Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

bartcs: Bayesian Additive Regression Trees for Confounder Selection in R

Yeonghoon Yoo
SungKyunKwan University

Chanmin Kim
SungKyunKwan University

Abstract

This article presents an overview of the **bartcs** R package, which employs a Bayesian additive regression trees-based method for selecting confounders. It uses a Dirichlet distribution as a common variable selection probability prior, updating both the exposure and outcome models simultaneously while fitting tree priors for each. This data-driven method determines which variables (i.e., confounders) affect both models by assigning more posterior weight to them. It supports continuous and binary exposure variables, as well as continuous outcome variables, and is written in C++ for improved computational speed. Additionally, it can take advantage of multiple threads for parallel computing if OpenMP is available on the platform.

Keywords: Bayesian nonparametric, causal inference, high-dimensional confounders, continuous outcome.

1. Introduction

In observational studies, drawing causality always relies on the ignorability assumption (Rosenbaum and Rubin 1983) that all confounders are included in the adjustment procedure. A confounder or confounding variable is a common cause that simultaneously affects both exposure and outcome (Figure 1 (a)). Two groups with different exposure levels, distinguished by the distribution of the confounding variable, also experience its impact on their respective outcome values. Therefore, to estimate the causal relationship between exposure and outcome, it is crucial to select this common cause in the data and adjust for it. In many recent applications, the number of potential confounders is often enormous, making it difficult to select the optimal set of true confounders among them. In this context, the optimal set is a confounder set with an appropriate level of uncertainty that reduces bias in estimating the final causal effect.

The main distinction between confounder selection and the traditional variable selection method is that variables that meet the ignorability assumption should be chosen. Several criteria need to be met by the selected confounders in order to reduce the bias of estimated causal effects. Among them, "disjunctive cause criterion"(VanderWeele 2019) requires that the chosen variables be related to exposure and/or outcome. In Figure 1 (a), a confounder set $\boldsymbol{X}$ that satisfies the disjunctive cause criterion consists of variables that either affect exposure A, affect outcome Y, or simultaneously affect both A and Y. A better condition than this is "disjunctive cause criterion without instruments"(VanderWeele 2019), which removes the variables related to exposure but not directly associated with outcome. An instrument, or instrumental variable, is a variable that influences exposure A but does not affect outcome Y. It is known to amplify bias in causal effect estimation when there is an unmeasured confounder (Myers, Rassen, Gagne, Huybrechts, Schneeweiss, Rothman, Joffe, and Glynn 2011). In Figure 1 (b), if a certain confounder from $\boldsymbol{X}$ is unmeasured and not adjusted for (i.e., in the presence of an unmeasured confounder), conducting adjustment for instrument Z leads to additional bias, known as "Z-bias" (Ding, VanderWeele, and Robins 2017). Therefore, the best practice is to remove this instrument during the covariate adjustment process. However, manually identifying a set of confounders that meet these criteria among a large number of potential confounders is challenging.

Methods based on data and statistical models for performing such confounder selection have recently been proposed. One such method is the Bayesian adjustment for confounding (BAC) method proposed by Wang, Parmigiani, and Dominici (2012); Lefebvre, Delaney, and McClelland (2014), which connects exposure and outcome models through common variable inclusion indicator variables to identify confounders. Wang, Dominici, Parmigiani, and Zigler (2015) later modified the BAC method to work with generalized linear outcome models. Wilson and Reich (2014) suggested a method based on decision theory with a similar goal, which performs well for a variety of sample sizes. In terms of selecting relevant covariates for use in propensity score, Shortreed and Ertefaie (2017) proposed the outcome-adaptive LASSO method. In addition, Häggström (2018) proposed a method for identifying the causal structure and estimating the causal effect using a probability graphical model.

Despite the advantages of the previously mentioned methods, they each have limitations. To address these shortcomings, Kim, Tec, and Zigler (2023) proposed a novel Bayesian nonparametric model that aims to overcome these limitations. They suggested a new method that employs Bayesian additive regression trees (BART; Chipman, George, McCulloch *et al.* (2010)) with a shared prior for the selection probabilities, which links the exposure and outcome models. This approach allows for the flexibility and precision of a Bayesian nonparametric model, while also identifying and integrating covariates that are related to both the exposure and outcome into the final estimator. This paper introduces **bartcs**, a new R (R Core Team 2021) package developed by Yoo (2023) that implements the Bayesian additive regression trees method for confounder selection proposed by Kim *et al.* (2023). The package, which is written in C++ and integrated into R via Rcpp for fast computation and easy use, can be downloaded from the Comprehensive R Archive Network (CRAN) at https://cran.r-project.org/package=bartcs. Certain sections of the code referred to the **BART** package by Sparapani, Spanbauer, and McCulloch (2021) under the GPL license, with modifications. In particular, the development of efficient code involved referencing the existing **BART** package algorithm in following aspects: 1) Code related to obtaining residuals in the Bayesian backfitting process; 2) Code dedicated to efficiently searching for variables eligible for splitting

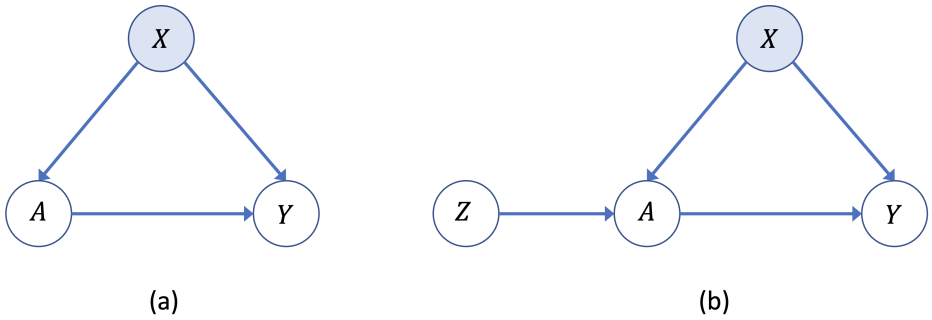


Figure 1: Directed Acyclic Graphs (DAGs): (a) the relationship between exposure A and outcome Y is confounded by covariates X; (b) Adjusting for instrument Z, which affects exposure A but is unrelated to outcome Y, may introduce additional bias if there is an unmeasured covariate in X.

when proposing a splitting variable during the tree alteration process; 3) Code for calculating the μ parameter value of leaf nodes; 4) Code for obtaining sufficient statistics for all bottom nodes.

In this paper, we provide an overview of the package, including installation instructions, usage examples, and a demonstration of its performance on simulated data. We also include a comparison with other existing confounder selection methods. Our aim is to provide researchers with a useful tool for identifying relevant confounders in their causal inference studies and to enable them to make more accurate causal inferences.

2. Overview of model

We first express causal estimation within a potential outcome framework (Rubin 1974). For each unit $i = 1, \cdots, N$, the potential outcome for the i-th unit is defined as $Y_i(a)$, representing the potential value of the outcome Y_i that could be observed under the binary exposure $A_i = a \in \{0, 1\}$. The target causal estimand is

$$\Delta(1, 0) = E[Y_i(1) - Y_i(0)],$$

which represents the average difference between two potential outcomes under two exposure levels 0 and 1. In the later section, we will also explain the utilization of the proposed model by extending it for cases involving continuous exposure.

However, unlike randomized trials, the exposure assignment is not randomized in observational studies, making it impossible to directly identify either $E[Y_i(1)]$ or $E[Y_i(0)]$ from observed data. With no unmeasured confounders $\boldsymbol{X}_i$, the following strong ignorable treatment assignment assumption (Rosenbaum and Rubin 1983) holds

$$\{Y_i(1), Y_i(0)\} \perp A_i | \boldsymbol{X}_i,$$

and $0 < Pr(A_i = 1 | \boldsymbol{X}_i = \boldsymbol{x}) < 1$ for all $\boldsymbol{x}$; $i = 1, \cdots, N$. The first part is also known as the unconfoundedness assumption, and the second part is referred to as the positivity or overlap assumption, which states that each unit has a non-zero probability of being assigned to each

Package	Prog. Lang.	Description
bacr (Wang *et al.* 2015)	R	Assume (generalized-) linear models (i.e., *parametric* models) for exposure and outcome. Supports binomial, Poisson, Gaussian exposure and outcome.
BayesPen (Wilson, Bondell, and Reich 2015)	R	Assume linear models (i.e., *parametric* models) for exposure and outcome. Support continuous outcome.
CovSelHigh (Häggström 2017)	R	Confounder selection performed via either Markov/Bayesian networks (Model-free selection of confounders).
BART† (Sparapani *et al.* 2021)	C++	Incorporate the Dirichlet sparse prior of Linero (2018) for variable selection in the BART outcome model. Support various outcome types (categorical, continuous, binary, survival outcome).
bcf ‡ (Hahn, Murray, and Carvalho 2020)	C++	Specify different BART models for confounding adjustment and heterogeneous effect estimation, and regularizing the treatment effect directly. Support continuous outcome.
bartCause‡ (Hill 2011)	C++	Fit exposure and outcome models using the BART algorithm, producing estimates of treatment effects. Support continuous and binary outcome.
bartcs (Yoo 2023)	C++	Use BART outcome and exposure models with the common Dirichlet prior for confounder selection. Support binary and continuous exposure, and continuous outcome.

Table 1: Summary of different confounder selection methods. †Note that this model (BART with a Dirichlet sprase prior (DART)) does not primarily focus on confounder selection, but rather variable selection, and this variable selection functionality is enabled by setting `sparse=TRUE` in `wbart` (continuous outcome) `pbart/lbart` (binary outcome) `mbart/mbart2` (categorical outcome) `surv.bart` (survival outcome) functions from the **BART** package. ‡ Note that these models lack the ability for both variable selection and confounder selection.

treatment condition. This strong ignorable treatment assignment assumption is sufficient to identify the target causal estimand $\Delta(1,0)$ (Rosenbaum and Rubin 1983; Ding and Li 2018; Li, Ding, and Mealli 2023). In practice, even if the true treatment assignment mechanism satisfies the above conditions, finite observed data may have only one treatment condition value for certain combinations of $\boldsymbol{X}$. In this case, a non-overlap region occurs for that $\boldsymbol{X}$ combination, and target causal estimates in such cases inevitably rely on extrapolation dependent on the model. When non-overlap is severe, it can amplify bias in the target causal estimate. Therefore, recent research interest lies in whether estimates in such regions are provided with an appropriate level of uncertainty (Papadogeorgou and Li 2020; Oganisian and Roy 2020; Li *et al.* 2023). Any method based on outcome regression cannot provide accurate

estimation in the non-overlapping region. Further discussion on this topic is available in Li *et al.* (2023).

Another notable aspect of this assumption is that it is untestable. Therefore, it is not possible to conduct tests based on the data to determine which confounder X satisfies the above assumption. However, confounders $\boldsymbol{X}$ that meet the criteria presented in the introduction (disjunctive cause criterion or disjunctive cause criterion without an instrument; (VanderWeele 2019)) can be considered a minimum basis for a "proper" confounder set.

With this strong ignorable treatment assignment assumption in place, we can represent the causal effect by the following equation of the observable quantities:

$$\Delta(1,0;\boldsymbol{x}) = E[Y_i|A_i = 1, \boldsymbol{X}_i = \mathbf{x}] - E[Y_i|A_i = 0, \boldsymbol{X}_i = \mathbf{x}],$$

and finally identify and estimate the target estimand $\Delta(1,0)$ by averaging over confounders $\boldsymbol{X}$. Thus, the two key tasks in estimating causal effects are identifying the confounders among a potentially large set of covariates, and determining the outcome model (i.e., $E[Y_i|A_i = a, \boldsymbol{X}_i = \boldsymbol{x}]$, $a \in \{0,1\}$) with flexibility and precision. The **bartcs** R package was developed to address these challenges by utilizing Bayesian additive regression trees (BART) models for confounder selection and causal effect estimation.

2.1. Overview of BART

The BART model (Chipman *et al.* 2010) is an ensemble of decision trees that can be represented by the following equation:

$$y_i = \sum_{t=1}^{T} g(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t) + \epsilon_i,$$

where ϵ_i follows a normal distribution with mean 0 and variance σ^2, and $g(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t)$ is a function that maps the tree structure and parameters to the response, for all $i = 1, \cdots, N$. For each of T distinct trees, $\mathcal{T}_t$ represents the structure of the t-th tree and $\mathcal{M}_t = \{\mu_{t,1}, \mu_{t,2}, \cdots, \mu_{t,n_t}\}$ represents its mean parameters at the terminal nodes. Each tree has internal nodes that are split based on a "splitting variable" X_j and "splitting value" c (Figure 2).

In the Markov Chain Monte Carlo (MCMC) update, Bayesian backfitting(Hastie, Tibshirani *et al.* 2000) is utilized within a Metropolis-within-Gibbs sampler. This involves fitting each tree in the ensemble sequentially, using the residual responses: $\mathbf{R}_{-t} := \mathbf{y} - \sum_{j \neq t} g(\boldsymbol{X}; \mathcal{T}_j, \mathcal{M}_j)$ where $\mathbf{R}_{-t}$ denotes unexplained outcome residuals for the t-th tree. In each iteration of the MCMC update, a new tree structure is proposed by randomly selecting one of three possible tree alterations:

GROW: Choose a terminal node at random, and create two new terminal nodes. This process involves randomly selecting a predictor, X_j, and its associated "splitting value," c, to create the two new terminal nodes.

PRUNE: Pick an internal node at random where both children are terminal nodes (known as a "singly internal node" (Kapelner and Bleich 2016)) and remove both of its children (thus making it a terminal node).

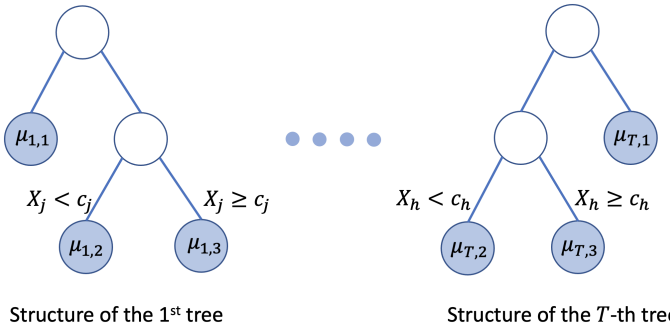


Figure 2: The tree structures consist of T trees, each with nodes represented by circles. Terminal nodes, shown in blue, have μ values. The outcome estimate $\hat{Y}$ of each observation is calculated by adding up the μ values of the terminal nodes where the observation falls within each tree. The method used to split each internal node into two different children nodes is the "splitting rule," which consists of a "splitting variable" (i.e., X_j) and a "splitting value" (i.e., c).

CHANGE: Select an internal node at random and modify its splitting variable and value according to the priors.

Specifically, when using the grow and change alterations, a new covariate is randomly selected from a set of P available covariates as the splitting variable, according to the assumed prior. The original BART model used a uniform prior of $\{1/P, 1/P, \cdots, 1/P\}$ on the selection probabilities $\boldsymbol{s} = (s_1, s_2, \cdots, s_P)$. However, to promote sparsity, Linero (2018) proposed using a Dirichlet prior $(s_1, s_2, \cdots, s_P) \sim \mathcal{D}(\alpha/P, \cdots, \alpha/P)$. This prior specification, as outlined in Table 1, enables the variable selection functionality of the **BART** package. Through this, it can be utilized as a Bayesian variable selection method to choose important predictors in regression problems. Kim *et al.* (2023) have adapted this method for causal inference, proposing a way to select confounders. By specifying a common Dirichlet prior on the selection probabilities of the outcome and exposure models, it allows for the selection of important variables (i.e., confounders) in both models. In the following section, we will explain the specific setting of this method and the steps involved in computing the posterior distributions.

2.2. BART confounder selection

The **bartcs** package in R is designed for selecting confounding variables, particularly when a large number of potential confounding variables are present, and for estimating the average treatment effect (ATE) given the chosen set of confounding variables. To accomplish this, the package uses the Bayesian additive regression trees (BART) model to specify the exposure

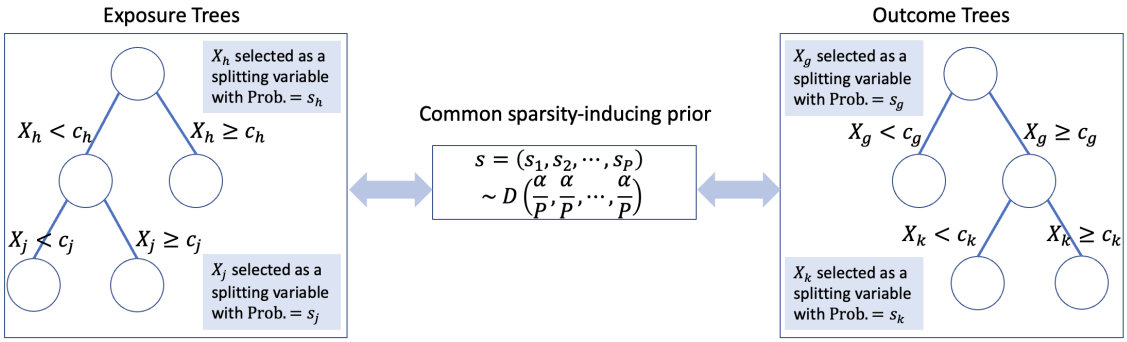


Figure 3: A shared sparsity-inducing prior for the selection probability vector connects the exposure model and outcome model, enabling the selection of the splitting variables in both models. The selection probability vector is updated based on the number of splitting variables used to describe each tree.

and outcome models as follows:

$$P(A_i = 1) = \Phi\left(\sum_{t=1}^{T} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t)\right) \quad (1)$$

$$\underline{\text{Separate Outcome Models}}: \quad Y_i = \sum_{t=1}^{T} g_y^a(\boldsymbol{X}_i; \mathcal{T}_t^a, \mathcal{M}_t^a) + \epsilon_i^a, \quad \epsilon_i^a \sim N(0, \sigma_a^2) \quad (2)$$

$$\underline{\text{Single Outcome Model}}: \quad Y_i = \sum_{t=1}^{T} g_y(A_i, \boldsymbol{X}_i; \mathcal{T}_t', \mathcal{M}_t') + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2), \quad (3)$$

for $i = 1, \cdots, N$ in Equations 1 and 3, and for $i \in \mathcal{I}_a$ where $\mathcal{I}_a$ denotes a set of units under each exposure arm $a \in \{0, 1\}$ in Equation 2. In Equation 1, $\Phi(\cdot)$ is the standard normal cumulative distribution function. Note that it is required to replace Equation 1 with $A_i = \sum_{t=1}^{T} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t) + \epsilon_i, \epsilon_i \sim N(0, \tau^2)$ when considering a continuous exposure (in Section 5). We incorporate a common sparsity-inducing Dirichlet prior $(s_1, s_2, \cdots, s_P) \sim \mathcal{D}(\alpha/P, \cdots, \alpha/P)$ in the exposure model (Equation 1) and the outcome model (Equation 2) resulting in a conjugate update (Figure 3).

If a particular covariate, X_j, is frequently used as a splitting variable in either the model for A or the model for Y, the model will assign more weight to the selection probability s_j through larger numbers of splits on X_j. This means that the selection probabilities will tend to favor covariates that have a relationship with A, Y, or both A and Y. The final confounders chosen for effect estimation in the model for Y will be those that were proposed for splitting through this prior and were accepted during the updating step of the model for Y, which will further prioritize variables that have a relationship with Y. This characteristic satisfies the "disjunctive cause criterion without instruments" in confounder selection.

Separate outcome models

For a binary exposure, we separate the outcome model into two distinct sub-models, in order to align the dimensions of the covariates in both the exposure and outcome models (note that the outcome model includes exposure A as an additional covariate if a single outcome model is

Algorithm 1 Posterior Computation (Separate Outcome Models)

Require: Samples from the previous iteration $(\mathcal{T}_t, \mathcal{T}_t^1, \mathcal{T}_t^0, \mathcal{M}_t, \mathcal{M}_t^1, \mathcal{M}_t^0)$ for $t = 1, \ldots, T$ and (σ_1^2, σ_0^2), and data $(y_i, A_i, \boldsymbol{X}_i)$ for $i = 1, \cdots, N$

for $r = 1, \ldots, M$ iteration **do**

  for $i = 1, \ldots, N$ **do**

$$Z_i \sim \begin{cases} N\left(\sum_{t=1}^T g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t), 1\right) I_{(Z_i > 0)} & \text{for } A_i = 1; \\ N\left(\sum_{t=1}^T g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t), 1\right) I_{(Z_i \leq 0)} & \text{for } A_i = 0 \end{cases}$$ ▷ latent exposure variable

  end for

  for $j = 1, \ldots, T$ **do**

    for $i = 1, \ldots, N$ **do**

      $R_{i,-j}^{(r)} = Z_i - \sum_{t \neq j} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t)$ ▷ residual of the exposure model

      $H_{i,-j}^{1,(r)} = y_i - \sum_{t \neq j} g_y^1(\boldsymbol{X}_i; \mathcal{T}_t^1, \mathcal{M}_t^1)$ ▷ residual of the outcome model for $i \in \mathcal{I}_1$

      $H_{i,-j}^{0,(r)} = y_i - \sum_{t \neq j} g_y^0(\boldsymbol{X}_i; \mathcal{T}_t^0, \mathcal{M}_t^0)$ ▷ residual of the outcome model for $i \in \mathcal{I}_0$

    end for

    $\mathcal{T}_j^{(r)} \sim [\mathcal{T}_j | R_{1,-j}^{(r)}, \cdots, R_{N,-j}^{(r)}, 1]$ ▷ based on one of the three acceptance ratios

    $\mathcal{T}_j^{1,(r)} \sim [\mathcal{T}_j^1 | \boldsymbol{H}_{\cdot,-j}^{1,(r)}, \sigma_1^2]$ ▷ based on one of the three acceptance ratios

    $\mathcal{T}_j^{0,(r)} \sim [\mathcal{T}_j^0 | \boldsymbol{H}_{\cdot,-j}^{0,(r)}, \sigma_0^2]$ ▷ based on one of the three acceptance ratios

    $\mathcal{M}_j^{(r)} \sim [\mathcal{M}_j | \mathcal{T}_j^{(r)}, R_{1,-j}^{(r)}, \cdots, R_{N,-j}^{(r)}, 1]$

    $\mathcal{M}_j^{1,(r)} \sim [\mathcal{M}_j^1 | \mathcal{T}_j^{1,(r)}, \boldsymbol{H}_{\cdot,-j}^{1,(r)}, \sigma_1^2]$

    $\mathcal{M}_j^{0,(r)} \sim [\mathcal{M}_j^0 | \mathcal{T}_j^{0,(r)}, \boldsymbol{H}_{\cdot,-j}^{0,(r)}, \sigma_0^2]$

    where, for each $a \in \{0, 1\}$, $\boldsymbol{H}_{\cdot,-j}^{a,(r)}$ denotes $\{H_{i,-j}^{a,(r)} | i \in \mathcal{I}_a\}$

  end for

$$\sigma_1^2 \sim \text{Inv.Gamma}\left(a_\sigma + \frac{|\mathcal{I}_1|}{2}, b_\sigma + \frac{1}{2}\left\{\sum_{i \in \mathcal{I}_1}\left(y_i - \sum_{t=1}^T g_y^1(\boldsymbol{X}_i; \mathcal{T}_t^1, \mathcal{M}_t^1)\right)\right\}\right)$$

$$\sigma_0^2 \sim \text{Inv.Gamma}\left(a_\sigma + \frac{|\mathcal{I}_0|}{2}, b_\sigma + \frac{1}{2}\left\{\sum_{i \in \mathcal{I}_0}\left(y_i - \sum_{t=1}^T g_y^0(\boldsymbol{X}_i; \mathcal{T}_t^0, \mathcal{M}_t^0)\right)\right\}\right)$$

  Update $\boldsymbol{s}^{(r)}$ via the Gibbs algorithm:

$$\boldsymbol{s}^{(r)} \sim \mathcal{D}\left(n_1^a + n_1^{y_1} + n_1^{y_0} + \alpha/P, \cdots, n_P^a + n_P^{y_1} + n_p^{y_0} + \alpha/P\right)$$

end for

Algorithm 2 Posterior Computation (Single Outcome Model)

Require: Samples from the previous iteration $(\mathcal{T}_t, \mathcal{T}'_t, \mathcal{M}_t, \mathcal{M}'_t)$ for $t = 1, \ldots, T$ and $(\sigma^2, \boldsymbol{s})$, and data $(y_i, A_i, \boldsymbol{X}_i)$ for $i = 1, \cdots, N$

for $r = 1, \ldots, M$ iteration **do**

  for $i = 1, \ldots, N$ **do**

$$Z_i \sim \begin{cases} N\left(\sum_{t=1}^{T} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t), 1\right) I_{(Z_i > 0)} & \text{for } A_i = 1; \\ N\left(\sum_{t=1}^{T} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t), 1\right) I_{(Z_i \leq 0)} & \text{for } A_i = 0 \end{cases}$$ ▷ latent exposure variable

  end for

  for $j = 1, \ldots, T$ **do**

    for $i = 1, \ldots, N$ **do**

      $R_{i,-j}^{(r)} = Z_i - \sum_{t \neq j} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t)$ ▷ residual of the exposure model

      $H_{i,-j}^{(r)} = y_i - \sum_{t \neq j} g_y(\boldsymbol{X}_i; \mathcal{T}'_t, \mathcal{M}'_t)$ ▷ residual of the outcome model

    end for

    $\mathcal{T}_j^{(r)} \sim [\mathcal{T}_j | R_{1,-j}^{(r)}, \cdots, R_{N,-j}^{(r)}, 1]$ ▷ based on one of the three acceptance ratios

    $\mathcal{T}_j'^{,(r)} \sim [\mathcal{T}'_j | H_{1,-j}^{(r)}, \cdots, H_{N,-j}^{(r)}, \sigma^2]$ ▷ based on one of the three acceptance ratios

    $\mathcal{M}_j^{(r)} \sim [\mathcal{M}_j | \mathcal{T}_j^{(r)}, R_{1,-j}^{(r)}, \cdots, R_{N,-j}^{(r)}, 1]$

    $\mathcal{M}_j'^{,(r)} \sim [\mathcal{M}'_j | \mathcal{T}_j'^{,(r)}, H_{1,-j}^{(r)}, \cdots, H_{N,-j}^{(r)}, \sigma^2]$

  end for

$$(\sigma^2)^{(r)} \sim \text{Inv.Gamma}\left(a_\sigma + \frac{N}{2}, b_\sigma + \frac{1}{2}\left\{\sum_{i=1}^{N}\left(y_i - \sum_{t=1}^{T} g_y(\boldsymbol{X}_i; \mathcal{T}_t'^{,(r)}, \mathcal{M}_t'^{,(r)})\right)\right\}\right)$$

  Update $\boldsymbol{s}^{(r)}$ based on the M-H algorithm:

$$\text{Proposal:} \quad \boldsymbol{s}^{(r)} \sim \mathcal{D}\left(n_0^y + c + \alpha/P, n_1^a + n_1^y + \alpha/P, \cdots, n_P^a + n_P^y + \alpha/P\right)$$

$$\text{Acceptance Ratio:} \quad P_{\text{AR}}(\boldsymbol{s} \to \boldsymbol{s}^{(r)}) = \min\left\{1, \left(\frac{1 - \sum_{j=1}^{P} s_j}{1 - \sum_{j=1}^{P} s_j^{(r)}}\right)^{\sum_{j=1}^{J} n_j^y}\right\}$$

end for

specified). For Equations 1 and 2, a sparsity-inducing prior is applied to $(s_1, s_2, \cdots, s_P)$, which is shared among three models: one for exposure and two for outcomes. The resulting update based on this prior is $(s_1, s_2, \cdots, s_P) \sim \mathcal{D}(\alpha/P + n_1^a + n_1^{y_1} + n_1^{y_0}, \cdots, \alpha/P + n_P^a + n_P^{y_1} + n_P^{y_0})$, where $n_j^{y_1}$ and $n_j^{y_0}$ represent the numbers of splits on the confounder X_j in two separate outcome models, and n_j^a represents the number of splits on X_j in the exposure model.

We use "Bayesian backfitting"(Hastie *et al.* 2000) to obtain posterior samples for the exposure and outcome models. For the exposure model, this involves a Metropolis-within-Gibbs sampler, where we fit each tree $\mathcal{T}_t$ iteratively using residual responses :

$$R_{i,-t} = Z_i - \sum_{j \neq t} g_a(\boldsymbol{X}_i; \mathcal{T}_j, \mathcal{M}_j)$$

for $i = 1, \cdots, N$ where Z_i is a latent variable for the binary exposure constructed with

$$Z_i \sim \begin{cases} N\left(\sum_{t=1}^{T} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t), 1\right) I_{(Z_i > 0)} & \text{for } A_i = 1; \\ N\left(\sum_{t=1}^{T} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t), 1\right) I_{(Z_i \leq 0)} & \text{for } A_i = 0. \end{cases}$$

Note that the variance parameter (σ^2) is assigned a value of 1 as a result of the construction of the latent variable. For each tree $\mathcal{T}_t$ for the exposure model, we propose a new tree structure $\mathcal{T}_t$ from the full conditional $[\mathcal{T}_t | R_{1,-t}, \cdots, R_{n,-t}]$ (i.e., grow, prune or change alterations), and update the parameters within the tree through the full conditional $[\mathcal{M}_t | \mathcal{T}_t, R_{1,-t}, \cdots, R_{n,-t}]$.

To draw samples for $\mathcal{M}_t$, we assume a prior $\mu \sim N(\mu_\mu/T, \sigma_\mu^2)$ on each of the leaf parameters $\mathcal{M}_t = \{\mu_1, \mu_2, \cdots, \mu_{t_b}\}$, where t_b is the number of terminal nodes in tree $\mathcal{T}_t$. The range center of latent variable Z_i's is set as the mean, μ_μ, and σ_μ^2 is empirically determined to satisfy $T\mu_\mu - 2\sqrt{T}\sigma_\mu = Z_{\min}$ and $T\mu_\mu + 2\sqrt{T}\sigma_\mu = Z_{\max}$ where $Z_{\min}$ and $Z_{\max}$ represent the minimum and maximum values of Z_i's (Kapelner and Bleich 2016).

We generate a sample μ_η from the posterior distribution for the η-th terminal node in tree $\mathcal{T}_t$ by using the following equation:

$$\mu_\eta \sim N\left(\frac{1}{1/\sigma_\mu^2 + n_\eta/\sigma^2}\left(\frac{\mu_\mu/T}{\sigma_\mu^2} + \frac{\sum_{i \in \mathcal{O}_\eta} R_{i,-t}}{\sigma^2}\right), \left(\frac{1}{\sigma_\mu^2} + \frac{n_\eta}{\sigma^2}\right)^{-1}\right),$$

where $\mathcal{O}_\eta$ and n_η correspond to the observation indices and the number of observations, respectively, for the η-th terminal node. In our implementation, we set the μ_μ value to 0, and consequently, the **bartcs** package is constructed to shift the Y and Z variables to have a mean value of 0.

For separate outcome models, we also perform a backfitting step to draw samples from $P(\mathcal{T}_1^a, \cdots, \mathcal{T}_T^a, \mathcal{M}_1^a, \cdots, \mathcal{M}_T^a, \sigma_a^2 | \boldsymbol{D})$ for each $A = a \in \{0, 1\}$ by computing the residual responses iteratively as follows:

$$H_{i,-t}^a = y_i - \sum_{j \neq t} g_y^a(\boldsymbol{X}_i; \mathcal{T}_j^a, \mathcal{M}_j^a) \quad \text{for } i \in \mathcal{I}_a,$$

where $\mathcal{I}_a$ represents the set of observations corresponding to $A = a \in \{0, 1\}$. Afterwards, a process is undertaken to update each tree based on $[\mathcal{T}_t^a | H_{\cdot,-t}^a, \sigma_a^2]$ and the parameters of its corresponding terminal nodes from $[\mathcal{M}_t^a | \mathcal{T}_t^a, H_{\cdot,-t}, \sigma_a^2]$ for each exposure level $a \in \{0, 1\}$. This process is analogous to the one described earlier for the exposure model.

For each MCMC iteration, once all the tree structures and corresponding parameters have been updated, we proceed to update the variance parameter (σ_a^2 in each outcome model (2)) using the Gibbs sampler. This is achieved by sampling from the inverse gamma distribution given by:

$$\sigma_a^2 \sim \text{Inv.Gamma}\left(a_\sigma + \frac{|\mathcal{I}_a|}{2}, b_\sigma + \frac{1}{2}\left\{\sum_{i\in\mathcal{I}_a}\left(y_i - \sum_{t=1}^{T} g_y^a(\boldsymbol{X}_i; \mathcal{T}_t^a, \mathcal{M}_t^a)\right)\right\}\right),$$

where $a_\sigma = b_\sigma = 3$.

Next, we update the parameter α in the prior distribution of selection probabilities $\boldsymbol{s} \sim \mathcal{D}(\alpha/P, \cdots, \alpha/P)$ based on a prior of the form $\alpha/(\alpha + P) \sim \text{Beta}(a_0, b_0)$, where $a_0 = 0.5$ and $b_0 = 1$ (Linero 2018). The Metropolis-Hastings algorithm is then used to update the parameter. Finally, we update $\boldsymbol{s}$ using a conjugate sampling update as follows: $\boldsymbol{s} \sim \mathcal{D}(\alpha/P + n_1^a + n_1^{y_1} + n_1^{y_0}, \cdots, \alpha/P + n_P^a + n_P^{y_1} + n_P^{y_0})$, where $n_j^{y_1}$ and $n_j^{y_0}$ represent the numbers of splits on the confounder X_j in two separate outcome models, and n_j^a represents the number of splits on X_j in the exposure model.

The posterior computation process for the approach employing the separate outcome models strategy is outlined in Algorithm 1 through pseudocode.

Single outcome model

Using two separate outcome models for two exposure levels, as outlined in Hill (2011) and Hahn *et al.* (2020), can result in biased estimates if there is a lack of common support in confounders. While a single outcome model can be a viable alternative, it can be challenging to apply a shared sparsity-inducing prior to $(s_1, s_2, \cdots, s_P)$ due to differences in covariate dimensions between the exposure and outcome models. Let $\boldsymbol{s} = (s_0, s_1, s_2, \cdots, s_P)$ represent the selection probabilities, with s_0 denoting the probability of exposure A used in the outcome model. To apply this vector to the exposure model, s is transformed to $\boldsymbol{s}' = (s_1/(1-s_0), s_2/(1-s_0), \cdots, s_P/(1-s_0))$. Then, updating $\boldsymbol{s}$ is based on the following equation (likelihood $\times$ prior):

$$Q = \left(\frac{1}{1-s_0}\right)^{\sum_{j=1}^{P} n_j^a} s_0^{n_0^y + \alpha/P - 1} s_1^{n_1^y + n_1^a + \alpha/P - 1} \cdots s_P^{n_P^y + n_P^a + \alpha/P - 1},$$

using the Metropolis-Hastings algorithm. The proposal distribution for $\boldsymbol{s}$ is designed to follow the full conditional in the separate outcome model, $\mathcal{D}(n_0^y + c + \alpha/P, n_1^a + n_1^y + \alpha/P, n_2^a + n_2^y + \alpha/P, \cdots, n_P^a + n_P^y + \alpha/P)$, and a positive value c is added to prevent proposals for infrequent exposure.

All posterior computation steps are identical to the separate outcome models method, except for the difference that there is only one outcome model. Therefore, updates for the trees and parameters of the outcome model are based on one $[\mathcal{T}_t' | \boldsymbol{H}_{\cdot,-t}, \sigma^2]$ and one $[\mathcal{M}_t' | \mathcal{T}_t', \boldsymbol{H}_{\cdot,-t}, \sigma^2]$ for each tree t. Subsequently, sampling for σ^2 is carried out based on the following inverse gamma distribution:

$$\sigma^2 \sim \text{Inv.Gamma}\left(a_\sigma + \frac{N}{2}, b_\sigma + \frac{1}{2}\left\{\sum_{i=1}^{N}\left(Y_i - \sum_{t=1}^{T} g_y(\boldsymbol{X}_i; \mathcal{T}_t', \mathcal{M}_t')\right)\right\}\right),$$

where $a_\sigma = b_\sigma = 3$. The posterior computation process for the approach employing the single outcome model strategy is outlined in Algorithm 2 through pseudocode.

Given the M set of posterior samples for BART parameters, the causal effect estimand $\Delta(1,0)$ can be estimated using either the separate model or the single model. For the separate outcome model, the estimate is obtained by

$$\hat{\Delta}(1,0) = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{M}\sum_{m=1}^{M}\left\{\sum_{t=1}^{T} g_y^{1,(m)}(\boldsymbol{X}_i;\mathcal{T}_t^1,\mathcal{M}_t^1) - \sum_{t=1}^{T} g_y^{0,(m)}(\boldsymbol{X}_i;\mathcal{T}_t^0,\mathcal{M}_t^0)\right\}\right],$$

where $g_y^{a,(m)}$ is the m-th posterior samples for $A = a \in \{0,1\}$. For the single outcome model, the estimate is obtained by

$$\hat{\Delta}(1,0) = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{M}\sum_{m=1}^{M}\left\{\sum_{t=1}^{T} g_y^{(m)}(1,\boldsymbol{X}_i;\mathcal{T}_t,\mathcal{M}_t) - \sum_{t=1}^{T} g_y^{(m)}(0,\boldsymbol{X}_i;\mathcal{T}_t,\mathcal{M}_t)\right\}\right],$$

where $g_y^{(m)}$ is the m-th posterior samples.

3. Simulated example

The **bartcs** R package makes it easy to implement the confounder selection process described in the previous section. It includes two main functions, `separate_bart()` for the separate outcome model and `single_bart()` for the single outcome model. The package not only offers a summary of the estimated causal effects but also includes visualizations of posterior inclusion probabilities and convergence.

bartcs offers multi-threading support through Open Multi-Processing (OpenMP), an API for shared memory parallel programming that manages thread creation, management, and synchronization for efficient data and computation division among different threads. This allows **bartcs** to specify intensive computations as parallel regions, leading to improved computational efficiency through parallel computing.

The package **bartcs** is available under the general public license (GPL $\geq$ 3) from the Comprehensive R Archive Network (CRAN) at https://cran.r-project.org/package=bartcs and can be installed and loaded into the current R session as follows:

```
R> install.packages("bartcs", dependencies=TRUE)
R> library("bartcs")
```

We will showcase the practical usage of the features in the **barcs** package using simulated examples and the Infant Health and Development Program (IHDP) data.

As a simple example of the **bartcs** package, we use a simulated dataset from Scenario 1 in Kim *et al.* (2023) to illustrate its features. The data-generating model incorporates both the nonlinear propensity score and outcome models, and serves to evaluate the ability to detect 5 true confounding variables out of a huge set of possible covariates, along with the precision of the model's estimation. The dataset consists of 300 observations with 100 potential confounders ($X_1 - X_{100}$), each generated from a normal distribution with mean 0 and variance 1. Of the 100 possible confounders, $X_1 - X_5$ are true confounders. The outcome model includes the five true confounders and two additional predictors, X_6 and X_7 as follows:

$$
\begin{aligned}
P(A_i = 1) &= \Phi(0.5 + 0.5h_1(X_{i,1}) + 0.5h_2(X_{i,2}) - 0.5|X_{i,3} - 1| + 1.5X_{i,4}X_{i,5}) \\
Y_i &\sim N(\mu(\boldsymbol{X}_i),\, 0.3^2) \\
\mu(\boldsymbol{X}_i) &= h_1(X_{i,1}) + 1.5h_2(X_{i,2}) - A_i + 2|X_{i,3} + 1| + 2X_{i,4} + \exp(0.5X_{i,5}) \\
&\quad -0.5A_i|X_{i,6}| - A_i|X_{i,7} + 1|
\end{aligned}
$$

where $h_1(x) = (-1)^{I(x<0)}$ and $h_2(x) = (-1)^{I(x\geq 0)}$ for $i = 1, \cdots, 300$. The data was generated with the following code:

```
R> set.seed(42)
R> N <- 300
R> P <- 100
R> cov <- list()
R> for (i in 1:P) {
+    cov[[i]] <- rnorm(N, 0, 1)
+  }
R> X <- do.call(cbind, cov)
R> h1 <- ifelse(X[, 1] < 0, 1, -1)
R> h2 <- ifelse(X[, 2] < 0, -1, 1)
R> prob <- pnorm(0.5 + h1 + h2 - 0.5 * abs(X[, 3] - 1) +
+                  1.5 * X[, 4] * X[, 5])
R> Trt <- rbinom(N, 1, prob)
R> mu1 <- 1 * h1 + 1.5 * h2 - 1 + 2 * abs(X[, 3] + 1) +
+    2 * X[, 4] + exp(0.5 * X[, 5]) -
+    0.5 * 1 * abs(X[, 6]) - 1 * 1 * abs(X[, 7] + 1)
R> mu0 <- 1 * h1 + 1.5 * h2 - 0 + 2 * abs(X[, 3] + 1) +
+    2 * X[, 4] + exp(0.5 * X[, 5]) -
+    0.5 * 0 * abs(X[, 6]) - 1 * 0 * abs(X[, 7] + 1)
R> Y1 <- rnorm(N, mu1, 0.3)
R> Y0 <- rnorm(N, mu0, 0.3)
R> Y <- Trt * Y1 + (1 - Trt) * Y0
```

Examining the standardized mean differences (SMD) of the (potential) confounders generated through the data generating process above, the following observations can be made. The Standardized Mean Differences (SMD) values presented below were computed using the **tableone** R package (Yoshida and Bartel 2022), which can be installed from CRAN.

```
R> library("tableone")
R> Xdata <- as.data.frame(cbind(Trt,X))
R> names(Xdata) <- c("Trt", paste0(rep("X", 100),1:100))
R> Table <- CreateTableOne(vars = paste0(rep("X", 12),1:12), strata = "Trt",
+    data = Xdata, test = FALSE)
R> print(Table, smd = TRUE)
                  Stratified by Trt
                   0            1            SMD
  n                  164          136
```

```
X1 (mean (SD))   0.28 (0.96) -0.39 (0.90)  0.718
X2 (mean (SD))  -0.25 (0.99)  0.24 (0.92)  0.517
X3 (mean (SD))  -0.14 (1.02)  0.03 (0.90)  0.178
X4 (mean (SD))   0.06 (1.08) -0.07 (1.04)  0.118
X5 (mean (SD))  -0.08 (0.86)  0.01 (1.05)  0.091
X6 (mean (SD))  -0.03 (1.06)  0.15 (0.98)  0.177
X7 (mean (SD))  -0.04 (1.03)  0.01 (0.94)  0.050
X8 (mean (SD))  -0.11 (0.99)  0.20 (1.00)  0.312
X9 (mean (SD))   0.07 (1.04)  0.05 (1.02)  0.017
X10 (mean (SD)) -0.04 (1.13) -0.13 (0.96)  0.087
X11 (mean (SD))  0.05 (1.02) -0.12 (0.98)  0.169
X12 (mean (SD))  0.13 (1.01) -0.23 (0.99)  0.363
```

When looking at the results for the first 12 X variables, it is noted that for true confounders X_1 and X_2, SMD values greater than 0.1, indicative of inadequate covariate balance between the groups, are observed. Similar lack of covariate balance between the groups is also noticed for X_3 and X_4. However, due to randomness, differences between the groups are observed for some covariates other than the true confounders. In this simulation scenario, with the partial presence of the signal from some covariates other than true confounders, the goal is to assess the performance of the model under consideration.

With a generated data set, we fit the BART confounder selection model (the separate outcome model) using `separate_bart()`.

```
R> library("bartcs")
R> separate_fit <- separate_bart(
+    Y = Y, trt = Trt, X = X, num_tree = 200, num_chain = 4,
+    num_burn_in = 10000, num_thin = 5, num_post_sample = 2000
+  )
```

The following are the main arguments used in the `separate_bart()` function call:

- `Y` represents a vector of observed outcome values.
- `trt` denotes a vector of exposure(treatment) values, which can be either binary or continuous depending on the function. Binary treatment values need to be either 0 or 1.
- `X` is a data frame of potential confounders.

The following are the remaining settings for the fit: 4 MCMC chains (`num_chain`) with 200 trees (`num_tree`) are used. Each MCMC chain runs 20000 iterations, with 10000 burn-in iterations (`num_burn_in`) and a thinning factor of 5 (`num_thin`). There are other optional arguments available for hyper-parameter settings with the following default values:

- $\alpha = 0.95$ (`alpha`) and $\beta = 2$ (`beta`): these govern the probability that a node at depth d is nonterminal as follows
$$\alpha(1+d)^{-\beta}.$$

- $\nu = 3$ (`nu`) and $q = 0.95$ (`q`): to set a conjugate prior for the variance σ^2 with $\sigma^2 \sim \nu\lambda/\chi^2_\nu$, we use the following equation to determine the values $P(\sigma < \hat{\sigma}) = q$, where $\hat{\sigma}$ represents the residual standard deviation obtained from a linear regression of Y on X.
- $P_{\text{GROW}} = 0.28, P_{\text{PRUNE}} = 0.28, P_{\text{CHANGE}} = 0.44$ (`step_prob = c(0.28, 0.28, 0.44)`): probabilities of three tree alteration steps.
- `dir_alpha` $= 5$: this is an initial value for hyperparameter α in the sparsity inducing Dirichlet prior $\mathcal{D}(\alpha/P, \alpha/P, \cdots, \alpha/P)$.

```
R> separate_fit

`bartcs` fit by `separate_bart()`

          mean       2.5%      97.5%
ATE -2.2851546 -2.6022894 -1.9692134
Y1   0.7195622  0.4663024  0.9833689
Y0   3.0047169  2.8116436  3.1946016
```

The `separate_bart()` returns a S3 `bartcs` object. A `bartcs` object includes the posterior means and 95% credible intervals for the sample average treatment effect (ATE), and the potential outcomes $Y(1)$ and $Y(0)$. It is important to note that the true values for the ATE, $E[Y(1)]$, and $E[Y(0)]$ are -2.55, 0.64, and 3.19 respectively, and the 95% credible intervals produced by the `separate_bart()` function include these values.

For a more in-depth understanding of the output, the `summary()` function can be used. It provides details regarding the treatment values, tree structure, MCMC chain, and outcomes for each of the chains.

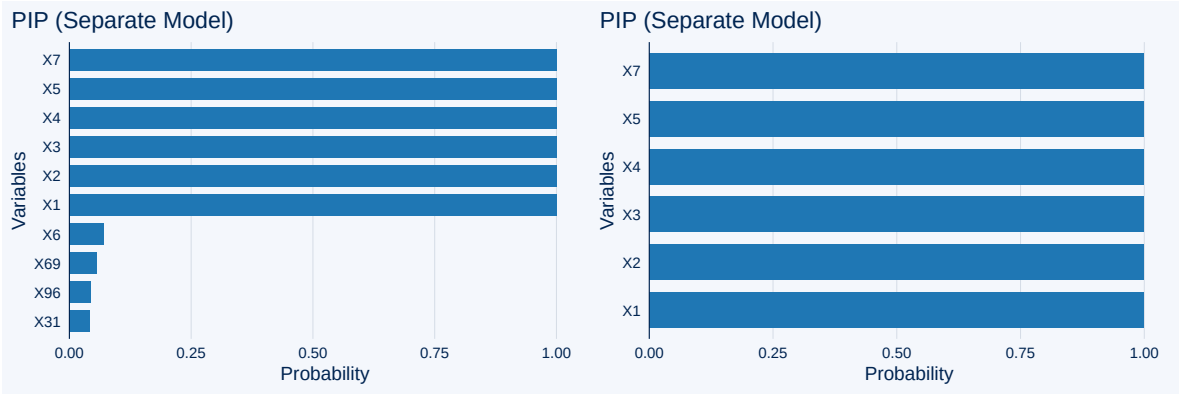
```
R> summary(separate_fit)

`bartcs` fit by `separate_bart()`

Treatment Value
  Treated group    :      1
  Control group    :      0

Tree Parameters
  Number of Tree   :    200              Value  of alpha    :   0.95
  Prob.  of Grow   :   0.28              Value  of beta     :      2
  Prob.  of Prune  :   0.28              Value  of nu       :      3
  Prob.  of Change :   0.44              Value  of q        :   0.95

Chain Parameters
  Number of Chains :      4              Number of burn-in  :  10000
  Number of Iter   :  20000              Number of thinning :      5
  Number of Sample :   2000
```



(a) Plotting PIP with `top_n` argument (b) Plotting PIP with `threshold` argument

Figure 4: Posterior inclusion probability (PIP) plots

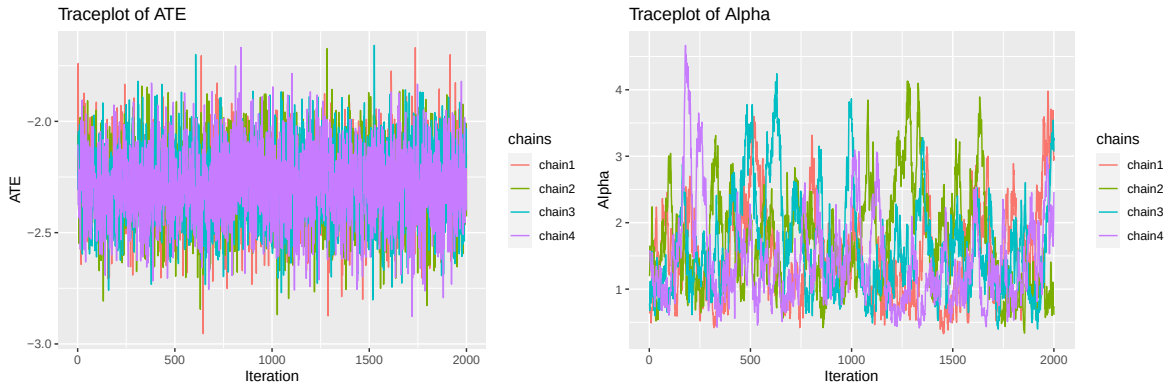
```
Outcome
 estimand chain       2.5%         1Q       mean     median         3Q      97.5%
      ATE     1 -2.6070044 -2.3892357 -2.2830389 -2.2800555 -2.1765359 -1.9757766
      ATE     2 -2.6013548 -2.4017854 -2.2877997 -2.2877071 -2.1798863 -1.9611401
      ATE     3 -2.5961329 -2.3952700 -2.2794876 -2.2793208 -2.1609143 -1.9644475
      ATE     4 -2.6090523 -2.4001084 -2.2902924 -2.2923171 -2.1812900 -1.9761443
      ATE   agg -2.6022894 -2.3965077 -2.2851546 -2.2842764 -2.1748201 -1.9692134
       Y1     1  0.4705203  0.6322748  0.7174467  0.7174147  0.8027479  0.9668359
       Y1     2  0.4707973  0.6305094  0.7223111  0.7213076  0.8153911  0.9851455
       Y1     3  0.4653391  0.6277828  0.7190511  0.7194586  0.8080547  0.9804701
       Y1     4  0.4614500  0.6273396  0.7194400  0.7175295  0.8087480  0.9920899
       Y1   agg  0.4663024  0.6292846  0.7195622  0.7185828  0.8087121  0.9833689
       Y0     1  2.8082437  2.9361088  3.0004857  2.9998629  3.0664869  3.1897135
       Y0     2  2.8189069  2.9442181  3.0101107  3.0107896  3.0778268  3.2013420
       Y0     3  2.8002284  2.9362280  2.9985387  2.9972708  3.0646989  3.1920314
       Y0     4  2.8210957  2.9427012  3.0097324  3.0133450  3.0772579  3.1960383
       Y0   agg  2.8116436  2.9404458  3.0047169  3.0053406  3.0713420  3.1946016
```

For each estimand category, there are five results (rows) that represent the output from each of the 4 MCMC chains and an aggregated output.

For visualization purposes, there are two options available as S3 methods for the `bartcs` object. The first option is the posterior inclusion probability (PIP) plot. PIP is the probability that a variable is used as a splitting variable, and can be interpreted as the importance of a variable. The `inclusion_plot()` function is a wrapper for the `bar_chart()` function from the **ggcharts** package, allowing the use of its arguments to customize the plot. The recommended arguments to use are `top_n` and `threshold`.

```
R> plot(separate_fit, method = "pip", top_n = 10)
R> plot(separate_fit, method = "pip", threshold = 0.5)
```

In Figure 4, the argument `top_n` allows us to select variables with the top `top_n` highest PIPs. The argument `threshold` displays variables with PIP greater than `threshold`. From a



(a) Traceplot of ATE

(b) Traceplot of `dir_alpha` parameter

Figure 5: Traceplots for multiple MCMC chains

decision-theoretical perspective (Barbieri and Berger 2004; Linero 2018), variables with PIPs larger than 0.5 can be considered chosen confounders. It is worth noting that the five true confounders $X_1 - X_5$ are all correctly selected as true confounders with PIPs of 1, along with one extra predictor X_7 in the outcome model.

The second option for visualization is the traceplot, which is mainly used to check MCMC convergence. The function provides a traceplot of the average treatment effect (ATE) for each MCMC chain. Traceplots of other parameters such as `dir_alpha` (the hyperparameter α in the sparsity-inducing Dirichlet prior $\mathcal{D}(\alpha/P, \cdots, \alpha/P)$) and `sigma2_out` (the variance parameter in the outcome model) are also available by using the argument `parameter`.

```
R> plot(separate_fit, method = 'trace')
R> plot(separate_fit, method = 'trace', parameter = 'dir_alpha')
```

In Figure 5, the traceplots of the ATE and `dir_alpha` parameters are shown for four different MCMC chains. Regarding the `dir_alpha` parameter (α), the actual value used as the hyper-parameter for the Dirichlet prior is obtained by dividing the total number of potential confounders, denoted as P (i.e., α/P). Considering the simulation data setting where $P = 100$ is used, the hyper-parameter to be estimated is a significantly small value, which is $\alpha/100$. Therefore, compared to the variation observed in the traceplot, the variation of the actual α/P can be interpreted as considerably smaller. While visual inspection using traceplots is convenient, it is advised to utilize the gelman-rubin diagnostics offered by the `gelman.diag()` function in the **coda** R package (Plummer, Best, Cowles, Vines, Sarkar, Bates, Almond, and Magnusson 2020) for a more thorough evaluation of convergence, as demonstrated in the following section.

We evaluated the performance of **bartcs** in comparison to other models, including those generated by the **bacr** R package (Wang *et al.* 2015) that inspired our model development. The **bacr** package is easily installed via CRAN and loaded into the current R session as follows:

```
R> install.packages("bacr", dependencies=TRUE)
R> library("bacr")
```

To fit the model of this package, we used the `bac()` function where the input data needs to be provided in the form of a data frame. To fit the exposure and outcome models in this case, a generalized linear model is used, and it is necessary to specify the family of the model based on the data type (e.g.. `familyX="binomial"` and `familyY="gaussian"`). The MCMC algorithm was run for 10000 iterations after discarding the first 10000 iterations as burn-ins. Additionally, no interaction between the exposure and each confounder was assumed.

```
R> Z <- as.data.frame(cbind(Y,Trt,X))
R> fit.bac <- bac(
+    data = Z, exposure = "Trt", outcome = "Y",
+    confounders = paste("V", 3:(P + 2), sep = ""),
+    interactors = NULL, familyX = "binomial", familyY = "gaussian",
+    omega = Inf, num_its = 20000, burnM = 10000, burnB = 10000, thin = 5
+  )
```

The result can be checked through the `summary()` function as follows:

```
R> summary(fit.bac)

BAC objects:

Exposure effect estimate:
        posterior mean     95% posterior interval
                  -1.6               (-2.1, -1.3)


Covariates with posterior inclusion probability > 0.5:
    posterior inclusion probability
V3                          1.00000
V4                          1.00000
V5                          1.00000
V6                          1.00000
V7                          1.00000
V99                         0.92100
V14                         0.70305
V54                         0.67480
V90                         0.62345
```

The posterior mean of the ATE was estimated to be -1.6, which was significantly different from the true ATE value of -2.55. Moreover, the 95% credible interval $(-2.1, -1.3)$ did not include the true value. When considering the importance of selected confounders based on the posterior inclusion probability, **bacr** included all important confounders $X_1 - X_5$ (that is, $V3 - V7$ in the summary), but also added X_{12}, X_{52}, X_{88}, and X_{97} (that is, $V14, V54, V90, V99$ in the summary) with high PIPs, which were not true confounders. Notably, X_6 and X_7, which are additional predictors of the outcome model, were not included. This result may be attributed to the fact that **bacr** relies on a parametric model and therefore may struggle to account for the non-linear and complex data structure.

3.1. Connection to coda package

To summarize the results, generic functions such as `summary()` and `plot()` were adapted to work on the `bartcs` objects. Additionally, `mcmc.list` objects were included as components in the `bartcs` object to allow for the use of functions from the **coda** R package (Plummer *et al.* 2020). The `mcmc_list` component of the **bartcs** object can produce summary statistics for each of $E[Y(1)], E[Y(0)], ATE$ using the summary function and generate trace plots and posterior densities for parameters using the plot function. Figure 6 displays plot of `mcmc_list` based on **coda** package.

```
R> summary(separate_fit$mcmc_list)

Iterations = 10005:20000
Thinning interval = 5
Number of chains = 4
Sample size per chain = 2000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

                 Mean        SD  Naive SE Time-series SE
ATE         -2.285155 0.1639173 1.833e-03      2.489e-03
Y1           0.719562 0.1328857 1.486e-03      2.122e-03
Y0           3.004717 0.0977850 1.093e-03      1.790e-03
dir_alpha    1.576421 0.7317213 8.181e-03      6.836e-02
sigma2_out1  0.001731 0.0003359 3.756e-06      5.533e-06
sigma2_out0  0.001310 0.0002334 2.609e-06      3.784e-06

2. Quantiles for each variable:

                2.5%       25%       50%       75%     97.5%
ATE         -2.6022894 -2.396508 -2.284276 -2.174820 -1.969213
Y1           0.4663024  0.629285  0.718583  0.808712  0.983369
Y0           2.8116436  2.940446  3.005341  3.071342  3.194602
dir_alpha    0.5772667  1.024000  1.423480  2.003651  3.390765
sigma2_out1  0.0011718  0.001490  0.001698  0.001935  0.002483
sigma2_out0  0.0009263  0.001146  0.001289  0.001446  0.001845

R> plot(separate_fit$mcmc_list)
```

The convergence of the MCMC object can be assessed by utilizing the convergence diagnostics offered by the **coda** package. To examine the convergence of six parameters. we can employ the `gelman.diag()` function on the `mcmc.list` object, specifically on `separate_fit$mcmc_list`.

```
R> library("coda")
R> gelman.diag(separate_fit$mcmc_list)
```

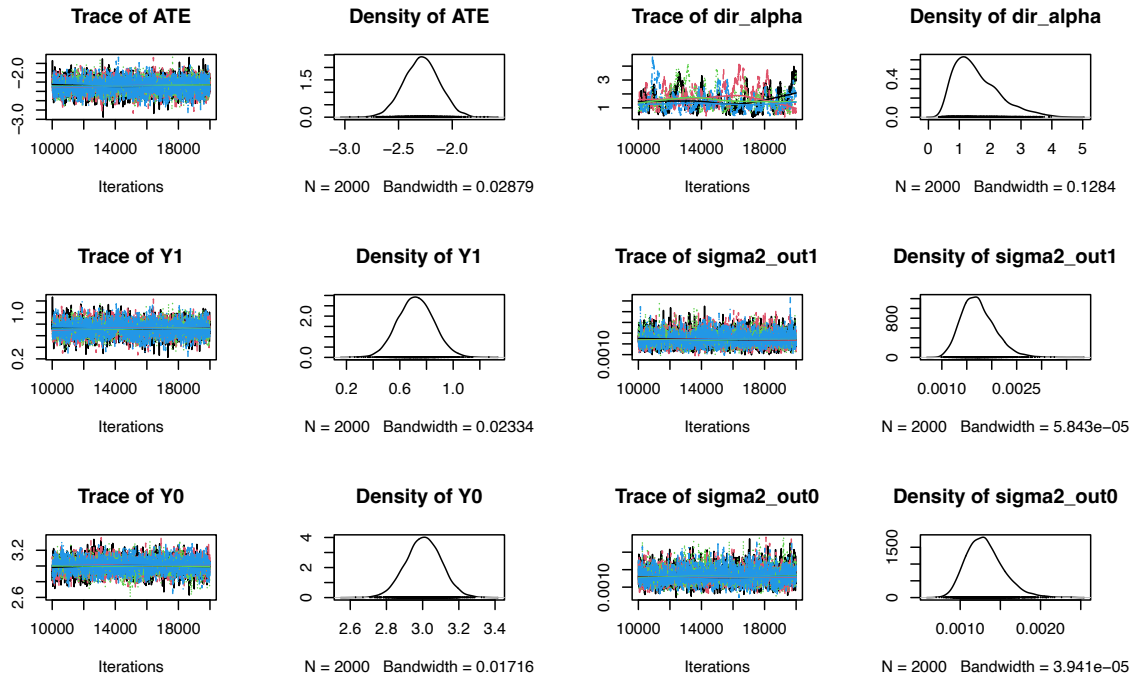


Figure 6: Plot of `mcmc_list` using the **coda** R package

```
Potential scale reduction factors:

            Point est. Upper C.I.
ATE               1.00       1.00
Y1                1.00       1.00
Y0                1.00       1.01
dir_alpha         1.02       1.07
sigma2_out1       1.00       1.00
sigma2_out0       1.00       1.00

Multivariate psrf

1.02
```

Based on the convergence diagnostics, it can be concluded that there are no issues with the convergence of the MCMC chain, similar to the visual inspection.

4. Real data example

In the previous section, the `separate_bart()` function was used to demonstrate a separate outcome model scheme. In this section, a single outcome model is tested using the `single_bart()` function, based on the Infant Health and Development Program (IHDP) dataset as an example. This dataset was collected from a longitudinal study that tracked the

development of low-birth-weight premature infants. The study participants in the treatment group received intensive care and home visits from trained providers and their cognitive test scores were evaluated at the end of the intervention period. The dataset includes a variety of pretreatment variables, including 6 continuous and 19 binary covariates. The original IHDP data is generated from a randomized experiment setting. However, the IHDP data used by Hill (2011) and Louizos, Shalit, Mooij, Sontag, Zemel, and Welling (2017) was manipulated to induce covariate imbalance between treatment groups by removing a subset of the treated group. Specifically, all children with nonwhite mothers were removed from the treated group. We utilize a synthesized variant of the IHDP data as presented in Louizos *et al.* (2017). This version was created employing the **NPCI** package (Dorie 2016) to ascertain the true counterfactual values. As seen in Figure 7, the data generated in this manner significantly violates the overlap assumption for estimating the Average Treatment Effect (ATE). This figure depicts the degree of overlap between two groups (Treated vs Control) for selected covariates. Red crosses represent the control group, and blue triangles represent the treated group. In certain intervals of extreme values for each covariate, there are regions where only control group data exists, or very few data points from the treated group are present. For example, in the interval where the X_5 covariate is less than -4, there is no data from the treated group. Non-overlap occurs in these regions. In the case of the binary covariate X_{18}, there is only one data point from the treated group at the value of 0. Therefore, technically speaking, situations like non-overlap can occur in the estimation process. In such a scenario, one of the objectives is to investigate whether a single outcome model can properly estimate the true ATE.

This data can be loaded by

```
R> data("ihdp", package = "bartcs")
```

and Table 2 displays the summary statistics of the variables. In the dataset, `y_factual` is the observed outcome Y (i.e., $Y(A)$) and `y_cfactual` is the counterfactual outcome Y (i.e., $Y(1-A)$).

We fit the single outcome model using the `single_bart()` function.

```
R> single_fit <- single_bart(
  Y               = ihdp$y_factual,
  trt             = ihdp$treatment,
  X               = ihdp[, 6:30],
  num_tree        = 50,
  num_chain       = 4,
  num_post_sample = 2000,
  num_thin        = 5,
  num_burn_in     = 10000
)
R> single_fit

`bartcs` fit by `single_bart()`

        mean     2.5%    97.5%
ATE 3.964842 3.747028 4.180764
```

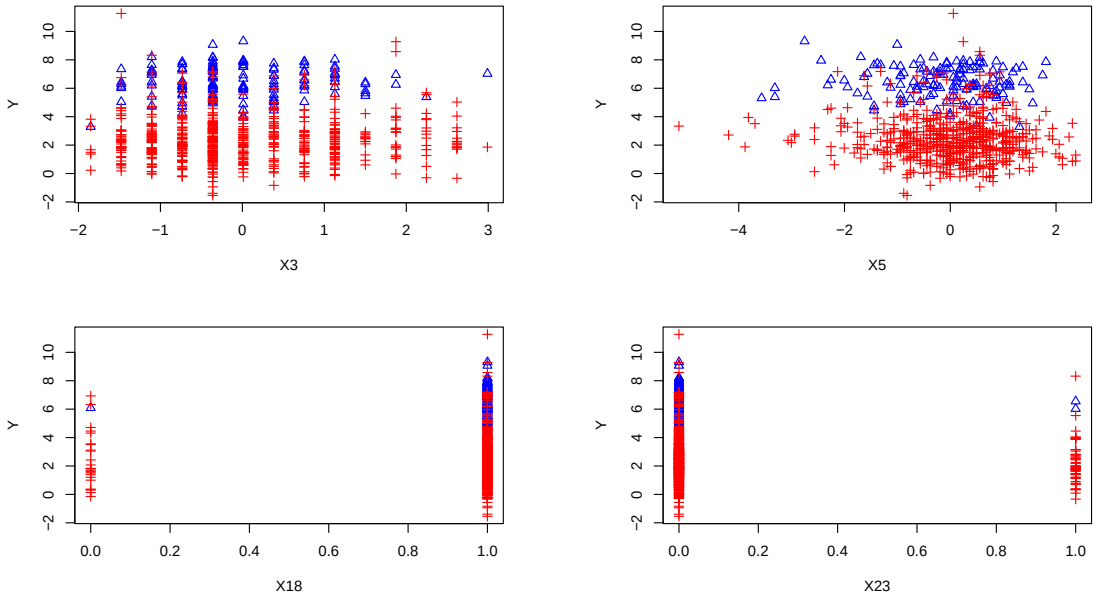


Figure 7: A plot illustrating the degree of overlap between two groups (Treated vs Control) for selected covariates. Red crosses represent the control group, and blue triangles represent the treated group. In certain intervals for each covariate, there are regions where only control group data exists, or very few data points from the treated group are present. Non-overlap occurs in these regions.

	`Treatment` = 1 (n=139)		`Treatment` = 0 (n=608)	
Variable	Mean	IQR	Mean	IQR
Y	6.43	(5.83, 7.34)	2.41	(1.45, 3.08)
$X_1^\star$	0.21	(-0.40, 0.95)	-0.05	(-0.75, 0.79)
$X_2^\star$	0.18	(-0.20, 0.60)	-0.04	(-0.60, 0.60)
$X_3^\star$	-0.04	(-0.73, 0.38)	0.01	(-0.73, 0.76)
$X_4^\star$	-0.22	(-0.88, 0.16)	0.05	(-0.88, 0.16)
$X_5^\star$	-0.14	(-0.69, 0.56)	0.03	(-0.50, 0.68)
$X_6^\star$	0.21	(-0.53, 0.96)	-0.05	(-0.86, 0.63)
X_7	0.52	(0.00, 1.00)	0.51	(0.00, 1.00)
X_8	0.09	(0.00, 0.00)	0.1	(0.00, 0.00)
X_9	0.68	(0.00, 1.00)	0.49	(0.00, 1.00)
X_{10}	0.29	(0.00, 1.00)	0.38	(0.00, 1.00)
X_{11}	0.25	(0.00, 0.50)	0.27	(0.00, 1.00)
X_{12}	0.22	(0.00, 0.00)	0.22	(0.00, 0.00)
X_{13}	0.38	(0.00, 1.00)	0.35	(0.00, 1.00)
X_{14}	1.58	(1.00, 2.00)	1.44	(1.00, 2.00)
X_{15}	0.14	(0.00, 0.00)	0.14	(0.00, 0.00)
X_{16}	0.94	(1.00, 1.00)	0.97	(1.00, 1.00)
X_{17}	0.69	(0.00, 1.00)	0.57	(0.00, 1.00)
X_{18}	0.99	(1.00, 1.00)	0.96	(1.00, 1.00)
X_{19}	0.15	(0.00, 0.00)	0.13	(0.00, 0.00)
X_{20}	0.06	(0.00, 0.00)	0.15	(0.00, 0.00)
X_{21}	0.17	(0.00, 0.00)	0.15	(0.00, 0.00)
X_{22}	0.04	(0.00, 0.00)	0.09	(0.00, 0.00)
X_{23}	0.01	(0.00, 0.00)	0.09	(0.00, 0.00)
X_{24}	0.06	(0.00, 0.00)	0.14	(0.00, 0.00)
X_{25}	0.27	(0.00, 1.00)	0.13	(0.00, 0.00)

Table 2: Summary statistics for the IHDP data set. $\star$ denotes a continuous potential confounder.

```
Y1  6.382810 6.188199 6.581852
Y0  2.417969 2.338264 2.496962
```

The function `single_bart()` returns a **bartcs** object, which displays the posterior means and 95% credible intervals for the average treatment effect (ATE), and the potential outcomes $Y(1)$ and $Y(0)$. The `summary()` and `plot()` functions can also be used with this **bartcs** object generated by `single_bart()`.

```
R> summary(single_fit)

`bartcs` fit by `single_bart()`

Treatment Value
  Treated group    :      1
  Control group    :      0
```

```
Tree Parameters
  Number of Tree   :     50              Value  of alpha    :   0.95
  Prob.  of Grow   :   0.28              Value  of beta     :      2
  Prob.  of Prune  :   0.28              Value  of nu       :      3
  Prob.  of Change :   0.44              Value  of q        :   0.95

Chain Parameters
  Number of Chains :      4              Number of burn-in  :  10000
  Number of Iter   :  20000              Number of thinning :      5
  Number of Sample :   2000

Outcome
 estimand chain     2.5%       1Q     mean   median       3Q    97.5%
      ATE     1 3.758373 3.894465 3.969119 3.968867 4.042380 4.183131
      ATE     2 3.744731 3.886575 3.957434 3.956101 4.026455 4.165961
      ATE     3 3.760480 3.905973 3.980315 3.980162 4.054086 4.206488
      ATE     4 3.730287 3.879606 3.952498 3.953050 4.028315 4.158430
      ATE   agg 3.747028 3.891543 3.964842 3.965384 4.038288 4.180764
       Y1     1 6.196530 6.318675 6.387760 6.387443 6.453303 6.589611
       Y1     2 6.181788 6.310026 6.376027 6.376233 6.439727 6.573960
       Y1     3 6.196317 6.329945 6.396885 6.397153 6.464297 6.601299
       Y1     4 6.169429 6.303404 6.370570 6.371514 6.435679 6.562172
       Y1   agg 6.188199 6.314489 6.382810 6.382215 6.449542 6.581852
       Y0     1 2.339020 2.391137 2.418640 2.418824 2.446414 2.498677
       Y0     2 2.336131 2.392407 2.418593 2.418124 2.446167 2.495229
       Y0     3 2.337997 2.388738 2.416570 2.416414 2.444457 2.495583
       Y0     4 2.340288 2.389536 2.418073 2.418218 2.446018 2.497264
       Y0   agg 2.338264 2.390199 2.417969 2.418042 2.445718 2.496962
```

We also fitted a separate outcome model to the `ihdp` data and compared the results from the single outcome model.

```
R> separate_fit <- separate_bart(
+    Y               = ihdp$y_factual,
+    trt             = ihdp$treatment,
+    X               = ihdp[, 6:30],
+    num_tree        = 50,
+    num_chain       = 4,
+    num_post_sample = 2000,
+    num_thin        = 5,
+    num_burn_in     = 10000
+  )
R> separate_fit

`bartcs` fit by `separate_bart()`
```

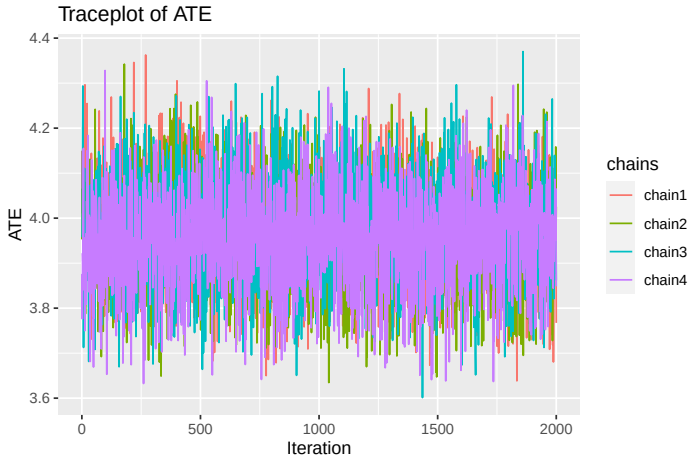


Figure 8: Traceplot of ATE for IHDP dataset

```
        mean     2.5%    97.5%
ATE 3.924013 3.702316 4.148937
Y1  6.342504 6.134043 6.550242
Y0  2.418491 2.340920 2.497081
```

Similar to the separate outcome model, in `single_bart()`, the `plot()` function for the **bartcs** object can also be employed to check the convergence of the MCMC chain. The traceplots for the ATE is presented in Figure 8 with the following line.

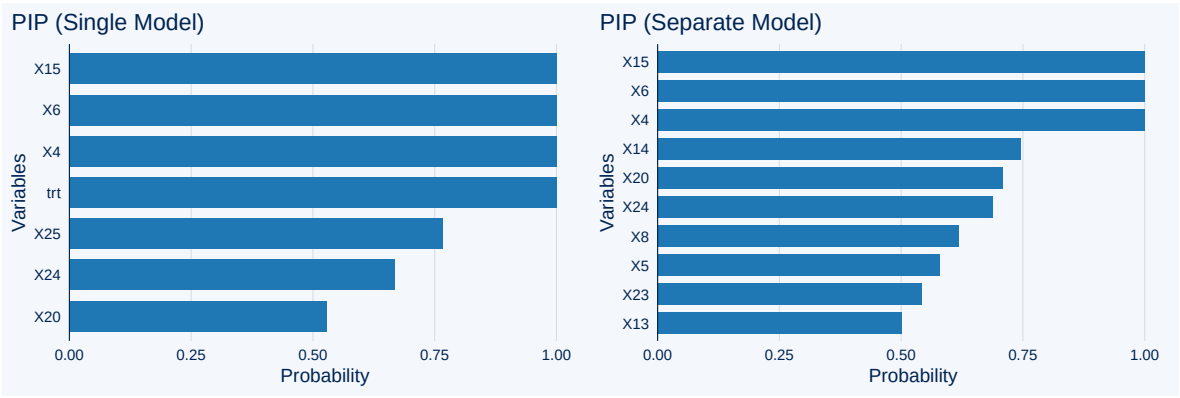
```
R> plot(single_fit, method = 'trace')
```

As this is a simulated version of the IHDP data, the true values are known and are 4.02 for the average treatment effect (ATE), 6.45 for $E[Y(1)]$, and 2.43 for $E[Y(0)]$. The outputs from the two models accurately reflect these true values within their 95% credible intervals. Additionally, the PIP plots (Figure 9) depict chosen confounders with PIP values larger than 0.5.

The important aspect here is that in the case of the single outcome model, the exposure variable (`trt`) is also incorporated into the selection process. As indicated in Equation 2, because the exposure variable is included as one of the covariates in the outcome model, it is subject to variable selection. This means that in the computation of PIP, it is treated similarly to other confounders, producing the following plot (a) in Figure 9. In Figure 9, plot (a) displays the potential confounders for the single outcome model, which have a posterior inclusion probability of 0.5 or more, while plot (b) illustrates the confounders with a posterior inclusion probability of 0.5 or more when the separate outcome model is used. It is noteworthy that X_4, X_6, and X_{15} were consistently chosen as confounders with posterior inclusion probability 1.

4.1. Computation speed

In Figure 10, the computational speed of two models, the separate and single models, is depicted for two different settings of the number of trees (100 vs. 200) based on the scenario in



(a) PIP plot of single model

(b) PIP plot of separate model

Figure 9: PIP plot for IHDP dataset

Section 3. The speed was assessed using 20000 MCMC iterations across various combinations of N and P. We considered three values of N (100, 500, and 1000) and three values of P (circle for $N \times 0.3$, triangle for $N \times 0.5$, and cross for $N \times 1$).

For 100 BART trees, the separate model required 34 to 393 seconds (70 to 761 seconds for 200 BART trees) for computation, while the single model took 30 to 343 seconds (58 to 670 seconds for 200 BART trees), depending on the (N, P) combination. Both models exhibited similar computational speeds overall, considering the MCMC iterations. However, the single model, which fits two BART models (exposure and one outcome model), was found to be more efficient with slightly smaller biases and mean square errors (MSEs) across various scenarios (Kim *et al.* 2023). Therefore, it is recommended to utilize the single model (`single_bart()` function), especially when N is large, due to its faster computational speed.

Additionally, depending on the number of trees used, a significant improvement in computation speed can be observed. It is generally suggested to start with 50 trees as a "good starting value," (Kapelner and Bleich 2016) so using a smaller number of trees is also advised to gain computational advantages in terms of speed. Details about the computer used to obtain the results are provided in the final computational details section.

5. Continuous exposure example

When it comes to a continuous exposure variable, the formula in Equation 1 is changed as follows:

$$A_i = \sum_{t=1}^{T} g_a(\boldsymbol{X}_i; \mathcal{T}_t, \mathcal{M}_t) + \epsilon_i, \epsilon_j \sim N(0, \tau^2).$$

This altered formula is used in conjunction with the single outcome model to perform confounder selection. However, the separate outcome model, which fits two distinct outcome models based on the two exposure levels, is not suitable for the continuous exposure variable. The **bartcs** has an advantage in handling continuous exposure through its `single_bart()` function. This function has the versatility to handle both binary and continuous treatments, and automatically identifies the binary treatment when there are only two unique values. To demonstrate this, we generate a data set similar to the previous example.

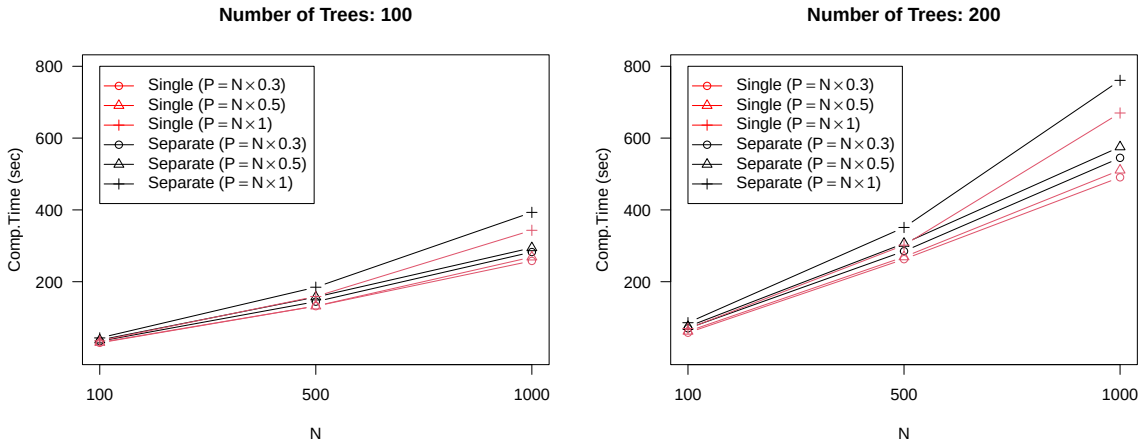
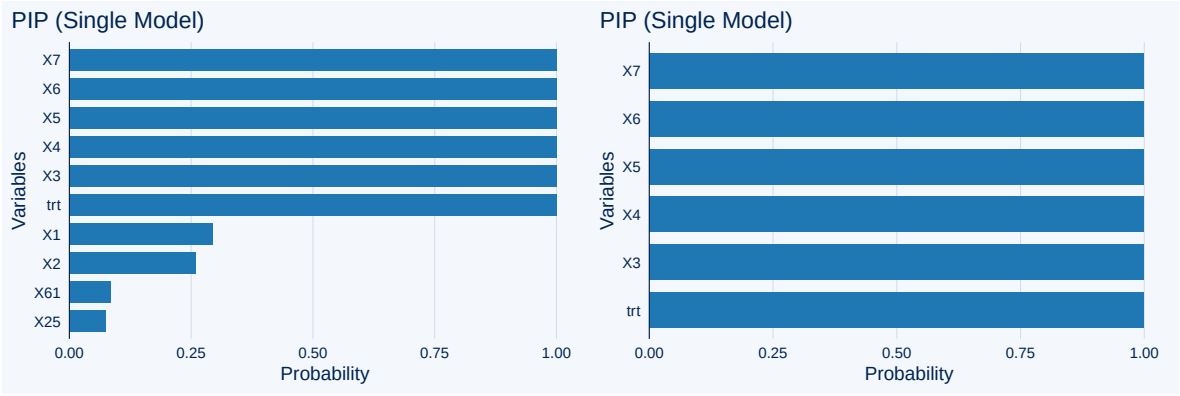


Figure 10: The computation times for both the single outcome model (red) and separate outcome model (black) based on the number of observations (N) under two different numbers of trees. A cross symbol ($+$) represents the scenario where the number of potential confounders (P) is equal to the number of observations (N), a triangle ($\triangle$) represents the scenario where $P = N \times 0.5$ and a circle ($\bigcirc$) represents the scenario where $P = N \times 0.3$. These results are obtained from 20000 MCMC iterations based on the scenario in Section 3.

```
R> set.seed(42)
R> N <- 300
R> P <- 100
R> cov <- list()
R> for (i in 1:P) {
+    cov[[i]] <- rnorm(N, 0, 1)
+  }
R> X <- do.call(cbind, cov)
R> h1 <- ifelse(X[, 1] < 0, 1, -1)
R> h2 <- ifelse(X[, 2] < 0, -1, 1)
R> mu_trt <- 0.5 + h1 + h2 - 0.5 * abs(X[, 3] - 1) + 0.5 * X[, 4] * X[, 5]
R> Trt <- rnorm(N, mu_trt, 0.3)
R> mu_y <- 1 * h1 + 1 * h2 - Trt + 1 * abs(X[, 3] + 1) +
+    1 * X[, 4] + exp(0.5 * X[, 5]) -
+    0.5 * Trt * abs(X[, 6]) - 0.5 * Trt * abs(X[, 7] + 1)
R> Y <- rnorm(N, mu_y, 0.3)
R> treatment <- quantile(Trt, 0.75)
R> control <- quantile(Trt, 0.25)
```

We use the function `single_bart()` to fit the generated data. The first and third quantile values of `Trt` will serve as the basis for comparing two different exposure levels. As arguments in `single_bart()`, we need to provide these two pre-specified exposure levels (a =`trt_treated` and a' =`trt_control`). In the this case, the causal estimand is $\Delta(a, a') = E[(Y(a) - Y(a')]$.

```
R> single_fit <- single_bart(
+    Y = Y, trt = Trt, X = X,
```



(a) Plotting PIP with `top_n` argument (b) Plotting PIP with `threshold = 0.5` argument

Figure 11: PIP plot for continuous exposure

```
+    trt_treated = treatment, trt_control = control,
+    num_tree = 200, num_chain = 4,
+    num_burn_in = 10000, num_thin = 5, num_post_sample = 2000
+  )
R> single_fit

`bartcs` fit by `single_bart()`

          mean       2.5%     97.5%
ATE -2.8097339 -4.2581469 -1.732448
Y1   0.9982417  0.2753606  1.677726
Y0   3.8079756  3.0967180  4.740133
```

Similar to other **bartcs** objects, the `summary()` and `plot()` functions can be applied to the continuous exposure scenario. Figure 11 displays a PIP plot, which demonstrates that out of 100 possible confounders, all of the true confounders except X_1, X_2, and two additional predictors were captured effectively, with high PIP values.

6. Summary and discussion

In conclusion, the **bartcs** R package is a powerful tool for causal inference using BART. It allows users to adjust for confounders and estimate treatment effects using a flexible non-parametric method. The package's ability to handle high-dimensional and non-linear confounding, binary exposure, and continuous exposure makes it a versatile tool for a wide range of applications. Additionally, the package's support for parallel computing and visualization of results make it a user-friendly and easy-to-interpret tool. The **bartcs** package is a valuable resource for researchers in various fields.

In this paper, we assessed the performance of the proposed method in a scenario where all true confounders are included in the potential confounder pool, and additional predictors for the outcome model are also present within the potential confounder pool. In this scenario, the proposed method demonstrated precise Average Treatment Effect (ATE) estimation and

accurately identified the true confounders. Moreover, in the study by Kim *et al.* (2023). the proposed method exhibited accurate confounder selection performance and reliable estimation of ATEs even in scenarios involving instrumental variables in the data generating process. This success is attributed to the satisfaction of the disjunctive cause criterion without instruments by the proposed method, as outlined in the introduction section (VanderWeele 2019). Additionally, the method demonstrated outstanding results in simulation scenarios with diverse effect sizes and varying numbers of true confounders.

The single outcome model and separate outcome model proposed in this paper both demonstrate excellent performance in confounder selection and average treatment effect estimation. However, in cases where a continuous treatment variable is required, the single outcome model should be applied. Additionally, as indicated in Section 4.1, the single outcome model has a slightly faster computation speed than the separate outcome model when the sample size is large because it uses one less BART model. On the other hand, the separate outcome model has the advantage of relatively faster convergence of the MCMC chain during the process of updating the selection probability vector of the BART prior using a simple Gibbs update. Therefore, it is necessary to selectively choose between the two models based on the context of the data being applied.

While not currently integrated into the **bartcs** package, the confounder selection method presented here using BART holds potential for extension to various data types. For count or categorical outcomes, it might be feasible to substitute the proposed outcome model with the log-linear BART model suggested by Murray (2021). Similarly, for survival outcomes, the survival BART model proposed by Sparapani, Logan, McCulloch, and Laud (2016) could serve as the outcome model. Exploring the specific computation algorithms for these extensions could be a fruitful avenue for future research.

One limitation of the proposed method is its lack of consideration for correlation and temporal relationships among potential confounders. Currently, no research has explored the distribution of weights in the selection probability vector when high correlation exists among covariates in the potential confounder pool. An approach worth investigating may involve leveraging a causal Directed Acyclic Graph (DAG) to constrain the selection of certain covariates in the prior setting of the selection probability vector. This too presents a promising direction for future research.

Computational details

The results in this paper were obtained using R 4.3.0 on a Mac Studio with a M1 chip and 128 GB of memory. **bartcs** 1.2.0 and **bacr** 1.0.1 were used for the analysis. R itself and all packages used are available from the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/.

Acknowledgments

This work is supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (NRF-2022R1F1A1062904).

References

Barbieri MM, Berger JO (2004). "Optimal Predictive Model Selection." *The Annals of Statistics*, **32**(3), 870–897.

Chipman HA, George EI, McCulloch RE, *et al.* (2010). "BART: Bayesian Additive Regression Trees." *The Annals of Applied Statistics*, **4**(1), 266–298.

Ding P, Li F (2018). "Causal inference." *Statistical Science*, **33**(2), 214–237.

Ding P, VanderWeele T, Robins JM (2017). "Instrumental variables as bias amplifiers with general outcome and confounding." *Biometrika*, **104**(2), 291–302.

Dorie V (2016). ***NPCI**: Non-parametrics for Causal Inference*. URL https://github.com/vdorie/npci.

Häggström J (2017). ***CovSelHigh**: Model-Free Covariate Selection in High Dimensions*. R package version 1.1.1, URL https://CRAN.R-project.org/package=CovSelHigh.

Häggström J (2018). "Data-Driven Confounder Selection via Markov and Bayesian networks." *Biometrics*, **74**(2), 389–398.

Hahn PR, Murray JS, Carvalho CM (2020). "Bayesian Regression Tree Models for Causal Inference: Regularization, Confounding, and Heterogeneous Effects." *Bayesian Analysis*, **15**(3), 965–1056. doi:10.1214/19-BA1195.

Hastie T, Tibshirani R, *et al.* (2000). "Bayesian Backfitting (with Comments and a Rejoinder by the Authors)." *Statistical Science*, **15**(3), 196–223.

Hill JL (2011). "Bayesian Nonparametric Modeling for Causal Inference." *Journal of Computational and Graphical Statistics*, **20**(1), 217–240. doi:10.1198/jcgs.2010.08162.

Kapelner A, Bleich J (2016). "bartMachine: Machine Learning with Bayesian Additive Regression Trees." *Journal of Statistical Software*, **70**(4), 1–40. doi:10.18637/jss.v070.i04.

Kim C, Tec M, Zigler CM (2023). "Bayesian Nonparametric Adjustment of Confounding." *Biometrics*, **79**(4), 3252–3265. doi:10.1111/biom.13833.

Lefebvre G, Delaney JA, McClelland RL (2014). "Extending the Bayesian Adjustment for Confounding Algorithm to Binary Treatment Covariates to Estimate the Effect of Smoking on Carotid Intima-Media Thickness: the Multi-ethnic Study of Atherosclerosis." *Statistics in Medicine*, **33**(16), 2797–2813.

Li F, Ding P, Mealli F (2023). "Bayesian causal inference: a critical review." *Philosophical Transactions of the Royal Society A*, **381**(2247), 20220153.

Linero AR (2018). "Bayesian Regression Trees for High-Dimensional Prediction and Variable Selection." *Journal of the American Statistical Association*, **113**(522), 626–636.

Louizos C, Shalit U, Mooij JM, Sontag D, Zemel R, Welling M (2017). "Causal Effect Inference with Deep Latent-Variable Models." *Advances in Neural Information Processing Systems*, **30**. doi:10.48550/arXiv.1705.08821. URL https://github.com/AMLab-Amsterdam/CEVAE.

Murray JS (2021). "Log-linear Bayesian additive regression trees for multinomial logistic and count regression models." *Journal of the American Statistical Association*, **116**(534), 756–769.

Myers JA, Rassen JA, Gagne JJ, Huybrechts KF, Schneeweiss S, Rothman KJ, Joffe MM, Glynn RJ (2011). "Effects of adjusting for instrumental variables on bias and precision of effect estimates." *American journal of epidemiology*, **174**(11), 1213–1222.

Oganisian A, Roy JA (2020). "Invited discussion for Bayesian regression tree models for causal inference: regularization, confounding, and heterogeneous effects." *Bayesian Anal*, **15**(3), 998–1006.

Papadogeorgou G, Li F (2020). "Invited discussion for Bayesian regression tree models for causal inference: regularization, confounding, and heterogeneous effects." *Bayesian Anal*, **15**(3), 1007–1013.

Plummer M, Best N, Cowles K, Vines K, Sarkar D, Bates D, Almond R, Magnusson A (2020). ***coda**: Output Analysis and Diagnostics for MCMC*. R package version 0.19-4, URL https://CRAN.R-project.org/package=coda.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Rosenbaum PR, Rubin DB (1983). "The Central Role of the Propensity Score in Observational Studies for Causal Effects." *Biometrika*, **70**(1), 41–55. ISSN 00063444.

Rubin DB (1974). "Estimating Causal Effects of Treatments in Randomized and Nonrandomized Studies." *Journal of Educational Psychology*, **66**(5), 688.

Shortreed SM, Ertefaie A (2017). "Outcome-Adaptive Lasso: Variable Selection for Causal Inference." *Biometrics*, **73**(4), 1111–1122.

Sparapani R, Spanbauer C, 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. doi:10.18637/jss.v097.i01.

Sparapani RA, Logan BR, McCulloch RE, Laud PW (2016). "Nonparametric survival analysis using Bayesian additive regression trees (BART)." *Statistics in medicine*, **35**(16), 2741–2753.

VanderWeele TJ (2019). "Principles of Confounder Selection." *European Journal of Epidemiology*, **34**(3), 211–219.

Wang C, Dominici F, Parmigiani G, Zigler CM (2015). "Accounting for Uncertainty in Confounder and Effect Modifier Selection when Estimating Average Causal Effects in Generalized Linear Models." *Biometrics*, **71**(3), 654–665.

Wang C, Parmigiani G, Dominici F (2012). "Bayesian Effect Estimation Accounting for Adjustment Uncertainty." *Biometrics*, **68**(3), 661–671.

Wilson A, Bondell HD, Reich BJ (2015). ***BayesPen**: Bayesian Penalized Credible Regions*. R package version 1.2, URL https://anderwilson.github.io/BayesPen.

Wilson A, Reich BJ (2014). "Confounder Selection via Penalized Credible Regions." *Biometrics*, **70**(4), 852–861.

Yoo Y (2023). ***bartcs**: Bayesian Additive Regression Trees for Confounder Selection*. R package version 1.2.0, URL https://CRAN.R-project.org/package=bartcs.

Yoshida K, Bartel A (2022). ***tableone**: Create 'Table 1' to Describe Baseline Characteristics with or without Propensity Score Weights*. R package version 0.13.2, URL https://CRAN.R-project.org/package=tableone.

Affiliation:

Chanmin Kim (corresponding author)
Department of Statistics
SungKyunKwan University
25-2 SungKyunKwan-ro, Jongno-gu
Seoul 03063, Korea
E-mail: chanmin.kim@skku.edu
URL: https://lit777.github.io/

and

Yeonghoon Yoo
Department of Statistics
SungKyunKwan University
25-2 SungKyunKwan-ro, Jongno-gu
Seoul 03063, Korea

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd