In the next section, we motivate our development with a synthetic example from Wu et al. (2007) using a single-tree model, and a simple example from Computer Experiments (Sacks et al., 1989; Kennedy and O'Hagan, 2001; Oakley and O'Hagan, 2002; Higdon et al., 2008; Gramacy and Lee, 2008) using the BART model. In Section 3, we develop our novel tree rotation proposal mechanism. In Section 4 we introduce our "perturb" move which is an enhanced version of the "change" proposal mechanism. In Section 5, we apply the new samplers to the motivating problems to demonstrate the improvements realized. Finally, we conclude in Section 6. Additional examples (a heart rate study and a computer model calibration example) as well as methodology details may be found in the online Supplementary Material (Pratola, 2016).

2 Background and Motivating Examples

An acknowledged challenge of Bayesian regression tree models has been the difficulty to sometimes achieve proper mixing of the Markov chain. And, while this problem has been recognized since such models were established (Chipman et al., 1998; Denison et al., 1998), little progress has been made. Today, the majority of implementations continue to rely on the birth/death/change/swap proposals that were originally described. A notable exception to this is the work of Wu et al. (2007).

Regression trees model data using a stochastic binary tree representation made up of interior nodes, T, and a set of maps, M, associated with the terminal nodes. Since the tree is binary, any interior node, say node η_i, always has a left and right child, denoted $l(\eta_i), r(\eta_i)$ respectively. All nodes except for the root have a parent node, $p(\eta_i)$. Frequently, it is common to refer to a node by its unique integer identifier i. For example, the root node η_1 is node 1. In this paper, we will sometimes also refer to a subtree starting at node η_i simply as T_i. The tree shown in Figure 1 summarizes our notation.

What does T represent? Each internal node of a regression tree contains a split rule that depends on some covariate, and a split location, or "cutpoint". The representation T is abstract, by which we mean that one might be referring to the tree T or one might be referring to this modeling structure encoded in T. This modeling structure is simply the parameterization of the split rules at each node in the tree and the topological arrangement of nodes and edges forming the tree. Consider the $n \times d$ design matrix X of covariates for our data. Each of the d columns represents a covariate variable $v, v = 1, \ldots, d$ and each row x corresponds to the observed settings of these covariates. Without loss of generality, assume that the covariates are scaled to the unit interval, so that $x_v \in [0, 1]$ and $x \in [0, 1]^d$. Then the split rule at a given interior tree node is of the form $x_v < c$ which is parameterized by the chosen split variable v and the cutpoint c.

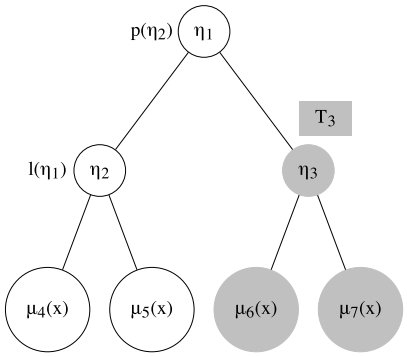

Figure 1: A sample binary regression tree with internal nodes $T = \{\eta_1, \eta_2, \eta_3\}$ and bottom-node maps $M = \{\mu_4(x), \mu_5(x), \mu_6(x), \mu_7(x)\}$. Each internal node has a rule $v < c$ for some variable v and cutpoint c. When a rule is true for an input x, branch left, otherwise branch right. Input x maps to $\mu_5(x)$ if it branches left at η_1 and right at η_2. Node η_2 is also the left child of the root, $l(\eta_1)$, while the root η_1 is also the parent $p(\eta_2)$. The sub-tree T_3 denoted in grey consists of node η_3 and all its children.

The stochastic regression tree representation arises by treating the split variable v as a discrete random quantity in $\{1, \ldots, d\}$ and the cutpoint c as a discrete random quantity in $\{0, \frac{1}{n_v - 1}, \ldots, \frac{n_v - 2}{n_v - 1}, 1\}$ where n_v is the resolution of discretization for variable v. For a continuous covariate, $n_v = 100$ is common, while for a discrete covariate this would be adjusted accordingly. The n_v's are usually specified as fixed, known. The modeling representation of T is often expressed as $T = \{(v_1, c_1), (v_2, c_2), \ldots\}$.

The regression tree is completed by specifying the maps at terminal nodes. For $n_b = |M|$ terminal nodes we have maps $M = \{\mu_1, \ldots, \mu_{n_b}\}$. These maps take as input the covariates x mapping to a given terminal node and produce a response $\mu_j(x)$. Common forms of the μ_j's are constants (i.e. $\mu_j(x) \equiv \mu_j$), linear regression models, Gaussian Processes (Gramacy and Lee, 2008), etc. Taken all together, T represents a partitioning of the covariate space χ and a mapping from an input covariate $x \in \chi$ to a response value encoded in M. To be more exact, the regression tree defines a function $g(x; T, M)$ which maps input x to response $\mu_j(x)$.

2.1 Bayesian Tree Models

The simplest Bayesian models (Chipman et al., 1998; Denison et al., 1998) implementing the above stochastic binary tree models the data using a single tree as

$$y = g(x; T, M) + \epsilon, \quad \epsilon \sim N(0, \sigma^2) \tag{1}$$

and uses priors of the form $\pi(T, M, \sigma^2) = \pi(M|T, \sigma^2)\pi(T)\pi(\sigma^2)$. A conjugate normal distribution is used to specify the prior on bottom-node μ's as well as for σ^2, so these parameters are drawn using the Gibbs sampler. The conjugacy also allows the marginal $T|y, \sigma^2$ to be expressed in closed-form by integrating out M. Drawing from $T|y, \sigma^2$

is then sampled using Metropolis–Hastings proposals, and is what poses challenges in Bayesian tree models.

In the BART model of Chipman et al. (2010), the idea is to represent the data y as a sum of simple trees,

$$y(x) = \sum_{j=1}^{m} g(x; T_j, M_j) + \epsilon, \quad \epsilon \sim N(0, \sigma^2) \tag{2}$$

and the priors have the form $\prod_{j=1}^{m} \pi(M_j|T_j, \sigma^2)\pi(T_j)\pi(\sigma^2)$. These simple trees are implemented by specifying constants for the terminal node maps, $M_j = (\mu_{j1}, \mu_{j2}, \ldots)$, and using a prior that penalizes the depth of each tree. The prior $\pi(T_j)$ thus favors shallow trees, or parsimony, over deep and complex trees. This parsimony is achieved by specifying the prior probability that a node η_{jk} at depth d_{jk} of tree T_j is non-terminal to be $\pi(\eta_{jk}) \propto \alpha(1 + d_{jk})^{-\beta}$ for $\alpha \in (0, 1)$ and $\beta \in [0, \infty)$.

The default number of trees used in this representation is $m = 200$, which seems to work well for a wide variety of problems. Conjugate normal priors on the terminal node μ_{jk}'s again leads to a standard Gibbs sampler for the terminal node maps. Similarly, a conjugate inverse chi-squared prior for σ^2 results in simple Gibbs updates for the variance. Discrete uniform priors are placed on the split variables and split cutpoints and combined with the prior on the probability that a node is non-terminal at a given depth leads to a Metropolis–Hastings algorithm for sampling from $\pi(T_j|y, \sigma^2)$ by growing or pruning nodes in the tree. This growing and pruning are handled by aptly named birth and death proposals which either split a currently terminal node on some variable v at some cutpoint c, or collapses two terminal nodes thereby removing a split.

In addition to the birth/death MH steps, which allows the parameter space of the nodes at the bottom of the trees to be explored by the MCMC algorithm, there are additional change and swap proposals aimed at exploring the parameter space of nodes that are internal to the tree. These four MH proposal mechanisms are summarized in Figure 11 of the Supplementary Material. For complete details of these MCMC algorithms, the reader is referred to Chipman et al. (1998); Denison et al. (1998); Chipman et al. (2010).

Our interest is to ensure good mixing of the MCMC for fitting the tree model. Using the four proposal mechanisms in single tree regression models, it has frequently been found that the sampler initially mixes well for the first few iterations as the tree grows to fit the data, but then becomes trapped in a local mode being unable to accept death moves with any reasonable probability. At the same time, change/swap moves tend to have very low acceptance rates, further limiting the mixing that can occur. As a result, while the in-sample prediction of these fits can be quite good, there is a danger of overfitting and the uncertainty intervals can be too small due to the MCMC sampler being stuck in a local mode.

One of the advantages of the model in equation (2), with its additive representation of simple trees, was to facilitate easier acceptance of birth/death proposals because most trees would be shallow and only represent a small portion of the overall response

signal. In addition, because of this shallow representation, removing the change/swap proposals was believed to be justified (e.g. Taddy et al. (2011); Pratola et al. (2014)). This seems reasonable for many regression problems investigated, but we have found that even this simple additive tree representation can suffer from poor mixing in certain problems. The applied problems where we have seen this occur come from computer experiments applications, where the measurement error variance σ^2 tends to be quite small and/or the dataset size is quite large.

To explore these issues, we next introduce some motivating examples that are problematic and exhibit poor mixing of the regression tree structure.

2.2 A Single-Tree Example

This example is taken from Wu et al. (2007), and it serves as a simple demonstration where proper mixing of the regression tree's topological structure is important. Their synthetic dataset had $d = 3$ covariates and the response y was calculated at $n = 300$ settings of these covariates as:

$$y(x) = \begin{cases} 1 + N(0, 0.25) & \text{if } x_1 \leq 0.5, x_2 \leq 0.5 \\ 3 + N(0, 0.25) & \text{if } x_1 \leq 0.5, x_2 > 0.5 \\ 5 + N(0, 0.25) & \text{if } x_1 > 0.5 \end{cases} \tag{3}$$

For this function, Wu et al. (2007) generated the covariates so the effects of x_1 and x_3 are confounded, as shown in the Supplementary Material. We fit this dataset using only $m = 1$ trees and found that the acceptance rate of tree moves (after the initial few steps of the sampler) was 0. In effect, the sampler collapsed on a single tree representation of the data and would not accept any birth/death proposal that might lead to an alternative representation of the data. The tree that was found is a 4-node representation that does not split on x_3. If one were to blindly believe this fit to the data, it would appear that x_3 has no effect on the response, whereas in fact we should conclude that either x_1 or x_3 (or both) may affect the response due to the confounding.

2.3 Computer Experiments Example

In computer experiments, a statistical emulator is used to model the outputs of simulators, $\eta(x)$, of complex physical processes (Sacks et al., 1989; Oakley and O'Hagan, 2002; Gramacy and Lee, 2008) as a function of the simulator inputs x. To simulate such an example, we treated the deterministic Friedman function (Friedman, 1991) as if it were our simulator, i.e., $y(x) = \eta(x) + \epsilon(x)$ where the Friedman function is $\eta(x) = 10sin(2\pi x_1 x_2) + 20(x_3 - .5)^2 + 10x_4 + 5x_5$. We sampled 5,000 observations from this simulator while adding i.i.d. normally distributed noise $\epsilon(x) \sim N(0, \sigma^2)$, and evaluated the use of BART with $m = 200$ trees as a flexible statistical emulator as in Pratola et al. (2014). We used the default BART priors as described in Chipman et al. (2010), in particular $\alpha = 0.95$ and $\beta = 2$ for the tree depth prior, and $\pi(\sigma^2) = \frac{\nu\lambda}{\chi^2_\nu}$ with default shape $\nu = 3$ and λ found by calibrating the 90th percentile of the prior on

σ is located at the sample standard deviation. The prior on the mean, $\pi(\mu_{jk})$ is i.i.d. $N(0, \tau^2)$ for all j, k where τ^2 is chosen so $\eta(x) = E[Y|x]$ is within (y_{min}, y_{max}) with 95% prior probability.

When the measurement error was large, e.g. $\sigma^2 = 1$, the MCMC algorithm (with birth/death proposals only) was found to mix reasonably well, having an acceptance rate around 18% and the 90% pointwise credible interval for $\eta(x)$ having an empirical coverage of 81%. However, as the error variance was decreased, this behavior changed drastically. The results of fitting BART when $\sigma^2 = 0.1$ are shown in the left pane of Figure 2. The MCMC was burned-in for 5,000 iterations and a further 5,000 iterations were drawn as samples from the posterior. In this case, the acceptance rate was very low at around 4%. In effect, the tree structure became stuck in a local mode with, for all practical considerations, zero chance of moving to a different area of tree-space that could give an equally good fit to the data.

The empirical coverage of the 90% pointwise credible interval for $\eta(x)$ shown in the figure is also very low at 53.8%. Since the tree structure of the model is not being explored by the sampler in this example, it suggests that the uncertainty coming from the terminal node μ_{jk}'s is only accounting for roughly half of the true uncertainty that should be explored by the MCMC sampler. This missing uncertainty is attributed to variability in partition rules of the interior tree nodes and other topological structural variability of the regression trees in this example.

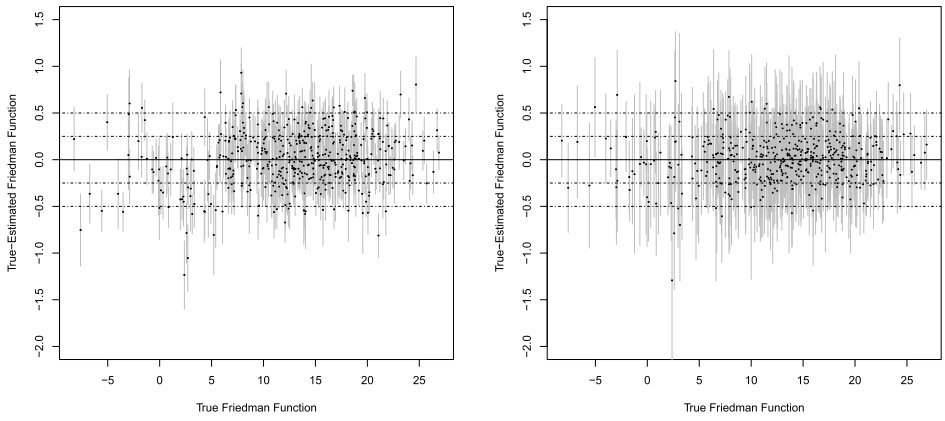

Figure 2: 90% credible intervals for posterior predictions of the Friedman function with $\sigma^2 = 0.1$ Left pane: The credible intervals using only birth/death proposals have an empirical coverage of 53.8%, under-represent the uncertainty, and the very low acceptance rate of birth/death proposals of 4% indicate poor mixing of the MCMC sampler. Right pane: The credible intervals using the proposed rotate and perturb proposal mechanisms have an empirical coverage of 90.6% and the acceptance rate is 25% indicating good mixing of the MCMC sampler.

2.4 Sampling $\pi(T|y, \sigma^2)$

The problems demonstrated in the above examples result from the difficulty in sampling $\pi(T|y, \sigma^2)$. As mentioned earlier, often one thinks of T as $T = \{(v_1, c_1), \ldots\}$ but this is a misleading simplification as underlying the variables and cutpoints is the tree topology, τ. The topology τ encodes the number of nodes in the tree, whether a node is internal or terminal, parent/child edge connections between all internal nodes, and node depths. Slow mixing results from the limited ability to sample this posterior topology, $\tau|y, \sigma^2, \{(v_i, c_i)\}$ using birth/death proposals, and limited or inefficient sampling of the variables and cutpoints, $(v_i, c_i)|y, \tau, \sigma^2, \{(v_{-i}, c_{-i})\}, \forall i$. This paper proposes a novel sampling algorithm allowing easier sampling of tree topology and efficient sampling of variables and cutpoints. The proposed algorithm for a single-tree consists of:

1. Draw $\tau|y, \sigma^2, \{(v_i, c_i)\}$ using birth/death or tree rotation proposals
2. Draw $(v_i, c_i)|y, \tau, \sigma^2, \{(v_{-i}, c_{-i})\}, \forall i$ using peturb or perturb within change-of-variable proposals
3. Draw $\mu_j|y, \tau, \sigma^2, \{(v_i, c_i)\}$ using conjugate Gibbs proposals
4. Draw $\sigma^2|y, \tau, \sigma^2, \{\mu_j\}, \{(v_i, c_i)\}$ using conjugate Gibbs proposals

In this algorithm, Step 3 and Step 4 are the usual Gibbs draws using conjugate priors, so the novelty of our proposed algorithm lies in the new tree rotation proposal of Step 1 and the perturb or peturb within change-of-variable proposals of Step 2. In the next section, we first introduce the tree rotation proposal that is used in Step 1 of the above algorithm, while in Section 4 we introduce the perturb and perturb within change-of-variable proposals that are used in Step 2.

3 Tree Rotation Proposal

A limitation of current Bayesian regression tree proposals is that they do not directly explore radically different tree arrangements nor do they change the dimension of the tree itself except for birth/death moves. In fact, of all the proposal mechanisms that have been developed in the literature, only the birth/death move changes dimensionality of the model. Because these moves only alter the bottom of the tree, it is unlikely for a regression tree MCMC algorithm to fully explore the space of nearly equivalent trees that have high posterior probability.

Here we develop by construction a more radical proposal, which can be thought of as a multivariate generalization of the simple univariate rotation mechanism found in the binary search tree literature (e.g. Sleator et al., 1988) and implemented in Gramacy and Lee (2008). This generalization allows dimension-changing proposals to occur at any interior node of a tree, and directly moves between modes of high likelihood. The basic idea of our rotate algorithm is demonstrated in Figure 3, which shows one possible trajectory of tree arrangements that can be constructed through rotation moves applied to node 2 (double circled). Note that Figure 3 indicates that if we start at the right tree, it is possible to rotate back to the original topology. The notion that a rotate traverses

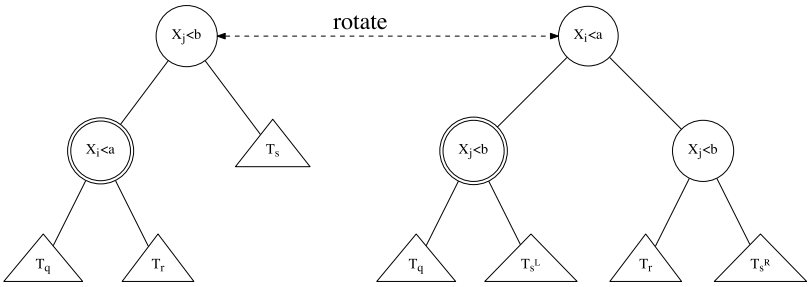

Figure 3: Two rotate moves applied sequentially at the same node (shown as double circled). The sub-trees T_q, T_r and T_s are arbitrary. Note that the rotate operation can be undone by a subsequent rotate.

between modes of high likelihood is motivated by viewing the effect of the first rotation performed in the example of Figure 3 in the plane of the X-space that is being affected. This is shown in Figure 14 of the Supplementary Material, which shows that the rotation has extended a rule further through the covariate space at this level of the tree. These features of the rotation operation allow us to satisfy the seemingly contradictory needs of a structural tree proposal that (i) is a local and computationally feasible operation, (ii) allows the sampler to move between very different tree arrangements of differing dimensionality, and yet (iii) moves directly between tree arrangements that have high likelihood.

3.1 The Rotation Operator, $\mathcal{R}$

More formally, for a rotatable node η_i (i.e. an interior node) who is the left child of its parent node $p(\eta_i)$, a right-rotation proposal $T' = \mathcal{R}[T]$ is constructed according to the pseudo-code shown in Listing 1 of the Supplementary Material (a similar algorithm for a left-rotation applies if η_i is the right child of $p(\eta_i)$). Note that a rotatable node is simply an internal node of the tree except for the root node, since left/right rotation at the root node is equivalent to rotation at its left/right children.

While the details of the rotation are a bit challenging due to the recursive formulation, we can summarize the details by viewing the rotation operator as the composition of simpler operations, that is, $\mathcal{R}[T] = \mathcal{R}^L_{merge}\mathcal{R}^R_{merge}\mathcal{R}^L_{cut}\mathcal{R}^R_{cut}\mathcal{R}^R_{rot}[T]$ where $\mathcal{R}^R_{rot}$ sets up the initial rearrangement of the tree structure for a right rotation, $\mathcal{R}^L_{cut}$, $\mathcal{R}^R_{cut}$ perform the cut operations and $\mathcal{R}^R_{merge}$, $\mathcal{R}^L_{merge}$ perform the merge operations. We describe each of these in further detail next when performing such a right rotation, and leave the analogous operations for performing left rotations to the reader.

$\mathcal{R}^R_{rot}$

First, the operation $\mathcal{R}^R_{rot} \equiv \mathcal{R}^R_{rot}(\eta_i; T)$ does the initial setup of the right-rotation at node η_i of tree T (a left rotation, $\mathcal{R}^L_{rot}$ would be similar). For instance, starting from the

top-left arrangement in Figure 3, this operation swaps the rules of η_i and its parent node $\mathrm{p}(\eta_i)$, and also introduces a new node for the right child of its parent node, $\mathrm{r}(\mathrm{p}(\eta_i))$, also using the same rule that was in $p(\eta_i)$. At the same time, the sub-tree starting at the right child $r(\eta_i)$ is moved to become the subtree at $l(r(p(\eta_i)))$. Subsequently, the sub-trees at the right child, $r(\eta_i)$, and the parents right-right child, $\mathrm{r}(\mathrm{r}(\mathrm{p}(\eta_i)))$, are initialized to duplicates of T_s, where T_s is the sub-tree starting at the parents right child $r(p(\eta_i))$ in the original tree T.

$\boldsymbol{\mathcal{R}^L_{cut}, \mathcal{R}^R_{cut}}$

Next, the left cut $\mathcal{R}^L_{cut} \equiv \mathcal{R}^L_{cut}(r(\eta_i), v_{p(\eta_i)}; T)$ of the subtree starting at right child node $r(\eta_i)$ along variable $v_{p(\eta_i)}$ at cutpoint $c_{p(\eta_i)}$ (the variable, cutpoint pair of η_i's parent) and the right cut $\mathcal{R}^R_{cut} \equiv \mathcal{R}^R_{cut}(r(r(p(\eta_i))), v_{p(\eta_i)}, c_{p(\eta_i)}; T)$ of the subtree starting at parents right-right child node $r(r(p(\eta_i)))$ along variable $v_{p(\eta_i)}$ at cutpoint $c_{p(\eta_i)}$ are performed. These occur since after the initial steps performed by $\mathcal{R}^R_{rot}$, both of the right subtrees $r(\eta_i)$ and $r(r(p(\eta_i)))$ of the tree are now under the constraint of the new rule appearing in $p(\eta_i)$, and they need to be made consistent with this rule. This is performed by the cutting operations $\mathcal{R}^L_{cut}, \mathcal{R}^R_{cut}$ which remove inadmissible sub-subtrees splitting on variable $v_{p(\eta_i)}$, leading to the modified T^L_s and T^R_s as shown in the top-right arrangement of Figure 3. Pseudo-code describing the left-wise cut operation is shown as Listing 2 in the Supplementary Material (a similar procedure performs right-wise cutting). In essence, the rotation operation leads to "dividing" the tree T_s along the parent node's cutpoint $c_{p(\eta_i)}$ of the parent node's variable $v_{p(\eta_i)}$. Accordingly, T^L_s is arrived at by removing all nodes from T_s that do not satisfy the rule $v_{p(\eta_i)} < c_{p(\eta_i)}$ while T^R_s is arrived at by removing all nodes from T_s that do not satisfy the rule $v_{p(\eta_i)} > c_{p(\eta_i)}$.

$\boldsymbol{\mathcal{R}^L_{merge}, \mathcal{R}^R_{merge}}$

Finally, the merge operation $\mathcal{R}^L_{merge} \equiv \mathcal{R}_{merge}(l(\eta_i), r(\eta_i), v_{\eta_i}, c_{\eta_i}; T)$ merges the sub-trees at $l(\eta_i)$ and $r(\eta_i)$ along variable v_{η_i} at cutpoint c_{η_i} and the merge operation $\mathcal{R}^R_{merge} \equiv \mathcal{R}_{merge}(l(r(p(\eta_i))), r(r(p(\eta_i))), v_{p(r(\eta_i))}, c_{p(r(\eta_i))}; T)$ merges the two subtrees $l(r(p(\eta_i)))$ and $r(r(p(\eta_i)))$ along variable v_{η_i} at cutpoint c_{η_i}.

The merge operation proceeds in a recursive fashion, always comparing the current nodes for the left tree, right tree and the merging variable and cutpoint (v_i, c_i) to the arrangements listed in Figure 4. For example, suppose we are attempting to merge T^L_s and T^R_s, and say we are at node 2 in both of these subtrees, which we will call η^L_2, η^R_2 respectively. If both nodes split on v_i (i.e. $v^L_2 = v^R_2 = v_i$) then clearly it must be the case that $c^L_2 < c_i$ and $c^R_2 > c_i$ and that in the original tree these two nodes were a parent-child pair or trivially had the v_i, c_i node above them (this corresponds to arrangement 7 in Figure 4). Ignoring the trivial case, the original tree could have had η^L_2 as the parent with η^R_2 as the right-child or η^R_2 as the parent with η^L_2 as the left-child, a situation which is described in the merge diagram of Figure 27 in the Supplementary Material. We work through a simple example shortly to help make this process more intuitive.

To see why such a merge operation is needed, consider that in a subsequent rotation the "dividing" of tree T_s can be possibly undone so that $Prob(\mathcal{RR}[T] = T) > 0$, as required for a valid MCMC algorithm. This is handled by the merge operations, which have the effect of going from the arrangement at the right of Figure 3 back to the original arrangement with some probability. The transition probability $P(T \to T')$ needs to take into account that the probability of moving from T to T' is affected by the number of ways that two subtrees can be merged into the new tree.

In performing the merge operation, one of the many possible reconstructions need be chosen randomly, and the total number of such random decisions is recorded so that the accept/reject calculation can be correctly calculated. In Listing 1, these counts are stored in the n_m^1, n_m^2 variables. These two count variables reflect the fact that the merging operation need be applied to $l(\eta_i) \cup r(\eta_i)$ merged along v_{η_i}, c_{η_i} and to $l(r(p(\eta_i))) \cup r(r(p(\eta_i)))$ merged along $v_{p(r(\eta_i))}, c_{p(r(\eta_i))}$, where here the symbol '$\cup$' represents merging. That is, n_m^1 and n_m^2 represent the total number of possible mergings that can occur in forming $l(\eta_i) \cup r(\eta_i)$ and $l(r(p(\eta_i))) \cup r(r(p(\eta_i)))$. Of the n_m^1 possible mergings of $l(\eta_i) \cup r(\eta_i)$ only one will be randomly selected, and similarly only one of the n_m^2 possibilities will be selected as the merging of $l(r(p(\eta_i))) \cup r(r(p(\eta_i)))$. This merging process should become more clear in Section 3.3 where we work through a simple example.

Similarly, to calculate the return transition probability $P(T' \to T)$ we need to take into account that the probability of returning to the current configuration from the rotated configuration T' is again affected by the number of ways that two sub-trees in the rotated tree can be merged in the inverse step to return to the current tree. In Listing 1, these counts are stored in the n_s^1, n_s^2 variables.

In calculating the number of merges possible in both the forward proposal and in calculating the probability of inverting the rotation, one need recognize that a non-trivial merge of both $l(\eta_i) \cup r(\eta_i)$ and $l(r(p(\eta_i))) \cup r(r(p(\eta_i)))$ leads to an inadmissible state. A non-trivial merge is one that does not retain the original variable and cutpoint node (e.g. v_{η_i}, c_{η_i} or $v_{p(r(\eta_i))}, c_{p(r(\eta_i))}$). If both merges are non-trivial, then the (variable, cutpoint) pair required for inverting the rotation has been removed from the tree which would not allow inversion to take place, so such proposals are rejected.

3.2 Calculating the M-H Ratio

When calculating the log-integrated likelihoods for the accept/reject step, one need only consider terminal nodes that are part of the sub-tree of $p(\eta_i)$ since the rest of the tree T remains unchanged by the rotation proposal. Hence, the computational cost of a rotation step, while greater than a simple birth/death proposal, is much reduced compared to a more drastic restructure move such as the proposal of Wu et al. (2007). While the operation of cutting unreachable sub-branches is entirely deterministic, there are usually many possible merges and the stochasticity of the rotation proposal comes from selecting one of these arrangements at random. The number of possible merges at any given level of the left and right trees is summarized in Figure 4. This figure is interpreted as follows. For each possible merging of two nodes, there are six topological properties to be considered, itemized by the last six columns. For instance, in merge

Arrangement	Merge Type	l->v=r->v	l->c=r->c	l->v=vi	r->v=vi	l is leaf	r is leaf
1	1 (i,ii)			x			x
2	2 (i,ii)				x	x	
3	3 (i,ii)					x	x
4	4 (i,ii)	x	x				
5	5 (i,ii)			x			
6	6 (i,ii)				x		
7	7 (i,ii,iii)	x		x	x		
otherwise	8						

Figure 4: Listing of possible merging arrangements and corresponding index of unique merge types for an arbitrary left (l) and right (r) tree. The label $l \to v$ ($r \to v$) corresponds to the left (right) tree's variables, and $l \to c$ ($r \to c$) corresponds to the left (right) trees cutpoints. The left and right trees are being merged under the split rule (v_i, c_i). The final two columns indicate whether the left or right tree is actually a terminal leaf node. Note that when a node is a leaf, it cannot have a v or c rule which corresponds to the hashed-out boxes. A diagram depicting merge type 7 is shown in Figure 27 while diagrams for the remaining merge types are found in the Supplementary Material (Figures 20–26).

type 4 we have $l \to v = r \to v$ as noted by the 'x' under that column, as well as $l \to c = r \to c$. This means that the node from the left tree and right tree both split on the same variable and cutpoint. This situation is addressed by the corresponding figure for merge type 4 which describes how the merge may be performed. Figures depicting how the merges are performed can be found in the Supplementary Material (Figures 20–26).

The dark grey cells of Figure 4 denotes columns which are not applicable. For example, in merge type 1 we have that r is a leaf and therefore has no variable or cutpoint. As such, three columns for merge type 1 are greyed out since they are not applicable. There are, in total, 7 particular merge types to consider plus an additional "otherwise" scenario shown as merge type 8.

The actual number of merge reconstructions can, of course, be higher due to the recursive definition of the merging operation. For example, merge type 7 of Figure 4 identifies three possibilities denoted (i), (ii) and (iii), which are shown in Figure 27. This latter diagram demonstrates the recursive nature of the merging operation, and this recursion is followed down the levels of the right and left trees when counting the number of possible merges in the proposal distribution. We explicitly recursively calculate these counts when constructing a rotation proposal.

A final item to note in calculating the acceptance ratio is determining the probability of rotating at node η_i in both the forward proposal and the inverse step. For the forward proposal, this is just the reciprocal of the number of internal nodes (less the root node) in the tree. However, for the inverse step it may be possible to rotate back to the current configuration from both η_i or the node "opposite" of η_i (which is $r(p(\eta_i))$ in a right-rotation). In this case, the probability is 2 over the number of internal nodes in the rotated configuration.

Taking all these considerations into account, the acceptance ratio is calculated as

$$min\left(1, \frac{\pi(T')p_r(T')p_s^1 p_s^2 L(T')}{\pi(T)p_r(T)p_m^1 p_m^2 L(T)}\right) \tag{4}$$

where $\pi(T), \pi(T')$ are the prior probabilities of the trees, $L(\cdot)$ represent the integrated likelihoods for the trees, $L(T) = \prod_{j=1}^{|M|} \int \mathcal{L}_j(y)\pi(\mu)d\mu$ where $\mathcal{L}_j$ is the likelihood function for the jth terminal node and $\pi(\mu)$ is a conjugate i.i.d. normal prior for the terminal nodes under a constant terminal-node mean function model, $p_r(T) = \frac{1}{|T|-1}$ is the probability of rotating at a particular internal node of T,

$$p_r(T') = \begin{cases} \frac{1}{|T'|-1} & \text{if only 1 way to invert} \\ \frac{2}{|T'|-1} & \text{if two ways to invert,} \end{cases}$$

and p_m^1, p_m^2, p_s^1 and p_s^2 are calculated as $p_m^1 = \frac{1}{n_m^1}$, $p_m^2 = \frac{1}{n_m^2}$, $p_s^1 = \frac{1}{n_s^1}$, $p_s^2 = \frac{1}{n_s^2}$, where the values of n_m^1, n_m^2, n_s^1 and n_s^2 are calculated through the recursive application of the merging process, using the merge types outlined in Figure 4.

For calculating $\pi(T) = \pi(\tau)\prod_{i=1}^{|T|}\pi(c_i|\tau, v_i, \{(v_{-i}, c_{-i})\}) \times \pi(v_i|\tau, c_i, \{(v_{-i}, c_{-i})) = \prod_{j=1}^{|M|}(1-\pi(\eta_j))\prod_{i=1}^{|T|}\pi(\eta_i) \times \pi(c_i|\tau, v_i, \{(v_{-i}, c_{-i})\}) \times \pi(v_i|\tau, c_i, \{(v_{-i}, c_{-i})\})$ we use the depth penalizing prior of Chipman et al. (2010), $\pi(\eta_i) = \gamma(1+d_i)^{-\beta}$ where d_i is the depth of node η_i, and $\pi(v_i|\cdot)$ represents the probability of splitting on variable v_i at node η_i which is $\frac{1}{\#\text{ available variables at }\eta_i}$ and $\pi(c_i|\cdot)$ represents the probability of splitting at node η_i on variable v_i at cutpoint c_i which is $\frac{1}{\#\text{ available cutpoints at }\eta_i}$, which can be determined based on Equation (5) derived in Section 4.

As with other tree proposal mechanisms, the acceptance probability (4) is modified by the constraint requiring all terminal nodes to contain data, leading to automatic rejection if this constraint is not met.

3.3 A Simple Rotation Example

A more concrete demonstration of the rotation proposal is shown in Figures 5–8 as well as in Figures 15–17 in the Supplementary Material. Here, the basic structure of the starting tree T in Figure 5 is similar to the left tree in Figure 3. In the first rotation we propose transitioning from $T \to T'$ by performing a rotation at node η_2 which is highlighted with a double circle. The rotation occurs at node η_2 with probability $p_r(T) = \frac{1}{|T_r|+|T_q|+5}$. The rotated tree will be arrived at by sequentially performing the operations $\mathcal{R}_{rot}^R, \mathcal{R}_{cut}^L, \mathcal{R}_{cut}^R, \mathcal{R}_{merge}^L$ and $\mathcal{R}_{merge}^R$. Applying $\mathcal{R}_{rot}^R$, we arrive at the tree arrangement of Figure 6. We see that the rules of the rotate node and its parent have been swapped, an extra node with rule "$X_2 < 0.5$" has been created on the right side of the tree and two copies of the subtree T_s are now present. The parent of the subtree T_r has also changed according to this operation.

Next, we need to perform the $\mathcal{R}_{cut}^L$ and $\mathcal{R}_{cut}^R$ steps by cutting the shaded subtree T_s along the rule $X_1 < 0.6$. This results in the shaded subtrees T_s^L and T_s^R shown in Figure 7. After performing the cut steps, we finally perform the merge steps $\mathcal{R}_{merge}^L, \mathcal{R}_{merge}^R$.

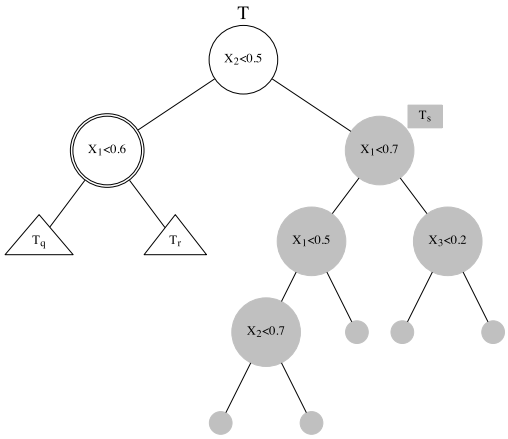

Figure 5: An example tree, T, where we will apply the rotate operator at node 2, denoted by double circles. Terminal nodes are denoted by the small empty circles. The sub-trees T_q and T_r are arbitrary. The sub-tree denoted T_s (shown in grey) starting with the $X_1 < 0.7$ rule is the tree that will be split during rotate.

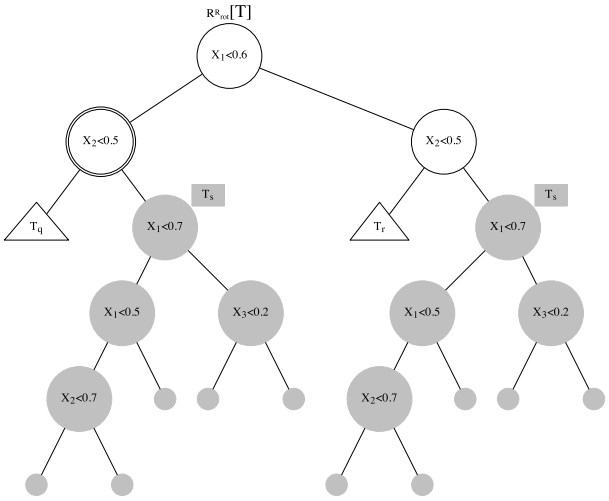

Figure 6: The state of tree T after the first step in the rotate operator, $\mathcal{R}^R_{rot}$ has been applied. Note that T_s now appears on both sides of the tree.

Here, the two merges that need occur in the rotation are the merge of $T_q \cup T_s^L$ along the rule $X_2 < 0.5$ and $T_r \cup T_s^R$ also along the rule $X_2 < 0.5$ as shown in Figure 8. For simplicity of exposition, let us assume that the merges $T_q \cup T_s^L$ and $T_r \cup T_s^R$ can only be performed in one way, retaining $X_2 < 0.5$ as shown in Figure 15. As such, $p_m^1 = p_m^2 = 1.0$.

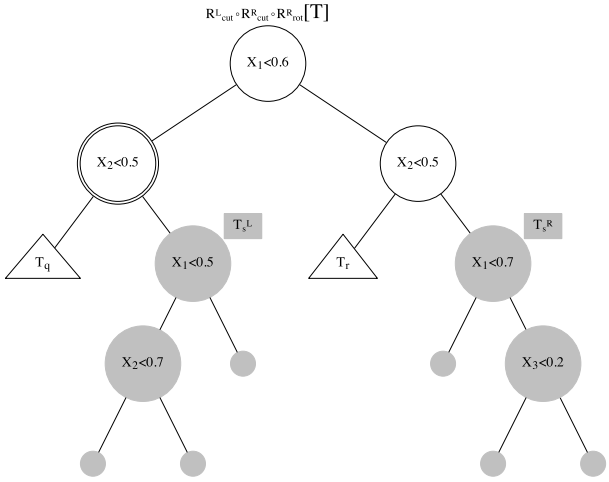

Figure 7: The state of tree T after the second step in the rotate operator, $\mathcal{R}^L_{cut}$ and $\mathcal{R}^R_{cut}$, have been applied. Note that both copies of the sub-tree T_s have been cut along the $X_1 < 0.6$ rule, resulting in the respective sub-trees T_s^L and T_s^R.

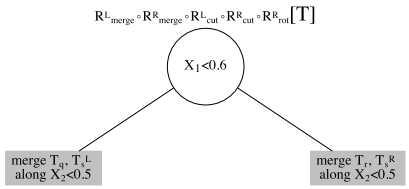

Figure 8: In the third stage of the rotate operator, the merge operation is applied to the left and right sub-trees of the node $X_1 < 0.6$. The merges are performed along the rule $X_2 < 0.5$.

In order to calculate the acceptance probability of the proposal $T' = \mathcal{R}[T]$ we also need to calculate the probability of transitioning from T' back to T. We can investigate this by the behavior of rotating T' at node η_2 or η_3 in Figure 15. Because the rotation can occur at either of these nodes, we have $p_r(T') = \frac{2}{|T_r|+|T_q|+6}$. The reconstruction will be the same in either case, so proceeding as if we rotate at node η_2 in T' (shown as double circled in Figure 15) we can apply $\mathcal{R}^R_{rot}, \mathcal{R}^L_{cut}$ and $\mathcal{R}^R_{cut}$ (not shown) thereby arriving at the arrangement of Figure 16 (found in the Supplementary Material) where the merge $T_q \cup T_r$ along $X_1 < 0.6$ need be performed for the left subtree and the merge $T_s^L \cup T_s^R$ along $X_1 < 0.6$ need be performed for the right subtree. For simplicity we again will assume that the merge $T_q \cup T_r$ can only be performed in one way, arriving at the arrangement in Figure 17 (found in the Supplementary Material), which is the same as the left subtree in the original Figure 5, and we have $p_s^1 = 1.0$.

For merging $T_s^L \cup T_s^R$ along $X_1 < 0.6$ there are 7 possible merges as shown in Figure 18 of the Supplementary Material, one of which is the trivial merge 7(ii) and one of which

forms the original arrangement in the right hand side of T, labeled as 7(iii),1(i),3(i). These labels identify the sequence of merge types outlined in Figure 4 that were applied to form each of the 7 possibilities. For the case of 7(iii),1(i),3(i), this sequence means that merge type 7 case (iii) (Figure 27 in the Supplementary Material) was applied to T_s^L, T_s^R, then merge type 1 case (i) (Figure 21 in the Supplementary Material), and finally merge type 3 case (i) (Figure 23 in the Supplementary Material). Since there are 7 such possible sequences, we have $p_s^2 = \frac{1}{7}$. Of these 7 possibilities, if we randomly select the merge sequence 7(iii),1(i),3(i), then $T'' = \mathcal{R}[T']$ shown in Figure 17 corresponds to the original arrangement of T in Figure 5, i.e. $T'' = T$, demonstrating reversibility.

Finally, combining these calculations with the integrated likelihoods $L(T), L(T')$ and the prior probabilities of the tree states $\pi(T), \pi(T')$ one can calculate the acceptance probability for T' using Equation (4) and perform the MH step.

4 Perturbation Proposal

Besides birth/death moves, a popular existing MH proposal for exploring the posterior of Bayesian regression tree models is the change proposal. This move can be thought of as changing a cutpoint, a variable or both simultaneously. We prefer to take a simple one-at-a-time approach to our sampling algorithm, and so will first focus on proposing a new cutpoint at internal node i, denoted c_i, given the cutting variable v_i and the rest of the tree structure T.

Interestingly, while the "change" proposal has been around for some time, there are some varied explanations of its implementation in the literature. Denison et al. (1998) draw from a uniform proposal distribution on the range of values v_i takes. This amounts to an independence sampler when the prior is uniform with endpoints taken as the min and max observed covariate values in the dataset. Chipman et al. (2002) take a similar approach, drawing proposals from the prior distribution. In Chipman et al. (1998), the proposal distribution is restricted to functions that depend only on the part of T ancestral to node i. Their default is again to take the proposal to be uniform, but this time restricted by the requirement that values drawn from this proposal cannot lead to empty terminal nodes. This requires propagating the data through the proposed tree to ensure this condition is met, which is an expensive operation. Wu et al. (2007) and Chipman et al. (2010) use the above approach in their change proposal while Gramacy and Lee (2008) use a random walk approach, essentially incrementing or decrementing the cutpoint by the single smallest increment available from the covariate matrix X.

It is useful to think about the meaning of the cutpoint and split variable components of regression tree models. Namely, the cutpoints of a regression tree relate to the "wiggliness" of the response being modeled while the split variables are indicative of the importance of that variable in modeling, or explaining the variability, of the response. For a given collection of split variables and cutpoints $\{v_i, c_i\}$ that form a fitted regression tree model, this collection represents a discrete encoding of the model. Changing just a single v_i or a single c_i represents an entirely different model. However, intuitively, changes involving c_i are in some sense more local than changes involving v_i.

A reasonable proposal for a particular c_i conditional on everything else is an interval $(a_i^{v_i}, b_i^{v_i})$ that should take into account the full tree structure $T \setminus c_i$. For node i, let $C_{p(i)}^{v_i}$ be the collection of cutpoints for all nodes ancestral of node i that split on variable v_i, and let $C_{l(i)}^{v_i}$ (similarly $C_{r(i)}^{v_i}$) be the collection of cutpoints for all nodes in the left subtree of node i (similarly right subtree) that split on variable v_i. The ancestral nodes can be factored into ancestral nodes which are "left ancestors" of node i and those which are "right ancestors" of node i, i.e. $C_{p(i)}^{v_i} = \{C_{p_l(i)}^{v_i}, C_{p_r(i)}^{v_i}\}$. A left ancestor is an ancestor of node i such that node i is on the ancestor node's right subbranch, and similarly a right ancestor is an ancestor such that node i is on the ancestor node's left subbranch.

A uniform proposal that is consistent with the full tree structure draws a cutpoint uniformly from the interval $(a_i^{v_i}, b_i^{v_i})$ where

$$a_i^{v_i} = max\left(0, max(C_{p_l(i)}^{v_i}), max(C_{l(i)}^{v_i})\right);\ b_i^{v_i} = min\left(1, min(C_{p_r(i)}^{v_i}), min(C_{r(i)}^{v_i})\right). \quad (5)$$

Note that the proposal does not directly depend on the data and so is easily computed.

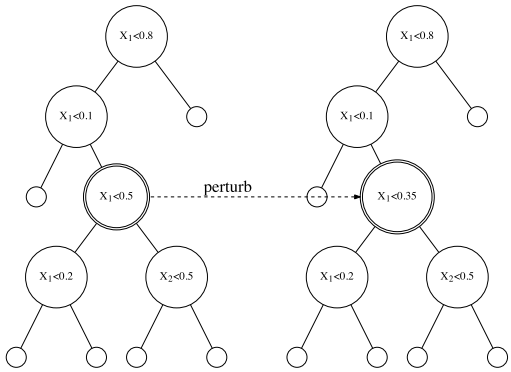

Figure 9: An example of a perturb proposal at node 5. In this case, we have $C_{p(5)}^{v_1} = \{0.1, 0.8\}$, $C_{p_l(5)}^{v_1} = \{0.1\}$, $C_{p_r(5)}^{v_1} = \{0.8\}$, $C_{l(5)}^{v_1} = \{0.2\}$ and $C_{r(5)}^{v_1} = \{\}$, giving the open interval $(a_5^{v_1}, b_5^{v_1}) = (0.2, 0.8)$ from which to draw proposals for a new cutpoint value.

A simple example of the perturb proposal is shown in Figure 9, where we try to perturb the cutpoint at node 5 which has initial rule "$x_1 < 0.5$". If we generate proposals from the prior on the interval $(0, 1)$, immediately 40% of these proposals are outside the valid range $(a_5^{v_1}, b_5^{v_1}) = (0.2, 0.8)$ and will be rejected. These rejections occur solely because some terminal nodes will be unreachable and will therefore contain no data, which automatically leads to rejection by the MH sampler. For example, proposing the cutpoint to be $c_5 = 0.05$ would be rejected because no data would be mapped to the right sub-tree of node "$x_1 < 0.2$" due to the constraint "$x_1 < 0.1$" at node 2. Or, proposing the cutpoint to be $c_5 = 0.9$ would be rejected because no data would map to the right sub-tree of node 5 due to the constraint "$x_1 < 0.8$" at node 1. Similarly, generating proposals from the prior conditioned on the structure of the tree above node 5 (i.e.

from $(0.1, 0.8)$) leads to about 17% of such proposals being outside the valid range and therefore guaranteed to be rejected. For instance, proposing $c_5 = 0.15$ would be rejected because no data would map to the right sub-tree of the node "$x_1 < 0.2$". In contrast, the perturb proposal generates cutpoints from the valid range $(a_5^{v_1}, b_5^{v_1}) = (0.2, 0.8)$, which accounts for constraints from ancestral (i.e. $C_{p(5)}^{v_1}$) and descendant (i.e. $C_{l(5)}^{v_1}, C_{r(5)}^{v_1}$) parts of the tree about node 5 to avoid such spurious rejections.

Greater flexibility is found by introducing a constant $\alpha \in (0, 1]$ (i.e. independent of node), or $\alpha_v \in (0, 1])$ (i.e. a unique scaling for each of the covariate variables), that scales the proposal to be more local, and α can be updated using adaptive MCMC to target higher acceptance rates. This approach chooses cutpoints in the scaled interval,

$$sa_i^{v_i} = max\left(c_i - \alpha\left(\frac{b_i^{v_i} - a_i^{v_i}}{2}\right), a_i^{v_i}\right);\ sb_i^{v_i} = min\left(c_i + \alpha\left(\frac{b_i^{v_i} - a_i^{v_i}}{2}\right), b_i^{v_i}\right), \quad (6)$$

where the $a_i^{v_i}, b_i^{v_i}$ are computed as in equation (5), thereby making the proposals in a more local neighborhood about the current cutpoint c_i within the valid interval $(a_i^{v_i}, b_i^{v_i})$. If we instead fix α to be very small, forcing minute local moves in cutpoint values, then the proposals behavior would be similar to that of Gramacy and Lee (2008) although one would expect mixing to be slow. In practice, we have fixed α to somewhere between 1%–10%, which has worked well in the problems we have tried. Alternatively, one might instead tune it automatically using adaptive MCMC if desired. For example, an interval which is too wide may lead to a proposed cutpoint that results in some terminal nodes to contain no data, and hence be rejected. Tuning the scaled perturb in Equation (6) allows one to minimize such poor proposals. Alternatively, the current practice of requiring terminal nodes to contain data can be relaxed – setting this to zero has no ill effect on either the samplers proposed in this paper or the resulting trees.

4.1 Perturb Within Change-of-Variable

While the perturbation proposal has the clear interpretation of exploring the "wiggliness", or spatial variability of the response being modeled, it is less intuitive what variability the usual change of variable proposal explores.

Changing a variable at some internal node within a regression tree implies that for all observations mapping to that internal node, the variability of the response is well represented by the tree structure below this internal node no matter if the node splits on the current variable or the variable we propose to change to. This could be reasonable if (i) the partition of observations when using the new variable is unique from the partition of observations using the current variable, but nonetheless yields two partitions that are adequately explained by the tree structure below the current node; or,(ii) the partition of observations is nearly the same using either variable, which occurs when the two covariates under consideration are dependent on one another, such as the highly correlated covariates found in the synthetic example of Section 2.3. In our opinion, the latter case seems more likely and so we develop a change-of-variable proposal that explores the variability in the posterior that results from such correlated covariates. The

approach taken is essentially to precondition the change-of-variable proposal by taking into account the empirical correlation structure of the covariate matrix.

Using the pairwise correlation between current variable v_k and the other variables $v_{\setminus k}$, a change of variable at node η_i is proposed in the following way. First, the probability of changing v_k to $v_j, j \neq k$ should be relatively higher if the current variable is highly correlated with another (or many other) variables in the dataset. On the other hand, this probability should be very small if the current variable is relatively independent (uncorrelated) with all other variables. This is accomplished by proposing a transition from v_k to v_j with probability proportional to

$$\frac{|Cor(X_k, X_j)| \times \mathcal{I}_{\left(a_i^{v_j}, b_i^{v_j}\right)\neq\{\}}}{\sum_l |Cor(X_k, X_l)| \times \mathcal{I}_{\left(a_i^{v_l}, b_i^{v_l}\right)\neq\{\}}} \tag{7}$$

where $Cor(\cdot,\cdot)$ represents a measure of relatedness between variables X_k and X_j. A simple approach would use the sample Pearson correlation between variables, however this limits the proposal to assuming a linear relationship between variables. Instead, we calculate the Spearman rank correlation between variables, which allows for non-linearly related variables to be highly correlated. Note that if the variables are negatively correlated, we proceed as if the variables were positively correlated but flip the left and right subbranches of the node where change-of-variable is being proposed.

The indicator functions in equation (7) ensure that we only give positive probability to transition to variables which have cutpoints available at the node in question. That is, $\mathcal{I}_{\left(a_i^{v_l}, b_i^{v_l}\right)\neq\{\}} = \begin{cases} 1 & \text{if } \exists \text{ cutpoints at } \eta_i \text{ for } v_l \\ 0 & \text{otherwise} \end{cases}$. Note that if v_k is independent of all other variables, then with high probability this formula will propose staying at variable v_k. If v_k is highly correlated with a single other variable v_j, then this formula will lead to proposals that stay at v_k about 50% of the time and propose transitions to v_j about 50% of the time, and so on.

This procedure could be made more flexible by only calculating the correlations using the x's that map to node i. This would then be taking into account more local information which may be useful when the relationship between covariates is more complex, such as when modeling waterways or other geographically constrained responses (e.g. Rathbun, 1998; Løland and Høst, 2003; Pratola et al., 2015). On the other hand, for nodes near the bottom of the tree, using only x's mapping to a near-bottom node i may lead to having a very small number of x's mapping to node i and if the covariate space is relatively large then the procedure is in some sense underdetermined and sensitive to the small number of observations mapping to i. Hence, in practice we have used the full sample correlations in forming our change-of-variable proposal procedure.

A final matter to note is the implementation of this preconditioner when the covariate dimension is large. In such instances, many spurious small correlations between variables may appear. We suggest treating such situations by using an empirical cutoff (e.g. all sample correlations ≤ 0.30 are replaced with 0), although other approaches are also feasible.

5 Examples

We now return to the examples introduced in Section 2 to investigate the performance of our proposed sampling strategies. We start with the single tree example and then explore the computer experiments example. An additional dataset from a study of heart rate conducted at The Ohio State University is discussed in the Supplementary Material. Unless otherwise indicated, we used a MH algorithm with birth/death, rotation and perturbation proposals. At each iteration of the MCMC, a rotation proposal at a randomly chosen internal node was selected 30% of the time while a birth or death proposal at a random terminal or next-to-terminal node occurred otherwise. Adapting this percentage is possible but we have found it reasonably easy to set the proportion manually and leave it fixed. Adapting α in the perturb proposal of Equation (6) was done by initializing α to 0.1 and allowing it to adapt every 1,000 iterations for 10,000 iterations.

5.1 Single-Tree Example

In this example, we use the change-of-variable and rotation proposals to allow the sampler to find 5-node (3 terminal nodes) and 7-node (4 terminal nodes) tree structures that are consistent with the data. Note that using birth/death proposals in Section 2.2 only a single 7-node representation was found, simply as a result of the starting random seed (we might have just as easily found a 5-node representation). If we were to augment the birth/death proposal with change, swap or restructure (Wu et al., 2007) proposals, which do not change the tree dimensionality, the diversity of tree structures found would certainly improve, but it is unlikely that the more parsimonious 5-node structure would be sampled if starting from the same random seed. This suggests the rotation proposal greatly increases the diversity of tree structures that can be explored by the MCMC sampler, potentially eliminating the need for random restart or multiple chain approaches to fully explore the posterior. The diversity of tree structures found for the

Sampler	Birth/Death	ChV	ChV and Rotate	Rotate
# trees of size 5	0	2	2	1
# trees of size 7	1	2	6	2
# trees of size other	0	0	0	0

Table 1: Number of unique regression trees found for the synthetic example using the tree rotation and change-of-variable (ChV) proposal mechanisms with $m = 1$ regression trees compared to the default birth/death only sampler, and ChV or rotate separately. Without using the rotation and/or change-of-variable proposals, only a single tree with 7 nodes (4 terminal nodes) was found in the posterior samples.

different samplers are summarized in Table 1. As noted, the birth/death sampler found a single tree, while adding the rotate proposal finds 3 trees and adding the change-of-variable proposal finds 4 trees. However, the greatest diversity is found with all moves used in the sampler, resulting in 8 unique tree topologies in the posterior samples.

Since x_1 and x_3 are highly correlated with each other (and independent of x_2), the preconditioned change-of-variable proposal is important in this example. Using this proposal mechanism allows the sampler to propose transitions from x_1 to x_3 (or vice-versa) about 50% of the time and maintains a high acceptance rate since proposals to transition to x_2 are rarely, if ever, made (and would always be rejected if proposed). The rotate proposal is also important since the root node of a tree fitting the data can start with either variable 2 or one of variable 1 or 3. Without the rotate proposal, it would be extremely unlikely that the MCMC would prune the tree back to a single node to allow for a potential transition to a different root node variable. One situation where poor

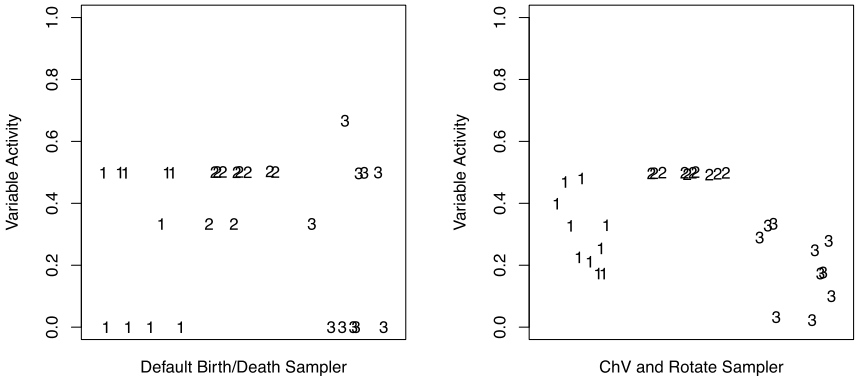

Figure 10: Variable activity measured as the proportion of internal tree nodes splitting on variables for the birth/death sampler (left pane) and the birth/death sampler augmented with change-of-variable (ChV) and rotate proposals (right pane). Each plotted number represents the estimated variable activity for a single replicate of the 10-fold cross-validation study. Note that in 9 of 10 cross-validation replicates, the birth/death sampler incorrectly concludes that the proportion of splits on either variable 1 or variable 3 is exactly 0.

sampling of the tree topologies may cause problems is in variable selection. Bleich et al. (2014) investigate the important inferential task of variable selection using BART based on measuring variable activity as the proportion of posterior tree internal nodes that split on each variable as proposed in Chipman et al. (2010). Yet, for variables that are related, poor sampling may lead to incorrect conclusions regarding variable selection. As shown in Table 1, in this example the default birth/death sampler does not explore posterior trees very well which may negatively affect variable selection. To explore this in detail, a 10-fold cross-validation study was performed, and in each fold we estimate the activity of variable i as the proportion of internal nodes splitting on i over all the posterior samples. We performed this study for the default birth/death sampler and the birth/death sampler with change-of-variable and rotate.

The results of this study are summarized in Figure 10. The left pane shows the variable activity for all 3 variables using the birth/death sampler and the right pane shows the variable activity using the full sampler. There are some stark differences between

the samplers. For instance, the poor sampling of posterior trees using only birth/death proposals results in a discretized spread of variable importance measures across the 10 folds, while the full sampler displays a more continuous spread of importance measures.

The most impactful difference is the interpretation of the correlated variables 1 and 3. Since we know that variable 1 is used in defining the true underlying function of Equation (3) and we know variables 1 and 3 are hopelessly confounded, given only observations of the data an analysis of variable activity should conclude that both variable 1 and 3 could be important. However, in Figure 10, using the birth/death sampler results in an importance of exactly 0 for variable 1 in four of the 10 cross-validation folds, while an importance of exactly 0 for variable 3 is determined in 5 of the 10 cross-validation folds. In other words, in 9 out of the 10 cross-validation folds, the analysis incorrectly concludes that either variable 1 or 3 is not an important variable. In contrast, using the full sampler, no variable is assigned an importance of exactly 0, and only in 2 of the 10 folds does the estimate of importance for variable 3 fall below 10%. Thus, the ability to more fully explore the posterior distribution using the samplers proposed in this paper can result in a vastly improved ability to infer variable activity.

5.2 Computer Experiments Example

In Section 2.3, it was shown that when modeling the Friedman function with BART using only birth/death proposals, the 90% credible interval had an empirical coverage of 53.8% indicating that the posterior was underrepresenting model uncertainty. It seemed likely that this behavior could be attributed to the poor mixing of the MCMC sampler, which had an acceptance rate for birth/death proposals of only 4%.

Applying the rotation and perturbation proposals to the same dataset showed a very noticeable improvement in the behavior of the MCMC sampler, shown in the right pane of Figure 2. In this case, we selected a rotation step for 30% of the MH proposals, and the usual birth/death step for the remaining 70% of MH proposals. For the perturbation proposal, α was initialized to 0.10 and then adapted every 1,000 iterations for 10,000 iterations. After adaptation, the sampler was allowed to burn-in for 5,000 iterations and then a final 5,000 iterations were saved as draws from the posterior. The resulting behavior was a very good acceptance rate of around 25%, and the empirical coverage of the 90% credible interval was 90.6%, indicating good coverage with no under-representation of model uncertainty. The effect of the percentage of rotate proposals on the convergence of the MCMC was investigated empirically by exploring the relationship between a range of rotate proposal percentages on the convergence of the traceplot for the error, σ. This was evaluated using the so-called fat-marker test (visual inspection of the traceplots) as well as effective sample size and the Geweke (1992) diagnostic. This is summarized in Table 2. It is difficult to asses convergence of a high and varying dimensional non-parametric model such as regression trees, but in our experience the traceplot of σ can be useful for such purposes. Table 2 suggests a useful range for the percentage of rotate proposals is 20–40% in order to ensure convergence of the MCMC. Considering computational cost suggests that setting the percentage of rotate proposals to 30% as we have done is reasonable.

Rotate	0%	10%	20%	30%
Avg. Tree Depth	2.3	1.8	1.5	1.3
Geweke Diagnostic	1.07	0.25	7.58	-0.23
Effective Sample Size	179	272	114	314
Effective Sample Size per Second	0.366	0.584	0.252	0.700
σ Converged?	N	N	Y	Y
Rotate	40%	50%	70%	90%
Avg. Tree Depth	1.1	0.9	0.5	0.16
Geweke Diagnostic	-2.05	4.58	16.98	0.53
Effective Sample Size	585	77	47	64
Effective Sample Size per Second	1.26	0.162	0.101	0.136
σ Converged?	Y	N	N	N

Rotate	Birth/Death Only
Avg. Tree Depth	2.4
Geweke Diagnostic	6.52
Effective Sample Size	204
Effective Sample Size per Second	0.356
σ Converged?	N

Table 2: The effect of varying the percentage of rotate proposals on the convergence of the traceplot of σ and the average depth of trees in the posterior for the example of Section 2.3.

The good acceptance rate in this example indicates that the sampler is able to easily explore birth/death proposals that have a good probability of acceptance, even as the rotation step explores different modes of the posterior by altering the internal structure of the regression trees. We also note in Table 2 that the average tree depth with only birth/death proposals was around 2.3–2.4 while the average tree depth using our proposals was around 1.1–1.5. This further suggests that the rotation proposal is able to help the MCMC sampler explore the model space to find a more parsimonious fit to the data along with superior predictive performance and coverage of credible intervals.

To more fully explore these behaviors, we ran a 10-fold cross-validation experiment for this data using the basic birth/death sampler, a perturb-only sampler, a rotate-only sampler, and both rotate and perturb. For each of these, we recorded the in-sample and out-sample coverage of the 90% pointwise credible interval for $\eta(x)$, the mean squared prediction error and run-times to evaluate computational tradeoffs. The experiment was performed for a typical error variance of $\sigma^2 = 1.0$ and a smaller variance of $\sigma^2 = 0.01$.

The overall pattern of empirical coverage for $\sigma^2 = 1.0$ and $\sigma^2 = 0.01$ is that the birth/death sampler underrepresents the uncertainty in the predicted function $\eta(x)$, and this problem becomes worse as σ^2 decreases and is worse for out of sample predictions compared to in-sample predictions. For instance, with $\sigma^2 = 1.0$ the average out of sample coverage of the 90% credible interval for $\eta(x)$ without the proposed samplers is 75%, and this decreases to 54% when $\sigma^2 = 0.01$. Having either perturb or both perturb and rotate gives nominal intervals when $\sigma^2 = 1.0$ and close to nominal for $\sigma^2 = 0.01$ (perturb

alone tends to be slightly conservative but adding in rotate arrives at nearly nominal coverage). The average out of sample prediction error for the birth/death sampler when $\sigma^2 = 1.0$ is 0.201 while using both perturb and rotate gives 0.160, a 20% reduction. When the variance is reduced to $\sigma^2 = 0.01$, the average prediction error for the birth/death sampler is 0.0306 while using both perturb and rotate gives 0.0203, a 33% reduction. The full results in Figures 12 and 13 of the Supplementary Material also demonstrate that these improvements are consistent across folds of the cross-validation study.

Variance	default birth/death	perturb	both	rotate
$\sigma^2 = 1.0$	3479	4539	5067	4850
$\sigma^2 = 0.01$	3462	4698	5051	4748

Table 3: Average runtimes of the four sampling schemes from the 10-fold cross-validation study of the Friedman function.

Average runtimes for the study are summarized in Table 3. These runtimes corroborate the shallower, more parsimonious trees found using the rotation sampler noted at the start of this subsection and in Table 2. For example, with $\sigma^2 = 1.0$ adding perturb increases average runtime by 30% and separately adding rotate increases average runtime by 40%. However, adding both perturb and rotate increases average runtime by only 45%. A similar behavior is evident when $\sigma^2 = 0.01$. In other words, the computational cost of introducing these various samplers is not simply additive as they interact with eachother. Namely, rotate helps find more parsimonious trees with fewer internal nodes to perturb – effectively reducing the cost of the perturb proposal.

6 Discussion

The underlying theme of the MCMC samplers we have discussed is to generate proposals that are consistent with the current state in tree-model space. This avoids needing to use the data to determine which proposals to try while also retaining a high acceptance rate. By consistent, we mean tree arrangements that could only be arrived at by the birth/death process – our proposals could be replaced by a sequence of birth/death moves, but because such a sequence of moves would need to traverse through a region of low posterior probability, it does not occur with reasonable probability in practice. This is a key idea underlying all three of the proposed sampling strategies outlined.

The three proposals described explore different sources of information that may account for variability in the posterior distribution. "Spatial" variability is explored by our perturbation proposal which proposes new cutpoint values that are consistent with the current tree. The change of variable proposal explores variability that comes from possible non-independence of covariates. One interpretation of this proposal is of a preconditioned version of the usual change of variable proposal, which leads to higher acceptance rates. Finally, we explore the structural variability of the tree posterior by the novel tree rotation proposal to move between tree structures that are nearly equivalent. This method performs local changes to the tree which, over many MCMC iterations, can yield trees with vastly different topological structures. It is also not restricted to

exploring trees with a fixed number of terminal nodes as the rotation proposal can arrive at trees of larger or smaller dimension. One interpretation of our rotation proposal is that it is similar to the usual swap proposal except we maintain consistency with the entire tree structure through the described rotation operation.

The approaches described are very different than Wu et al. (2007), which tries to explore all three sources of variability in one mechanism. Their approach is powerful, but is practically limited by the dimensionality of the covariates. An advantage of our approach is the user may decide which mechanisms to use according to their data or questions of interest. If it is known that the covariates are nearly independent, it is reasonable to dispense with change-of-variable proposals. If the user is only interested in prediction and does not require the interpretability from fully exploring the tree structure, then it may be reasonable to dispense with rotation proposals and rely entirely on perturbation. Ideally, one might like to use all three mechanisms to ensure accurate exploration of the posterior, but in practice a reasonable compromise might be needed.

In the examples we explored in this paper, the proposed methods gave good performance with acceptance rates at least in the low 20% range, and often higher. For the single tree example, we used the change-of-variable and tree rotation proposals which lead to posterior samples that fully explored all possible trees of depth 2. The computer experiments example demonstrated similarly good results. Using the tree rotation and perturbation proposals, the acceptance rate improved from 4% to 25% and the empirical coverage of the 90% credible interval improved from 53.8% to 90.6%.

In conclusion, the developments in this paper shed new light on ideas for improving the mixing of Bayesian regression tree models in situations where mixing is problematic.

Supplementary Material

Supplementary Material of "Efficient Metropolis–Hastings Proposal Mechanisms for Bayesian Regression Tree Models" (DOI: 10.1214/16-BA999SUPP; .pdf).

References

Bleich, J., Kapelner, A., George, E. I., and Jensen, S. (2014). "Variable Selection for BART: An Application to Gene Regulation." *The Annals of Applied Statistics*, 8(3): 1750–1781. MR3271352. doi: http://dx.doi.org/10.1214/14-AOAS755. 905

Chipman, H., George, E., and McCulloch, R. (1998). "Bayesian CART Model Search." *Journal of the American Statistical Association*, 93(443): 935–960. MR1631325. doi: http://dx.doi.org/10.2307/2670105. 885, 887, 888, 889, 900

Chipman, H., George, E., and McCulloch, R. (2002). "Bayesian Treed Models." *Machine Learning*, 48: 299–320. 885, 900

Chipman, H., George, E., and McCulloch, R. (2010). "BART: Bayesian Additive Regression Trees." *The Annals of Applied Statistics*, 4(1): 266–298. MR2758172. doi: http://dx.doi.org/10.1214/09-AOAS285. 885, 886, 889, 890, 897, 900, 905

Denison, D., Mallick, B., and Smith, A. (1998). "A Bayesian CART Algorithm." *Biometrika*, 85(2): 363–377. MR1649118. doi: http://dx.doi.org/10.1093/biomet/85.2.363. 885, 887, 888, 889, 900

Friedman, J. H. (1991). "Multivariate Adaptive Regression Splines." *The Annals of Statistics*, 19: 1–67. MR1091842. doi: http://dx.doi.org/10.1214/aos/1176347963. 890

Geweke, J. (1992). "Evaluating the Accuracy of Sampling-Based Approaches to the Calculation of Posterior Moments." In: *Bayesian Statistics 4*. Oxford University Press. MR1380276. 906

Gramacy, R. and Lee, H. (2008). "Bayesian Treed Gaussian Process Models with an Application to Computer Modeling." *Journal of the American Statistical Association*, 103(483): 1119–1130. MR2528830. doi: http://dx.doi.org/10.1198/016214508000000689. 885, 886, 887, 888, 890, 892, 900, 902

Higdon, D., Gattiker, J., Williams, B., and Rightley, M. (2008). "Computer Model Calibration Using High-Dimensional Output." *Journal of the American Statistical Association*, 103(482): 570–583. MR2523994. doi: http://dx.doi.org/10.1198/016214507000000888. 887

Kennedy, M. and O'Hagan, A. (2001). "Bayesian Calibration of Computer Models (with Discussion)." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68: 425–464. MR1858398. doi: http://dx.doi.org/10.1111/1467-9868.00294. 887

Lakshminarayanan, B., Roy, D. M., and Teh, Y. W. (2013). "Top-Down Particle Filtering for Bayesian Decision Trees." In: *Proceedings of the International Conference on Machine Learning*. ICML. 886

Lakshminarayanan, B., Roy, D. M., and Teh, Y. W. (2015). "Particle Gibbs for Bayesian Additive Regression Trees." arXiv:1502.04622v1. 886

Løland, A. and Høst, G. (2003). "Spatial Covariance Modeling in a Complex Coastal Domain by Multidimensional Scaling." *Environmetrics*, 14: 307–321. 903

Oakley, J. and O'Hagan, A. (2002). "Bayesian Inference for the Uncertainty Distribution of Computer Model Outputs." *Biometrika*, 89(4): 769–784. 887, 890

Pratola, M. T. (2016). "Supplementary Material of "Efficient Metropolis–Hastings Proposal Mechanisms for Bayesian Regression Tree Models"." *Bayesian Analysis*. doi: http://dx.doi.org/10.1214/16-BA999SUPP. 887

Pratola, M., Chipman, H., Gattiker, J., Higdon, D., McCulloch, R., and Rust, W. (2014). "Parallel Bayesian Additive Regression Trees." *Journal of Computational and Graphical Statistics*, 23: 830–852. MR3224658. doi: http://dx.doi.org/10.1080/10618600.2013.841584. 885, 890

Pratola, M. T., Harari, O., Bingham, D., and Flowers, G. E. (2015). "Design and Analysis of Experiments on Non-Convex Regions." *Technometrics*. doi: http://dx.doi.org/10.1080/00401706.2015.1115674. 903

Rathbun, S. L. (1998). "Spatial Modeling in Irregularly Shaped Regions: Kriging Estuaries." *Environmetrics*, 9: 109–129. 903

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). "Design and Analysis of Computer Experiments (with Discussion)." *Statistical Science*, 4: 409–423. MR1041765. 887, 890

Sleator, D., Tarjan, R., and Thurston, W. (1988). "Rotation Distance, Triangulations, and Hyperbolic Geometry." *Journal of the American Mathematical Society*, 3: 647–681. MR0928904. doi: http://dx.doi.org/10.2307/1990951. 886, 892

Taddy, M., Gramacy, R., and Polson, N. (2011). "Dynamic Trees for Learning and Design." *Journal of the American Statistical Association*, 106(493): 109–123. MR2816706. doi: http://dx.doi.org/10.1198/jasa.2011.ap09769. 885, 886, 890

Wu, Y., Tjelmeland, H., and West, M. (2007). "Bayesian CART: Prior Specification and Posterior Simulation." *Journal of Computational and Graphical Statistics*, 16(1): 44–66. MR2345747. doi: http://dx.doi.org/10.1198/106186007X180426. 886, 887, 890, 895, 900, 904, 909

Acknowledgments

The author thanks Dr. Hugh Chipman for helpful feedback on an earlier draft of this manuscript. The author also thanks two anonymous reviewers for their helpful comments and suggestions.