A new BART prior for flexible modeling with categorical predictors

Sameer K. Deshpande*

November 9, 2022

Abstract

Default implementations of Bayesian Additive Regression Trees (BART) represent categorical predictors using several binary indicators, one for each level of each categorical predictor. Regression trees built with these indicators partition the levels using a “remove one a time strategy.” Unfortunately, the vast majority of partitions of the levels cannot be built with this strategy, severely limiting BART’s ability to “borrow strength” across groups of levels.

We overcome this limitation with a new class of regression tree and a new decision rule prior that can assign multiple levels to both the left and right child of a decision node. Motivated by spatial applications with areal data, we introduce a further decision rule prior that partitions the areas into spatially contiguous regions by deleting edges from random spanning trees of a suitably defined network. We implemented our new regression tree priors in the flexBART package, which, compared to existing implementations, often yields improved out-of-sample predictive performance without much additional computational burden. We demonstrate the efficacy of flexBART using examples from baseball and the spatiotemporal modeling of crime.

*Dept. of Statistics, University of Wisconsin–Madison. sameer.deshpande@wisc.edu
1 Introduction

1.1 Motivation

Bayesian additive regression trees (BART; Chipman et al., 2010) is a Bayesian sum-of-trees model that, at a high-level, approximates an unknown function $f$ using a large ensemble of regression trees (i.e. piecewise constant step functions). Usually, the function $f$ is non-linear and involves complicated high-order interactions between several predictors. Such non-linearities and interactions are typically impossible to specify correctly \textit{a priori}. With BART, however, users often obtain extremely accurate predictions of function evaluations along with reasonably well-calibrated uncertainty intervals \textit{without having to pre-specify the functional form of $f$}. Better still, despite BART’s dependence on several prior hyper-parameters, users generally obtain excellent results using default values suggested by Chipman et al. (2010). The ease-of-use and generally excellent, tuning-free performance have made BART a popular “off-the-shelf” tool to be used within larger modeling workflows. For instance, BART-based methods have demonstrated great success at estimating treatment response surfaces and heterogeneous causal effects (Hill, 2011; Dorie et al., 2019; Hahn et al., 2020).

Despite BART’s popularity and success, we argue that BART’s handling of categorical predictors leaves much to be desired. Like many other statistical methods, BART converts each multi-level categorical predictor into several binary indicators, one for each level of the categorical variable. BART then treats each indicator like it treats any other continuous predictor. In doing so, however, BART severely limits its ability to “borrow strength” across different levels of a categorical variable. We present a simple alteration to the basic BART model that facilities much more flexible modeling with categorical predictors. To motivate our development, consider the following problems.

\textbf{Pitch framing in baseball}. In baseball, when the batter decides not to swing at a pitch, the umpire must classify the pitch as a ball or a strike. The official rules of baseball mandate deterministic ball/strike decisions: if any part of the pitched baseball passes through the strike zone — an imaginary rectangle over home plate that extends from the middle of the batter’s chest to their kneecaps — it must be called a strike. Despite the rule, observed ball/strike decisions are incredibly noisy, varying considerably within and between umpires. Many professional teams have heavily invested in catchers adept at \textit{pitch framing}, or systematically influencing umpires’ ball/strike decisions. To quantify the effect of pitch framing, Deshpande and Wyner (2017) estimated each catcher’s effect on called strike probabilities.
using a hierarchical logistic regression model that partially pooled data across umpires. Their model allowed the effect of pitch location to vary across umpires but assumed that the effects of individual players were constant across umpires. In so doing, their model could not probe the possibility that some (but not all) catchers could influence some (but not all) umpires on pitches thrown to some (but not all) locations.

A more realistic — and from the perspective of a professional team, more useful — model would incorporate two- and three-way interactions between umpires, pitch location, and players. Beyond the general difficulty in correctly specifying higher-order interactions, incorporating such interactions into Deshpande and Wyner (2017)’s model requires a huge number of additional parameters: nearly 100,000 parameters are needed to capture potential interactions between the roughly 100 catchers and 100 umpires! For context, there are only about 350,000 called pitches in a given season. The difficulty in elaborating Deshpande and Wyner (2017)’s parametric model to allow player, umpire, and location to interact makes a nonparametric approach based on BART rather compelling.

Crime in Philadelphia. Balocchi and colleagues have recently used Bayesian hierarchical models to study the spatiotemporal variation of crime at the census-tract level in the city of Philadelphia (Balocchi and Jensen, 2019; Balocchi et al., 2021, 2022) These works find that although models that “share strength spatially” often outperform models that do not, failing to account for spatial discontinuities can produced highly biased crime forecasts. Spatial discontinuities can occur along visible geographic landmarks like highways, parks, or rivers. But they can also coincide with less visible differences in neighborhood-level demographic and socioeconomic measures that have resulted, at least in part, from discriminatory policies like redlining (Jacoby et al., 2018; Rhynhart, 2020).

Figure 1 illustrates the challenges faced when modeling crime in Philadelphia. Figure 1a shows the spatial variability in the mean levels of a transformation of the monthly crime density (defined as crime counts per square mile; see Section 4.3 for details) in each of Philadelphia’s 384 census tracts averaged across the 16 year period from January 2006 to December 2021. Although some spatially adjacent tracts have very similar means and time series, other tracts display markedly different crime patterns than their neighbors (Figure 1b).
Our exploratory analysis suggests that an accurate model of the spatiotemporal variability in crime density must account for (i) non-linearities in time and (ii) interactions between time and space. Unfortunately, correctly pre-specifying such non-linear interactions in a parametric model is extremely difficult, motivating a more flexible, nonparametric approach. Our ideal model would exploit spatial similarities when they are present but also adapt to spatial discontinuities suggested by the data. In other words, we want to model spatial variation in a data-adaptive fashion, rather than imposing strong spatial smoothness through, say, rigid conditionally auto-regressive priors on spatial random effects.

1.2 Our Contributions

In both problems, we want to flexibly model a function that may (i) be highly non-linear; (ii) depend on the complex interaction and interplay between categorical and continuous covariates; and (iii) have a functional form that is difficult to pre-specify correctly in a parametric model. From this perspective, BART appears ideally suited for modeling the called strike probabilities and the Philadelphia crime data.
Unfortunately, as we argue conceptually in Section 3.1 and demonstrate empirically in Section 4, the original BART model performs quite poorly in these contexts. BART’s surprisingly poor performance on these problems derives from the way it operationalizes categorical predictors. Briefly, the use of binary indicators to represent players and umpires prevents BART from pooling data across multiple groups of players and umpires in the pitch framing problem. Further, by representing census tracts with separate binary indicators in the crime modeling problem, BART is unable to leverage the tracts’ spatially adjacency structure and adapt to any spatial smoothness.

We present a simple alteration to BART that facilitates much more flexible modeling with categorical predictors, including those displaying network structure. Formally, we introduce a prior over a more general class of regression trees that produce a richer range of partitions of discrete space. The remainder of the paper is structured as follows. We carefully review the original BART model in Section 2 before revealing how one-hot encoding categorical variables severely restricts the representational flexibility of regression tree ensembles in Section 3.1. Then, in Section 3.2, we introduce our more general class of regression trees and specify a prior that facilitates fitting more flexible BART models. Using semi-synthetic and real data examples, we illustrate the improvements in predictive quality offered by our new class of trees in Section 4. We conclude in Section 5 with a discussion of several extensions of our modeling framework.

2 A review of the original BART model

Chipman et al. (2010) introduced BART in the context of the general nonparametric regression problem: given \( n \) observations of a pair of a \( p \)-dimensional covariate vector \( \mathbf{x} \) and a scalar output \( y \) that are modeled as \( y \sim \mathcal{N}(f(\mathbf{x}), \sigma^2) \), we want to learn the regression function \( f \). We briefly review the basic BART model in the setting where all \( p \) covariates are continuous and lie, without loss of any generality, in the interval \([0, 1]\). BART works by approximating \( f \) as a sum of \( M \) binary regression trees, specifying a prior over the collection of trees, and then computing a posterior distribution over tree ensembles, which in turn induces an approximate posterior distribution over the function \( f \).
2.1 A sum-of-trees model

A regression tree is a triplet \((T, D, \mu)\) consisting of (i) a finite rooted binary tree \(T\) containing \(L(T)\) terminal or leaf nodes and a collection of non-terminal nodes; (ii) a collection of decision rules \(D\), one for each non-terminal node in \(T\); and (iii) a collection \(\mu\) of \(L(T)\) scalars or jumps, one for each leaf node in \(T\). When \(T\) contains a non-root node, every non-terminal node \(nx\) is connected to two children, which we refer to as the left and right child of \(nx\).

The basic BART model parametrizes the decision rule associated the non-terminal node \(nx\) with a pair \((v(nx), c(nx))\), where \(v(nx) \in \{1, 2, \ldots, p\}\) is the splitting variable index and \(c(nx) \in [0, 1]\) is the cutpoint. For every \(x = (x_1, \ldots, x_p) \in [0, 1]^p\); we can trace a path from the root of \(T\) to exactly one leaf node as follows. Starting from the root, once the path reaches the non-terminal node \(nx\), it proceeds to the left child of \(nx\) if the \(v(nx)\)-th coordinate of \(x\) is less than \(c(nx)\) (i.e. if \(x_{v(nx)} < c(nx)\)) and to the right child of \(nx\) otherwise (i.e. if \(x_{v(nx)} \geq c(nx)\)). The decision-following path for each \(x\) terminates at a single leaf node. By associating each point \(x \in [0, 1]^p\) with exactly one of the \(L(T)\) leaf nodes, the decision tree \((T, D, \mu)\) partitions \([0, 1]^p\) into \(L(T)\) disjoint sets, one for each leaf node. In fact, these sets are axis-parallel rectangular boxes. By associating each leaf \(\ell\) with a scalar jump \(\mu_\ell \in \mathbb{R}\), the regression tree \((T, D, \mu)\) represents a piecewise constant step function (see Figure 2 for several examples).

BART represents unknowns functions using a sum of \(M\) a priori independent regression trees. Because the total number of parameters, which include all of the tree structures, decision rules, and jumps, increases rapidly with \(M\), taking \(M\) to be large endows BART with tremendous representational flexibility. Indeed, Chipman et al. (2010) recommended using \(M = 200\) regression trees as a default. As Figure 2 suggests, rather complicated functions, which might be represented with a single deep tree with many leaf nodes, can usually be written as a sum of simple regression trees containing only a few leaf nodes. Accordingly Chipman et al. (2010) specified a regression tree prior that aggressively regularized towards shallow trees containing a small number of leaves.

2.2 The default BART regression tree prior

Chipman et al. (2010) specified their regression tree prior implicitly using a stochastic process for simulating prior draws. We can describe the prior tree-drawing process using three steps. In the first step, the process samples the tree \(T\) using a branching process first described in Chipman et al. (1998). Then, conditionally on \(T\), the process sequentially draws random
Figure 2: The sum of regression trees is also a regression tree whose jumps are sums of jumps from each tree in the ensemble. Relatively smooth functions can be well-approximated using ensembles of rather simple, discontinuous step functions.

decision rules at each non-terminal node. Finally, conditionally on $T$, the process draws the $L(T)$ jumps in $\bm{\mu}$ independently from a $\mathcal{N}(0, \tau^2/M)$ distribution.

**Drawing $T$.** Chipman et al. (1998)’s branching process grows $T$ in a recursive fashion starting from a single root node, which is considered to be a terminal node of depth zero. Each time a new terminal node is created at depth $d$, the branching process grows the tree from that node by adding two children nodes with probability $0.95(1 + d)^{-2}$. Because the probability that the process continues growing decays quadratically with tree depth, the branching process terminates with finitely many leaves.

**Drawing $D|T$.** The decision rule at non-terminal node $\mathbf{nx}$ is drawn in two steps. First, the splitting variable index $v(\mathbf{nx})$ is drawn uniformly from the set $\{1, 2, \ldots, p\}$. Then, conditionally on $v(\mathbf{nx})$, $c(\mathbf{nx})$ is drawn uniformly from a set $\mathcal{A}(v(\mathbf{nx}), \mathbf{nx})$ that contains all values of the $v(\mathbf{nx})$-th covariate available at the node $\mathbf{nx}$. The decision rules associated with the ancestors of $\mathbf{nx}$ in the tree fully determine the set $\mathcal{A}(v(\mathbf{nx}), \mathbf{nx})$. Conceptually, we could compute $\mathcal{A}(v(\mathbf{nx}), \mathbf{nx})$ by scanning over all $\mathbf{x} \in [0, 1]^p$ and adding $x_{v(\mathbf{nx})}$ to the set if $\mathbf{x}$’s decision-following path reaches $\mathbf{nx}$. In practice, we can quickly compute $\mathcal{A}(v(\mathbf{nx}), \mathbf{nx})$ by recursing up the tree from $\mathbf{nx}$ to the root and taking appropriate set intersections.
Remark 1. Before proceeding, we pause to note that the three-step tree-drawing process described above differs slightly from the process described in Chipman et al. (2010, §2.2). In their process, decision rules are drawn as part of the tree-generating branching process. Practically, simultaneously growing $T$ and $D$ is equivalent to first drawing $T$ and then conditionally drawing $D$ as we describe.

When all $p$ covariates are continuous, the set $\mathcal{A}(v(nx), nx)$ will be an interval. By drawing $c(nx) \in \mathcal{A}(v(nx), nx)$, we ensure that (i) each non-trivial node is visited by the decision-following path of at least one $x \in [0, 1]^p$ and (ii) the decision tree $(T, D)$ partitions $[0, 1]^p$ into exactly $L(T)$ axis-parallel rectangular boxes. In this way, the default regression tree prior used by BART induces a prior over the space of partitions of $[0, 1]^p$ into such boxes. It turns out, moreover, the induced prior on partitions places positive density on every such partition of $[0, 1]^p$. Coupled with the absolutely continuous $\mathcal{N}(0, \tau^2/M)$ prior on the jumps, the regression tree prior places positive density on essentially every piecewise constant step function.

Remark 2. Whereas we allow cutpoints $c(nx)$ to be drawn uniformly from an interval, Chipman et al. (2010) restrict cutpoints to the set of observed covariate values. Their implementation in the BART package (Sparapani et al., 2021) further truncates the prior to those trees in which every leaf node is reached by the decision-following path of at least five observations\(^1\). Such restrictions sacrifice Bayesian coherence and place zero prior probability on regression trees whose cutpoints lie outside the set of observed covariate values. The prior described above, in contrast, makes no such restrictions, thereby retaining Bayesian coherence and the desirable full support property.

2.3 Posterior computation

Chipman et al. (2010) proposed a Gibbs sampling strategy to simulate posterior samples of the regression tree ensemble. In each iteration, each regression tree is updated conditionally fixing the remaining $M - 1$ trees. Two steps are used to update individual regression trees. First, a new decision tree $(T, D)$ is drawn from its conditional posterior distribution given the remaining $M - 1$ regression trees and the data. Then, a new collection of jumps $\mu$ is drawn from its conditional posterior distribution given the just-drawn decision tree $(T, D)$, the other $M - 1$ regression trees, and the data.

\(^1\)The restriction is enforced on line 47 of the internal bd() function here.
The new decision tree \((T,D)\) is drawn with a Metropolis-Hasting (MH) step. While there are several proposal distributions encouraging both small and large modifications to both \(T\) and \(D\) (see, e.g., Pratola, 2016; Mohammadi et al., 2020), we here focus on a simple proposal distribution that either randomly prunes or grows the decision tree. Pruning \((T,D)\) involves (i) deleting two leaf nodes that share a common parent, which is selected uniformly at random, and their incident edges in \(T\); (ii) deleting the decision rule associated with the parent of the deleted leaf nodes; and (iii) leaving the rest of the tree and decision rules unchanged. Growing \((T,D)\) in contrast, involves (i) adding two nodes to the tree and connecting them to an existing leaf node selected uniformly at random; (ii) drawing a decision rule for the selected node; and (iii) leaving the rest of the tree and decision rules unchanged. For simplicity, when growing a tree, we draw the decision rule for the newly-created non-terminal node from the prior introduced in Section 2.2; that is, we (i) draw the splitting variable index uniformly and (ii) uniformly draw a cutpoint from the interval of covariate values available at the node.

### 3 BART with a more flexible class of regression trees

When all \(p\) covariates are continuous and lie in the unit interval, the default BART prior described in Section 2.2 places (i) positive density on every partition of \([0,1]^p\) into disjoint axis-parallel rectangular boxes and (ii) positive density on every piecewise constant function defined over such partitions. Together, these full support properties endow BART with tremendous flexibility: basically, BART is not prevented \textit{a priori} from using any given step function to approximate \(f(x)\).

However, as we will demonstrate in Section 3.1, when categorical covariates are converted into binary indicators, the default BART prior places zero prior probability on a staggeringly overwhelming majority of partitions of the covariate space, which is the product of continuous and unordered discrete spaces. Consequently, BART is prevented from using the great majority of step functions defined over the covariate space to help approximate \(f\), severely diminishing its representational capacity. In Section 3.2, we propose a simple modification — namely a more general type of decision rule — that substantially expands the representational capacity of BART’s regression tree ensemble.
3.1 A binary indicator-induced hole in the BART prior

Consider, for the moment, a setting with no continuous covariates and only one categorical covariate $X$ that takes values in $\{\lambda_1, \ldots, \lambda_K\}$. By default, BART operationalizes the predictor by introducing $K$ binary indicators, $X_1, \ldots, X_K$, where $X_k = 1 (X = \lambda_k)$ for each $k = 1, \ldots, K$. Suppose that $(T, D, \mu)$ is a regression tree with at least two leaf nodes drawn using the three-step process described in Section 2.2. Since the decision tree $(T, D)$ partitions the entire space $[0, 1]^D$, it automatically partitions the set of standard basis vectors $\{e_1, \ldots, e_K\}$ where the $k$-th entry of $e_k$ is equal to one and all other entries are equal to zero. Since we can identify $e_k$ with the level $\lambda_k$ of $X$, we conclude that the decision tree $(T, D)$ partitions the levels of $X$. In fact, $(T, D)$ partitions the levels of $X$ into several singleton sets with one level each and at most one set with multiple levels. Figure 3a shows two such examples of the partitions of the Philadelphia census tracts induced by the default BART prior. Both of these partitions feature one large set of tracts and some singleton sets.

To understand why the default prior can produce only such partitions, consider the decision-following paths for each of the $e_k$'s. Without loss of generality, suppose the splitting variable index at the root is 1. Because the first coordinates of $e_2, \ldots, e_K$ are all zero, the paths for these $K - 1$ points will proceed to the left child of the root while the path for $e_1$ will proceed to the right child. Consequently, when partitioning the points $\{e_1, \ldots, e_K\}$, the tree separates $e_1$ in a singleton set away from the other $K - 1$ points. At every subsequent non-terminal node, at most one path can proceed to the right child while all others proceed to the left child. In this way, by tracing the decision-following paths of the $e_k$'s, we see that the decision tree $(T, D)$ recursively partitions the set $\{\lambda_1, \ldots, \lambda_K\}$ by removing at most one element at a time. Ultimately, then, the default BART prior places positive probability on only those partitions that can formed by recursively removing at most one element at a time.

It turns out that there are only $2^K - K$ such partitions. When $X$ contains more than two levels (i.e. $K > 2$), this number is much smaller than the total number of partitions of $K$ objects (i.e. the $K$-th Bell number; Stanley (1997, pg. 33)). For instance, when $K = 5$, the default BART prior places positive probability on only 27 of the 52 possible partitions and when $K = 10$, the default BART prior supports less than 1% of all possible partitions (1,014 versus 115,975). So by converting the inherently discrete and unordered $X$ into several continuous ordered covariates, the default BART prior places zero probability on a potentially enormous number of partitions of the levels of $X$. 

10
Figure 3: Examples of partitions of the Philadelphia census tracts induced by regression trees drawn using (a) the default BART prior’s “remove one at a time” strategy, (b) our new prior with the uniform unordered process, and (c) our new prior with the uniform spanning tree process. In each example, we have colored the census tracts according to their corresponding leaf in the regression tree.

3.2 A new class of decision rules

The argument in Section 3.1 extends naturally to more realistic settings featuring both continuous and categorical predictors. Namely, if we convert each categorical predictor into binary indicators, the default BART prior place positive density on an extremely narrow range of step functions defined over the covariate space, which is a product of continuous and unordered discrete spaces. Consequently, BART is severely limited in its ability to model functions that display similar variation over multiple levels of a categorical predictor. Concretely, in the crime modeling problem, BART cannot include functions that are constant within each of the circled regions of Figure 1a in the ensemble.

To expand BART’s flexibility when some covariates are categorical, we recommend against
one-hot encoding categorical predictors. Formally, suppose that we have $p_{\text{cont}}$ continuous predictors $X_1, \ldots, X_{p_{\text{cont}}}$ which, without loss of generality, lie in the interval $[0, 1]$, and $p_{\text{cat}}$ categorical predictors $X_{p_{\text{cont}}+1}, \ldots, X_p$ where $p = p_{\text{cont}} + p_{\text{cat}}$. Further suppose that the $j$-th categorical covariate $X_{p_{\text{cont}}+j}$ takes on $K_j$ values that are contained in the set $\mathcal{K}_j$. We wish to use BART to estimate a function $f : [0, 1]^{p_{\text{cont}}} \times K_1 \times \cdots \times K_{p_{\text{cat}}} \to \mathbb{R}$. Observe that the covariate space is the product of continuous ordered spaces and discrete unordered spaces.

Unfortunately, we can no longer use the regression trees introduced in Section 2 for this purpose because the behavior of the decision-following paths described above is undefined when some coordinates of $x$ are unordered. Fitting more flexible BART models with both continuous and unordered categorical covariates requires a more general class of decision rule. Specifically, we instead work with decision rules $(v, \mathcal{C})$, which are parametrized by a variable splitting index $v$ and a cutset $\mathcal{C}$. With such rules, we can trace a decision-following path from root to leaf for each $x \in [0, 1]^{p_{\text{cont}}} \times K_1 \times \cdots \times K_{p_{\text{cat}}}$ as follows. Starting from the root, once the path reaches a non-terminal node $\mathbf{n}_x$, the path proceeds to the left child if $x_v(\mathbf{n}_x) \in \mathcal{C}(\mathbf{n}_x)$ and to the right child otherwise.

**A new decision rule prior.** Fitting a BART model using regression trees with decision rules of the form $(v, \mathcal{C})$ formally requires specifying (i) a prior over the new class of regression trees and (ii) a proposal distribution for updating decision trees $(T, \mathcal{D})$ with Metropolis-Hastings. Insofar as we are only changing the decision rules, we will use exactly the same priors for the tree $T$ and jumps $\mu|T$ described in Section 2.2. We only introduce a new decision rule prior. Further, just like in Section 2.3, we only use grow and prune proposals in our Gibbs sampler and draw decision rules created during grow moves from our new prior.

Like before, we specify our new decision rule prior at non-terminal node $\mathbf{n}_x$ implicitly by describing a process for simulating prior draws. At non-terminal node $\mathbf{n}_x$, we first draw the variable splitting index $v(\mathbf{n}_x)$ uniformly as before. Then, we compute $\mathcal{A}(v(\mathbf{n}_x), \mathbf{n}_x)$, the set of available values for the $v(\mathbf{n}_x)$-th predictor at node $\mathbf{n}_x$. To ensure that the resulting decision tree $(T, \mathcal{D})$ partitions the space $[0, 1]^{p_{\text{cont}}} \times K_1 \times \cdots \times K_{p_{\text{cat}}}$, we take $\mathcal{C}(\mathbf{n}_x)$ to be a randomly drawn non-trivial subset of $\mathcal{A}(v(\mathbf{n}_x), \mathbf{n}_x)$. That is, we draw $\mathcal{C}(\mathbf{n}_x) \subset \mathcal{A}(v(\mathbf{n}_x), \mathbf{n}_x)$ with the restriction that it is non-empty and not equal to $\mathcal{A}(v(\mathbf{n}_x), \mathbf{n}_x)$.

How we draw the cutset depends on whether the $v(\mathbf{n}_x)$-th covariate is continuous or categorical. Briefly, when the $v(\mathbf{n}_x)$-th covariate is continuous, we draw the cutset using what we call the *continuous uniform process*. When the $v(\mathbf{n}_x)$-th covariate is categorical, if we do not have an *a priori* preference to co-cluster certain pairs of levels more often than others,
we draw the cutset using what we call the *uniform unordered process*. On the other hand, if we specify a network encoding our co-clustering preferences, we draw the cutset using the *uniform spanning tree process*. We detail these processes in Section 3.3.

3.3 Conditional priors over cutsets \( \mathcal{C} \)

**The continuous uniform process.** When the \( \nu(\mathbf{x}) \)-th covariate is continuous, \( \mathcal{A}(\nu(\mathbf{x}), \mathbf{x}) \) is an interval and we draw the cutset \( \mathcal{C}(\mathbf{x}) \) by first selecting \( c \in \mathcal{A}(\nu(\mathbf{x}), \mathbf{x}) \) uniformly and then setting \( \mathcal{C} = \mathcal{A}(\nu(\mathbf{x}), \mathbf{x}) \cap [0, c) \). This process produces exactly the same sorts of decision rules as the process described in Section 2.2.

**The uniform unordered process.** When the \( \nu(\mathbf{x}) \)-th covariate is categorical, the set of available values \( \mathcal{A}(\nu(\mathbf{x}), \mathbf{x}) \) is discrete and we now must draw a non-trivial random subset \( \mathcal{C}(\mathbf{x}) \subset \mathcal{A}(\nu(\mathbf{x}), \mathbf{x}) \). Arguably the simplest option is to sample \( \mathcal{C}(\mathbf{x}) \) uniformly from the set of all non-trivial subsets of \( \mathcal{A}(\nu(\mathbf{x}), \mathbf{x}) \), using what we call the discrete uniform process. Practically, the process works by initializing \( \mathcal{C}(\mathbf{x}) = \emptyset \) and then sequentially adding elements of \( \mathcal{A}(\nu(\mathbf{x}), \mathbf{x}) \) to \( \mathcal{C}(\mathbf{x}) \) with probability 1/2. If, at the end of this sweep, \( \mathcal{C}(\mathbf{x}) \) remains empty or is exactly \( \mathcal{A}(\nu(\mathbf{x}), \mathbf{x}) \), we reset \( \mathcal{C}(\mathbf{x}) = \emptyset \) and repeat the sweep until \( \mathcal{C}(\mathbf{x}) \) is non-trivial.

Because they induce partitions over the entire covariate space \([0, 1]^{p_{\text{cont}}} \times \mathcal{K}_1 \times \cdots \times \mathcal{K}_{p_{\text{cat}}} \), regression trees drawn from our new prior partition each of the discrete sets \( \mathcal{K}_j \). In fact, our discrete uniform process ensures that the induced prior over partitions of \( \mathcal{K}_j \) places positive probability on all possible partitions, in sharp contrast to the “remove at most one at a time” process described in Section 2.2. Figure 3b shows two examples of the partitions of the census tracts in Philadelphia induced by regression trees drawn using our new prior equipped with the uniform unordered process. Unlike the partitions in Figure 3a, partitions drawn from our new prior can contain multiple multi-element sets. By symmetry, each level of the categorical predictor is equally likely to be co-clustered with every other level.

**Remark 3.** Our uniform unordered process is biased towards drawing cutsets containing about half of the available levels. Rather than fixing the probability of inclusion into \( \mathcal{C} \) to be 0.5, a more general process would first sample the inclusion probability and then assign elements of \( \mathcal{A} \) to \( \mathcal{C} \) with that probability. Such generalization also places positive probability on all partitions of the sets of categorical levels. Chipman et al. (1998) proposed exactly such a generalization in the prior underlying their single-tree Bayesian CART model. Curiously, however, they later abandoned the prior in BART, opting instead for the more conventional
binary encoding of categorical variables.

The uniform spanning tree process. A closer examination of the partitions in Figure 3b reveals that the clusters of tracts are not spatially contiguous. The resulting regression trees effectively pool data from geographically disparate census tracts. In light of the spatial smoothness in Figure 1a, we might instead prefer spatially contiguous partitions like those in Figure 3c, which only pool data from neighboring tracts.

When we have spatial data aggregated into areal units (e.g. blocks, census tracts, states), we can represent the areas with a network $G$ whose vertices correspond to the areal units and whose edges encode spatial adjacencies. Figure 7a shows the adjacency network of the Philadelphia census tracts. When modeling such areal data, we can represent location using the vertex label in $G$, which is a structured categorical variable.

More generally, we might sometimes express a priori preference that certain categorical levels be co-clustered more often than others, at odds with the symmetric nature of the uniform unordered process. In these scenarios, we can represent those preferences using a network $G$ whose vertices are the levels of the categorical covariate. We further draw edges in $G$ between the levels that we prefer to co-cluster. In the crime modeling example, the network edges encode the preference that spatially adjacent tracts co-cluster more frequently together than non-adjacent tracts. As evidenced by Figure 3b, recursive partitioning with the uniform unordered process tends to create partitions whose clusters do not induce connected subgraphs of $G$. The following uniform spanning tree process produces cutsets that more faithfully respect $G$’s connectivity.

Given a set of available levels $A$ at a non-terminal node, the uniform spanning tree process draws a random subset $C \subseteq A$ such that both $C$ and $A \setminus C$ induce connected subgraphs of $G$. The process works by partitioning $G[A]$, the subgraph of $G$ induced by $A$, into two connected components. Specifically the process first draws a spanning tree of $G[A]$ uniformly at random by performing a loop-erased random walk on the vertices of $G[A]$ (i.e., we use Wilson’s algorithm; Wilson, 1996). Then it deletes a uniformly selected edge from the spanning tree, thereby partitioning the spanning tree into two connected components. The process then assigns the vertices in one of these components to $C$. Figure 4 illustrates how our uniform spanning tree process works.

Our regression tree prior with the uniform spanning tree process places positive probability on every partition of a network into finitely many connected components. Our uniform
Figure 4: Cartoon illustration of our uniform spanning tree process. (a) Induced subgraph of levels available at a particular non-terminal node in the decision tree. (b) Uniformly drawn random spanning tree. (c) Deleting a uniformly selected edge (the edge (2, 8) in this case) from the spanning tree disconnects the tree and partitions the vertices into two subsets. (d) The subgraphs induced by the two vertex sets produced in (c) remain connected.

The uniform unordered and uniform spanning tree process adaptively construct new covariates on which a regression tree can split. These new covariates have the form $1(x_v \in C)$, which can be written as the sum of binary indicators corresponding to each element of $C$ when $X_v$ is categorical. In other words, these processes allow trees to split on new covariates formed by agglomerating multiple levels of a categorical covariate into a new covariate. Breiman similarly suggested randomly combining categorical levels in his random forests algorithm (Breiman, 2001, §5.1).
3.4 Our new regression tree prior

To summarize, our new regression trees are based on decision rules that send paths to the left if \( x_v \in C \) and to the right otherwise. We draw the cutset \( C \) based on the type of the splitting variable at each non-terminal node. When the splitting variable is continuous, we draw \( C \) from the uniform continuous process. When the splitting variable is categorical, we use the uniform unordered process if we have no \textit{a priori} preferences about how the categorical levels ought to co-cluster. We use the uniform spanning tree process if we have specified a network encoding co-clustering preferences amongst the levels.

4 Implementation and empirical results

For posterior computation with our new regression trees, we use a Gibbs sampling strategy that is virtually identical to the one described in Section 2.3. The only substantive change is that we draw the cutsets of decision rules introduced in grow proposals using one of the processes described in Section 3.3. Relative to the sampler described in Section 2.3, our sampler can be more computationally demanding because we must run Wilson’s algorithm each time we draw a cutset using the uniform spanning tree process.

We implemented our sampler in a new \texttt{R} package that we call \texttt{flexBART}, which is available at \texttt{https://github.com/skdeshpande91/flexBART}. While \texttt{flexBART}'s architecture is heavily inspired by that of Sparapani et al. (2021)'s \texttt{BART} package, we have made substantial changes and improvements, which we briefly summarize in Section 4.1. In Section 4.2, we report the results of a cross-validation experiment with the pitch framing data that highlights the benefits of trees built using the uniform unordered process relative to trees built using binary indicators. Then, in Section 4.3, we report the results of a full leave-one-out analysis of the Philadelphia crime data that demonstrates the additional benefit of the uniform spanning tree process relative to the uniform unordered process. Finally, we describe a semi-synthetic simulation study of a more general network-indexed regression problem in Section 4.4. In each, experiment we compared the performance of our implementation (hereafter \texttt{flexBART}) to the \texttt{BART} package’s implementation of BART (hereafter \texttt{BART}) and Linero (2018)'s variant of BART that draws the splitting variable indices \( v(n_x) \) from a sparsity-inducing Multinomial-Dirichlet hierarchical prior (hereafter \texttt{DART}). We ran our experiments on a high-throughput computing cluster.
4.1 Implementation details

Closely following the BART package, we wrote flexBART’s core sampling functions in C++. Although the overall design of our code is similar to theirs, we entirely re-built the underlying regression tree class to accommodate our new decision rules. While implementing our sampler, we noticed that the sampler in BART performs many redundant calculations in every regression tree update. Basically, while updating a single tree, they identify the leaf to which every observation is associated by tracing every decision-following path from the root to a leaf. Because at most two leaf nodes are changed during the MH step, the map from observations to leaf nodes does not change much iteration to iteration. Consequently, many of the path-tracings performed in the BART implementation are largely redundant. Eliminating these redundant calculations led to substantial speedup. Further, as evidenced by our experiments in Section 4.3 and 4.4, the speedup more than offsets the additional burden of repeatedly running Wilson’s algorithm.

4.2 Pitch framing

We scraped pitch-by-pitch data from each Major League Baseball season between 2013 and 2019 using the baseballr package (Petti and Gilani, 2022). Our dataset includes the ball/strike decision; the horizontal and vertical coordinates of each pitch as it crosses the front edge of home plate (i.e. the pitch location); the identifies of the batter, catcher, pitcher, and umpire involved in a pitch; and several other contextual covariates like the inning number, the number of previously called balls and strikes in the at-bat, and whether there were any baserunners when the pitch was thrown. In each season, there were around 350,000 called pitches, of which about 34% were called strikes, and around 1000 batters, 100 catchers, 800 pitchers, and 90 umpires.

Letting $y$ be a binary indicator of ball ($y = 0$) or strike ($y = 1$) and concatenating pitch location, player and umpire identifies, and contextual covariates into the vector $x$, we fit a BART model with probit link; that is, we use BART to learn a function $f(x)$ satisfying $P(y = 1) = \Phi(f(x))$ where $\Phi$ is the standard normal cumulative distribution function. For each season, we performed a 10-fold cross-validation experiment to compare the predictive performances of flexBART, BART, and DART.

Note that BART and DART convert the categorical player and umpire identities into roughly 2000 additional covariates per season. When forming each training–testing split, we observed that some players were included only in the testing set. In this way, some of the binary
indicators created by BART and DART are constant on the training dataset. By default both BART and DART drop such constant covariates, thereby ignoring certain players and umpires during training. In contrast, flexBART remains aware of all possible players and umpires during training, even if it does not observe data for some of them. To facilitate a fair comparison, we did not allow BART and DART to drop these constant columns by running both with the option rm.const = FALSE. In this way, all methods remained aware of all possible players and umpires during training.

For each implementation, we simulated 2,000 iterations of a single Markov chain and discarded the first 1,000 samples as burn-in. We estimated the posterior mean called strike probability for each held-out pitch. Figure 5 compares three error metrics — Brier score, misclassification rate, and log-loss — computed on the 70 training and testing folds we created for this experiment. These error metrics were computed using each implementation’s posterior mean estimates of the called strike probability.

Figure 5: The Brier score, misclassification rate, and log-loss of BART, DART, and flexBART on every training (top row) and testing (bottom row) fold in our pitch framing experiment.
Figure 5 reveals BART’s and DART’s exceptionally poor out-of-sample predictive performance. Such performance is driven almost entirely by gross overfitting, especially to players and umpires who appeared relatively infrequently (if at all) in the training data. As one example, BART and DART tended to return predicted called strike probabilities near zero (resp. near one) every time they encountered a player or umpire in the testing data who appeared only once in the training data and was involved in a pitch that was called a ball (resp. a strike). Further, both BART and DART were extremely confident in these predictions. flexBART, in sharp contrast, tended to return much less extreme predictions about called strike probabilities for pitches featuring infrequently occurring players and umpires. Thanks to its underlying uniform unordered process, flexBART was able to partially pool data across a wider range of players and umpires than the other two methods, thereby providing more regularization to predictions made about infrequently observed players and umpires. We further observed flexBART’s in- and out-of-sample error metrics were quite similar to each other and both were considerably smaller than those reported by Deshpande and Wyner (2017). We further found that, averaging across all seasons and cross-validation folds, flexBART drew 2,000 posterior samples in about 48 minutes, which was considerably faster than both BART (4.9 hours) and DART (5.3 hours). Together, these results point to the benefits of (i) BART-based models of called strike probabilities over Deshpande and Wyner (2017)’s parametric models; (ii) our new regression trees over the ones used by the BART package; and (iii) eliminating redundant computations in our implementation.

4.3 Philadelphia crime data

We obtained our crime data from opendataphilly.org, where the Philadelphia Police Department releases the type, time, and location (longitude and latitude) of every reported crime in the city. From the raw data, we computed the number of crimes reported in each census tract in each month between January 2006 and December 2021. Following Balocchi et al. (2022), rather than directly modeling the crime counts, we work instead with crime densities, defined as number of crimes per square mile, which are more directly comparable across heterogeneous regions. To counteract the considerable skewness in the monthly crime densities, we applied an inverse hyperbolic sine transformation to the densities. As noted by Burbidge et al. (1988), this transformation is well-defined at zero and is otherwise extremely similar to the logarithmic transformation.

Formally, let \( c_{v,t} \) be the total number of crimes reported in census tract \( v \) at time \( t \), with
Letting $A_v$ be the area of census tract $v$ in square miles, we computed

$$y_{v,t} = \log \left( \frac{c_{v,t}/A_v}{1 + (c_{v,t}/A_v)^2} \right) - \log(2).$$

For each tract $v$ and all times $t = 1, 2, \ldots, 192$, we model $y_{v,t} \sim \mathcal{N}(f(t,v), \sigma^2)$. Note that the function $f$ depends on a discrete ordered covariate (the time index $t$) and an unordered categorical covariate (the census tract label $v$). In our experiments, we rescaled the time indices to the interval $[0, 1]$ and modified the continuous uniform process by additionally intersecting the cutset with the set of rescaled time indices.

We performed a full leave-one-out experiment in which we trained our BART models on all the data from all but one census tracts and attempted to predict the entire trajectory of transformed crime densities for the held-out tract. We compared two versions of flexBART to BART and DART: the first version used the uniform unordered process (hereafter flexBART-UU) and second used the uniform spanning tree process (hereafter flexBART-UST) to split on the census tract label. For each cross-validation fold, we did not delete the held-out vertex and its incident edges from the network encoding tract adjacency, which is shown in Figure 7a. In a sense, our leave-one-out experiment probes the ability of various BART implementations to “in-fill” or interpolate observations at an existing vertex of a network rather than extrapolate to entirely new vertices. Figure 6 compares the root mean square error on each training and testing fold and the runtime of each implementation.

Figure 6: Root mean square errors on each training (left) and testing (middle) fold in our leave-one-out analysis of the Philadelphia crime data. Runtimes (log-scale) for each implementation (right).
Unsurprisingly, \texttt{flexBART-UST}, which leveraged data from surrounding tracts to make predictions in the held-out tract, displayed much better out-of-sample predictive performance than the other three implementations. It is interesting, however, to note that despite \texttt{flexBART-UU} pooling data across multiple tracts, it often demonstrated worse predictive performance than \texttt{BART} and \texttt{DART}. Essentially, in borrowing strength across tracts in a relatively non-informative and non-spatially contiguous fashion, \texttt{flexBART-UU} was unable to adapt to and exploit the spatial smoothness present in Figure 1a. Additionally, the large discrepancy between \texttt{DART}'s training and testing RMSE suggests that it severely overfit the training data.

For each cross-validation fold, the training dataset contained 73,536 total observations (192 observations from 383 census tracts). Because the \texttt{flexBART} implementations do not have to loop over these observations multiple times per tree update, they are considerably faster than the \texttt{BART} implementations: \texttt{flexBART-UU} drew 2,000 posterior samples in about seven minutes while \texttt{BART} needed upwards of two and a half hours to draw the same number of samples. Because it repeatedly runs random walks on networks when drawing cutsets, \texttt{flexBART-UST} is predictably slower than \texttt{flexBART-UU} (20 minutes compared to seven minutes). That said, \texttt{flexBART-UST} is still faster than \texttt{BART} and \texttt{DART}, further underscoring the importance of eliminating redundant computations.

### 4.4 Semi-synthetic network-indexed regression

The Philadelphia crime data example is special case of a more general problem, which we term \textit{network-indexed regression}, in which we observe several covariate-response pairs at each vertex of a fixed network. The network may be a social network, with vertices representing people and edges encoding friendship relationships. Or, as in Section 4.3, the network may represent the spatial adjacency structure of geographical regions. The goal of the network-indexed regression problem is to fit a flexible regression model to the data at each vertex without assuming that the functional relationship between covariates and response is the same at every vertex. Concretely, suppose that at vertex $v$ of a fixed network $\mathcal{G}$, we observe $n_v$ covariate-outcome pairs $(\mathbf{x}_{vt}, y_{vt})$ and that we model $y_{vt} \sim \mathcal{N}(f(\mathbf{x}_{vt}, v), \sigma^2)$. We wish to estimate the function $f(\mathbf{x}, v)$ without imposing strong assumptions about how $f$ varies vertex-to-vertex or about $f$’s functional form.

\texttt{flexBART-UST} is ideally suited to this problem. To illustrate, we conducted a semi-synthetic experiment in which we simulated data at the vertices of the Philadelphia census tract network shown in Figure 7a. Specifically, at each vertex $v$, we generated 100 pairs $(\mathbf{x}_{vt}, y_{vt})$.
by drawing $x_{vt}$ uniformly from the 10-dimensional unit square $[0, 1]^{10}$ and drawing $y_{vt} \sim \mathcal{N}(f(x_{vt}, v), 1)$. The true function $f(x, v)$ was a convex combination of two base functions $f_0(x)$ and $f_1(x)$, where for all $x \in [0, 1]^{10}$ and vertices $v$, we set

$$f(x, v) = w_v f_0(x) + (1 - w_v) f_1(x)$$

$$f_0(x) = 3x_1 + (2 - 5 \times \mathbb{1}(x_2 > 0.5)) \times \sin(\pi x_1) - 2 \times \mathbb{1}(x_2 > 0.5)$$

$$f_1(x) = (3 - 3 \times \cos(6\pi x_1)) \times \mathbb{1}(x_1 > 0.6) - (10\sqrt{x_1}) \times \mathbb{1}(x_1 < 0.25).$$

The vertices in Figure 7a are colored according to their weights, with dark blue corresponding to a weight of zero and dark red according to a weight of one. In this way, for each $x \in [0, 1]^{10}$, the function $f(x, \cdot)$ smoothly interpolates between $f_0(x)$ and $f_1(x)$.

Figure 7: (a) Network representing the adjacency structure of Philadelphia census tracts. Vertices correspond to tracts and edges are drawn between vertices with spatially adjacent tracts. Vertex colors correspond to the weights $w_v$ used to generate the data for the semi-synthetic data described in Section 4.4. (b) Predictive RMSE for each implementation averaged across vertices present (top) and held out during training (bottom).

We trained our models using all the data from 90% of all vertices and compared the abilities of each implementation to predict the value of $f$ evaluated at new $x$ values at (i) vertices seen during training and (ii) held-out vertices. Figure 7b compares each implementation’s predictive RMSEs for both tasks across 100 Monte Carlo replications.
Unsurprisingly, BART, DART, and flexBART-UU, which cannot effectively leverage network adjacency information, are much worse at predicting evaluations $f(x, v)$ at held-out vertices than they are at predicting evaluations at vertices seen during training. In sharp contrast, flexBART-UST is nearly as good at predicting function evaluations at held-out vertices as it is at predicting function evaluations at training vertices. Interestingly, although flexBART-UU is better than BART and DART at predicting $f(x, v)$ at vertices seen during training, it is much worse at predicting $f(x, v)$ at held-out vertices. On further inspection, we found that degradation in performance was due in large part to flexBART-UU attempting to pool data across vertices with very different weights and very different covariate-response relationships.

5 Discussion

Following well-established statistical practice, previous implementations of BART represented categorical predictors using several binary indicators, one for each level of each categorical variable. By adopting such a representation, the original BART prior places zero probability on a large majority of partitions of the categorical levels, severely limiting BART’s representational capacity. Essentially, BART is a priori prevented from including functions that are constant over multiple groups of categorical levels in the regression tree ensemble.

We overcome this limitation using a more general class of regression tree. In our trees, multiple levels of a categorical covariate can be assigned to both the left and right child of decision nodes. BART ensembles built using our new trees can much more flexibly “share statistical strength” across categorical levels. Motivated by a spatial analysis of crime in Philadelphia, we introduced the uniform spanning tree process that, when used in our regression tree prior, can recursively partition a network into connected components. We implemented our new regression tree prior in the flexBART package. On both simulated and real data, we found that our implementation was faster and yielded much more accurate out-of-sample predictions than the implementations provided in the BART package.

There are several potential directions for future work, a few of which we outline below. First, although our analysis of the Philadelphia crime data focused on census tracts, we can seamlessly model variation at multiple spatial resolutions using flexBART. Specifically, we could introduce additional categorical covariates that encode membership in other geographic areas like counties or police precincts and compute the corresponding spatial adjacency matrices. Including these covariates into flexBART is relatively straightforward and we
plan to explore such multi-resolution models more fully in future work.

In our uniform spanning tree process, we deleted a uniformly selected edge from a uniformly sampled spanning tree. This was done to ensure that both the cutset $\mathcal{C}$ and its complement $\mathcal{A} \setminus \mathcal{C}$ induce connected subgraphs in the network that encoded some a priori co-clustering preferences between the levels. We have found that the process can create very small clusters containing only a few vertices (see the bottom panel of Figure 3c for an example). To encourage partitions with more balanced cluster sizes, we could instead delete edges from the spanning tree with probability proportional to the size of the smallest cluster that results when the edge is deleted. We plan to explore other network partitioning processes that still ensure that $\mathcal{C}$ induces a connected subgraph. For instance, we could form $\mathcal{C}$ by randomly selecting an element of $\mathcal{A}$ and then including all vertices in the network that are within a randomly chosen graph distance from the selected element.

Recursive partitioning with our uniform spanning tree process and other alternatives define kernel functions over the vertices of a network. These kernel functions quantify how often two vertices in the network are co-clustered by the recursive partitioning process. Studying these kernels can provide more insight into how precisely our BART prior smoothes over a network. It would further be interesting to see whether these recursive partitioning kernels are related to the Laplacian kernels studied by Smola and Kondor (2003).

Acknowledgements

I would like to thank Prof. Cecilia Balocchi (Edinburgh) for several helpful discussions about flexible spatial modeling of areal crime data that motivated this work. I am deeply grateful to my colleagues Profs. Keith Levin and Jun Zhu at Wisconsin for many productive conversations about networks, spanning trees, and spatial clustering. Thanks are also due to Susan Glenn, Ajinkya Kokandakar, and Kelvui Yao, who provided immensely helpful feedback on early versions of the flexBART package.

This research was performed using the computing resources and assistance of the UW–Madison Center For High Throughput Computing (CHTC) in the Department of Computer Sciences. The CHTC is supported by UW–Madison, the Advanced Computing Initiative, the Wisconsin Alumni Research Foundation, the Wisconsin Institutes for Discovery, and the National Science Foundation, and is an active member of the OSG Consortium, which is supported by the National Science Foundation and the U.S. Department of Energy’s Office
Support was also provided by the University of Wisconsin–Madison, Office of the Vice Chancellor for Research and Graduate Education with funding from the Wisconsin Alumni Research Foundation.

References

Balocchi, C., Deshpande, S. K., George, E. I., and Jensen, S. T. (2022). Crime in Philadelphia: Bayesian clustering with particle optimization. arXiv:1912.00111.

Balocchi, C., George, E. I., and Jensen, S. T. (2021). Clustering areal units at multiple levels of resolution to model crime incidence in Philadelphia. arXiv:2112.02059.

Balocchi, C. and Jensen, S. T. (2019). Spatial modeling of trends in crime over time in Philadelphia. *Annals of Applied Statistics*, 13(4):2235–2259.

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

Burbidge, J. B., Magee, L., and Robb, A. L. (1988). Alternative transformations to handle extreme values of the dependent variable. *Journal of the American Statistical Association*, 83(401):123–127.

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. *Annals of Applied Statistics*, 4(1):266–298.

Deshpande, S. K. and Wyner, A. J. (2017). A hierarchical Bayesian model of pitch framing. *Journal of Quantitative Analysis in Sports*, 13(3):95–112.

Dorie, V., Hill, J., Shalit, U., Scott, M., and Cervone, D. (2019). Automated versus do-it-yourself methods for causal inference: Lessons learned from a data analysis competition. *Statistical Science*, 34(1):43–68.

Hahn, P. R., Murray, J. S., and Carvalho, C. M. (2020). Bayesian regression models for causal inference: Regularization, confounding, and heterogeneous effects. *Bayesian Analysis*, 15(3):965–1056.
Hill, J. L. (2011). Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics*, 20(1):217–240.

Jacoby, S. F., Dong, B., Beard, J. H., Wiebe, D. J., and Morrison, C. N. (2018). The enduring impact of historical and structural racism on urban violence in Philadelphia. *Social Science & Medicine*, 199:87–95.

Li, F. and Sang, H. (2019). Spatial homogeneity pursuit of regression coefficients for large datasets. *Journal of the American Statistical Association*, 114(527):1050–1062.

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

Luo, Z. T., Sang, H., and Mallick, B. (2021a). Bast: Bayesian additive regression spanning trees for complex constrained domain. In *Advances in Neural Information Processing Systems*.

Luo, Z. T., Sang, H., and Mallick, B. (2021b). A Bayesian contiguous partitioning method for learning clustered latent variables. *Journal of Machine Learning Research*, 22(37):1–52.

Luo, Z. T., Sang, H., and Mallick, B. (2022). BAMDT: Bayesian additive semi-multivariate decision trees for nonparametric regression. In *Proceedings of the 39th International Conference on Machine Learning*.

Mohammadi, R., Pratola, M., and Kaptein, M. (2020). Continuous-time birth-death MCMC for Bayesian regression tree models. *Journal of Machine Learning Research*, 21:1–26.

Petti, B. and Gilani, S. (2022). *baseballr: Acquiring and analyzing baseball data*.

Pratola, M. T. (2016). Efficient Metropolis-Hastings proposal mechanisms for Bayesian regression tree models. *Bayesian Analysis*, 11(3):885–911.

Rhynhart, R. (2020). Mappling the legacy of structural racism in Philadelphia. Report, City of Philadelphia, Office of the Controller.

Smola, A. J. and Kondor, R. I. (2003). Kernels and regularization on graphs. In *Learning theory and kernel machines*, pages 144–158. Springer.

Sparapani, R., Spanbauer, C., and McCulloch, R. (2021). Nonparametric machine learning and efficient computation with Bayesian additive regression trees: The BART r package. *Journal of Statistical Software*, 97(1):1–66.
Stanley, R. P. (1997). *Enumerative Combinatorics*, volume 1. Cambridge University Press.

Teixeira, L. V., Assunção, R. M., and Loschi, R. H. (2015). A generative spatial clustering model for random data through spanning trees. In *2015 IEEE International Conference on Data Mining*.

Teixeira, L. V., Assunção, R. M., and Loschi, R. H. (2019). Bayesian space-time partitioning by sampling and pruning spanning trees. *Journal of Machine Learning Research*, 20(85):1–35.

Wilson, D. B. (1996). Generating random spanning trees more quickly than the cover time. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing*. 
