Generalizing Trimming Bounds for Endogenously Missing Outcome Data Using Random Forests

Introduction

In many experimental and quasi-experimental studies, the outcomes of interest are only observed for subjects who choose (or are chosen) to engage in the activity generating the outcome. This results in endogenously missing outcome data, where random or conditionally random treatment assignment alone cannot fully identify treatment effects. For example, consider the study by Santoro and Broockman (2022), where subjects were asked to have an online conversation about their “perfect day” with someone from a different political party (an outpartisan). Subjects informed about their conversation partner’s partisanship were more likely to complete the conversation and the post-treatment survey, introducing potential bias due to differential completion rates.

When outcome data is missing based on subjects' choices, traditional methods for estimating treatment effects can become biased or imprecise (Montgomery et al., 2018). Non-parametric partial identification bounds provide a way to address this missing data without making strong assumptions. However, basic bounding methods often produce very wide bounds that offer limited information (Lee, 2009). To overcome this, we propose a method that uses generalized random forests (GRF) to incorporate a large number of covariates, thereby narrowing the bounds and allowing for more precise and reliable inference.

Our methodology builds on the concept of trimming bounds, which leverages a monotonicity assumption to bound the average causal effect for always-responders—subjects whose outcomes are always observed regardless of the treatment assignment (Lee, 2009). We enhance this approach by applying GRF, a machine learning algorithm, to adjust for covariates. This approach allows for flexible control over covariates and avoids the need for parametric assumptions (Wager and Athey, 2018).

We demonstrate the benefits of our method through simulations and replication exercises, showing that our approach provides more informative bounds compared to basic trimming bounds, especially in scenarios with high-dimensional covariates. Specifically, we use a simulation study and two replication exercises to illustrate the advantages of our approach. We also offer an open-source R package, CTB, for practitioners to implement this method in various empirical settings.

Our method allows researchers to achieve more precise estimates of treatment effects in the presence of endogenously missing data, enhancing the robustness and credibility of experimental and quasi-experimental studies (Chernozhukov et al., 2018).

Methods

Set up

Consider a randomized or natural experiment with subjects indexed by i=1,...,Ni = 1, ..., Ni=1,...,N, for which we have measured PPP pre-treatment covariates collected in the vector Xi=(Xi1,Xi2,...,XiP)X_i = (X_{i1}, X_{i2}, ..., X_{iP})Xi=(Xi1,Xi2,...,XiP). For each subject, we always observe values of the covariates, the treatment status Di∈{0,1}D_i \in \{0, 1\}Di∈{0,1}, and the response indicator Si∈{0,1}S_i \in \{0, 1\}Si∈{0,1}. The realized outcome YiY_iYi is observed only when Si=1S_i = 1Si=1. For Di=1D_i = 1Di=1 and Di=0D_i = 0Di=0, respectively, potential outcomes are given by (Yi(1),Yi(0))(Y_i(1), Y_i(0))(Yi(1),Yi(0)) and potential response indicators are given by (Si(1),Si(0))(S_i(1), S_i(0))(Si(1),Si(0)). The realized outcome is Yi=DiYi(1)+(1−Di)Yi(0)Y_i = D_i Y_i(1) + (1 - D_i) Y_i(0)Yi=DiYi(1)+(1−Di)Yi(0) and the realized response is Si=DiSi(1)+(1−Di)Si(0)S_i = D_i S_i(1) + (1 - D_i) S_i(0)Si=DiSi(1)+(1−Di)Si(0). Define UiU_iUi to be unobserved factors that affect both (Si(1),Si(0))(S_i(1), S_i(0))(Si(1),Si(0)) and (Yi(1),Yi(0))(Y_i(1), Y_i(0))(Yi(1),Yi(0)).

We make the following assumption on treatment assignment:

Assumption 1 (Strong Ignorability):
(Yi(1),Yi(0),Si(1),Si(0))⊥Di ∣ Xi(Y_i(1), Y_i(0), S_i(1), S_i(0)) \perp D_i \mid X_i(Yi(1),Yi(0),Si(1),Si(0))⊥Di ∣ Xiϵ<P(Di=1 ∣ Xi) <1−ϵ, with ϵ>0\epsilon < P(D_i = 1|X_i) < 1 - \epsilon, \text{ with } \epsilon > 0ϵ<P(Di=1 ∣ Xi)<1−ϵ, with ϵ>0

$$(Yi(1), Yi(0), Si(1), Si(0)) \perp Di \mid Xi, (Y_i(1), Y_i(0), S_i(1), S_i(0)) \perp D_i | X_i (Yi(1), Yi(0), Si(1), Si(0)) \perp Di \mid Xi \epsilon < P(D$$

The first part states that the treatment is conditionally independent of both the potential outcomes and the potential responses. The second part requires that the propensity score, p(Xi)=P(Di=1 ∣ Xi)p(X_i) = P(D_i = 1|X_i)p(Xi)=P(Di=1 ∣ Xi), is strictly bounded between 0 and 1. This assumption holds in experiments when treatment is randomly or conditionally randomly assigned. In observational studies, the assumption implies that the covariate vector XiX_iXi includes all confounders (Rosenbaum and Rubin, 1983).

The directed acyclic graph (DAG) in Figure 1 illustrates relationships that our setting admits. For the study by Santoro and Broockman (2022), UiU_iUi could represent the degree of political interest. Our approach also admits the possibility that YiY_iYi directly affects missingness. Conditioning on SiS_iSi introduces collider biases, as explained by Elwert and Winship (2014) and Montgomery et al. (2018).

To understand the source of bias, we classify subjects into four different types based on their responses to the treatment: always-responders (Si(0)=Si(1)=1S_i(0) = S_i(1) = 1Si(0)=Si(1)=1), never-responders (Si(0)=Si(1)=0S_i(0) = S_i(1) = 0Si(0)=Si(1)=0), compliers (Si(0)=0,Si(1)=1S_i(0) = 0, S_i(1) = 1Si(0)=0,Si(1)=1), and defiers (Si(0)=1,Si(1)=0S_i(0) = 1, S_i(1) = 0Si(0)=1,Si(1)=0). These subgroups are examples of principal strata, similar to those in instrumental variable settings (Frangakis and Rubin, 2002). As shown in Table 1, we see that among units with observed outcomes (Si=1S_i = 1Si=1), the control group may consist of both defiers and always-responders, while the treatment group is a mixture of compliers and always-responders.

We work with the following assumption on the selection process:

Assumption 2 (Monotonic Selection):

$$Si(1) \geq Si(0) S_i(1) \geq S_i(0) Si(1) \geq Si(0)$$

This assumption excludes the existence of defiers and is similar to the “no defiers” assumption for instrumental variables (Angrist et al., 1996). For example, in the study by Santoro and Broockman (2022), subjects who would engage when uninformed of their partners’ partisanship would be assumed to engage when informed. While our re-analysis suggests that monotonicity may not hold unconditionally, we maintain this assumption for now and discuss its relaxation in Section 6.

In Table 2, we provide empirical examples from different fields of political science (American politics, comparative politics, public administration, international relations, political economy, and methodology) that fit into our setup, including the implications of Assumption 2 and the sub-population captured by always-responders in each context.

Simulation

To demonstrate the usage of the main functions in the `CTB` package, we created a simulated dataset called `dat_full`. This dataset consists of 1,000 units where the treatment is randomly assigned with a probability of 0.5. For each unit, we included 10 covariates (`X1` to `X10`), all uniformly distributed on the interval [0,1][0, 1][0,1]. Among these covariates, only `X1` influences both the outcome and the response, simulating a realistic scenario where only a subset of covariates are relevant. Additionally, there is an unobservable variable, denoted as `U`, which also affects both the outcome and the response and follows a uniform distribution on [−2,2][-2, 2][−2,2].

The data generation process (DGP) ensures that the treatment effect and the response mechanism are driven by both observable and unobservable factors, making the simulated dataset a robust test case for the `CTB` functions. In this setup, the covariates and the unobservable variable introduce variability and potential bias in the treatment effect estimates, reflecting common challenges in real-world data.

To visualize the data, we first plotted the potential outcomes (`Y0` and `Y1`) against one of the covariates (`X2`). This plot shows the potential outcomes for treated and control units, highlighting the differences due to treatment. Black points represent the potential outcomes under treatment, while white points represent the outcomes without treatment.

Next, we plotted the experimental outcomes, differentiating between units that completed the response and those that did not (attrited). Black points indicate the observed outcomes for units that completed the response, while red points represent outcomes for units that attrited. This plot helps us understand how attrition affects the observed data.

Finally, we visualized the observed outcomes for units that completed the response, distinguishing between treated and control units. This plot provides a clear comparison of the treatment effect across the covariate `X2`, showing the impact of treatment on the observed outcomes.

These visualizations demonstrate the practical application of the `CTB` package in handling endogenously missing outcome data, offering insights into how covariates and unobservable factors influence treatment effects in experimental data.

Features of the CTB Package

The `CTB` function, designed by Samii, Wang, and Zhou (2023), estimates aggregated and conditional trimming bounds using information from covariates. This method leverages covariates to tighten the bounds on treatment effect estimates in the presence of endogenously missing outcome data, enhancing the robustness and precision of causal inference.

To use the `CTB` function, you need to provide a data frame containing all necessary variables. The function signature is as follows:

```
CTB(data, seed = NULL, Y, D, S, X = NULL, W = NULL, Pscore = NULL,
    regression.splitting = FALSE, cv_fold = 5, trim_l = 0, trim_u = 1,
    aggBounds = TRUE, cBounds = FALSE, X_moderator = NULL,
    direction = NULL, cond.mono = TRUE)
```

The main arguments of the function include `data`, which is the dataset containing the outcome indicator `Y`, the treatment indicator `D`, and the response indicator `S`. The treatment and response indicators should be binary variables (0 and 1). You can also specify covariates using the `X` argument, which accepts a vector of column names representing the covariates.

Additionally, the function allows for specifying a random seed through the `seed` argument, ensuring reproducibility of the sampling splitting. If you have sampling weights, you can include them with the `W` argument. The propensity score can be provided using the `Pscore` argument; if it is not specified, the function will estimate the probability of being treated from the data.

For advanced usage, the `regression.splitting` flag can be set to `TRUE` to enable regression splitting. The `cv_fold` argument specifies the number of folds used in cross-validation, with a default value of 5. The `trim_l` and `trim_u` arguments define the lower and upper bounds of the trimming range, respectively.

The function calculates aggregated bounds by default, controlled by the `aggBounds` flag. Conditional bounds can also be estimated by setting the `cBounds` flag to `TRUE` and providing covariate values for `X_moderator`. The `direction` argument can be used to specify the direction of monotonic selection if applicable, and the `cond.mono` flag controls whether the assumption of conditional monotonic selection is applied.

Here's an example of how to use the `CTB` function in practice. First, load the `CTB` package and your dataset. Then, call the function with the appropriate arguments:

```
RCopy code
library(CTB)
data(simData)

out <- CTB(data = simData, seed = 1234, Y = "Y", D = "D", S = "S",
           X = c(names(simData)[2:5]), Pscore = "Ps", regression.splitting = FALSE,
           cv_fold = 5, aggBounds = TRUE, cBounds = TRUE, X_moderator = NULL,
           direction = NULL, cond.mono = TRUE)

print(out)
```

In this example, the dataset `simData` is used with specified columns for the outcome (`Y`), treatment (`D`), and response (`S`). Covariates are included from columns 2 to 5. The function calculates both aggregated and conditional bounds with default settings, providing comprehensive insights into the treatment effects.

Dependencies for the function include the `CTB` package, which must be installed and loaded into your R session. When working with large datasets, consider optimizing computational resources or using parallel processing to enhance performance. Common pitfalls include ensuring all specified columns in the `data` argument exist and are correctly named, as mismatched names can lead to errors.

The output of the `CTB` function includes estimates of the lower and upper bounds, both aggregated and conditional, along with their standard errors. These results can be interpreted to understand the range within which the true treatment effect lies, providing valuable insights for researchers dealing with endogenously missing data in their studies.

Application

Example: Santoro and Broockman (2022) Study

In this section, we demonstrate how to apply the `CTB` package using data from the Santoro and Broockman (2022) study. This study examines the impact of informing subjects about their conversation partner's political affiliation on their expressed warmth towards outpartisans. The experiment involves subjects engaging in an online conversation about their "perfect day" with someone from a different political party.

▼ Direct from original paper:

Santoro and Broockman (2022, Study 1) invited subjects to have a video chat on an online platform with a partner from a different party. The theme of the conversation is what their perfect day would be like. The study started with 986 subjects who satisfied the screening criteria. The subjects were then randomly assigned into either the treatment group (Di = 1), in which they were informed that the partner would be an outpartisan, or the control group (Di = 0), in which they received no extra information.8 Among subjects that were assigned to a treatment condition, 45.2% of the treated subjects and 39.5% of the control subjects completed the conversation and post-treatment survey. The authors examined the treatment effect on a series of outcome variables that measure a subject's attitude toward the other parties and found significant effects of the treatment in the short run.
The authors indicate that the treatment effect on the response rates had a p-value of 0.055, and their omnibus test for covariate balance with respect to education, race/ethnicity, gender, age, and party identification had a p-value of 0.28. Nonetheless, the difference in response rates is not trivial, and other confounding factors may be imbalanced beyond those tested. We can use our covariate-tightened trimming bounds to assess the robustness of their findings. Our inference targets the "always responders" that would complete the conversation and survey regardless of being informed about their partner's partisanship. We focus on the "warmth toward outpartisan voters" outcome (measured by a rescaled thermometer) and rely on the same covariates the authors selected for their analysis, including the age, gender, race, education level, and party identification of the subjects, as well as their pre-treatment outcome.

In their 2022 study, Santoro and Broockman embarked on an intriguing research endeavor, examining the effects of inter-party dialogue on participants' attitudes toward outpartisans. The study design was as follows: Subjects were invited for a video chat with a partner from a different political party, with the conversation revolving around the theme of their "perfect day". Initially, the pool of subjects comprised 986 individuals who fulfilled the screening criteria. These subjects were then randomly allocated to one of two groups.

In the treatment group (Di = 1), subjects were cognizant of the fact that their conversation partner belonged to a different political party. Conversely, subjects in the control group (Di = 0) were not privy to this information. The completion rates for the post-treatment surveys revealed an interesting pattern: Among the subjects designated to a treatment condition, 45.2% in the treatment group and 39.5% in the control group completed the post-conversation survey.

The authors performed a comprehensive examination of the treatment effect on a series of outcome variables that gauged a subject's political attitudes, specifically towards outpartisans. The results demonstrated significant short-term

effects of the treatment.

However, the authors also noted a p-value of 0.055 for the treatment effect on response rates, and a p-value of 0.28 for their omnibus test for covariate balance with respect to key demographic factors, such as education, race/ethnicity, gender, age, and party identification. While these results suggest some degree of balance, they also underscore a non-trivial difference in response rates and the potential for other imbalances in untested confounding factors.

To assess the robustness of these findings, we applied our covariate-tightened trimming bounds. Our analysis focuses on the "always responders", defined as subjects who would complete the conversation and survey irrespective of their awareness of their partner's partisanship. We evaluated the "warmth toward outpartisan voters" outcome, measured by a rescaled thermometer, and considered the same covariates used by the authors for their analysis. These covariates included age, gender, race, education level, and party identification of the subjects, in addition to their pre-treatment outcome.

~~The treatment in this study is the disclosure of the conversation partner's political affiliation before the conversation. The primary outcome of interest is the level of warmth expressed by the subjects towards their conversation partner after the conversation. Only subjects who completed the post-conversation survey provided their outcome data, introducing endogenously missing data.~~

To analyze this data, we use the `CTB` function from the `CTB` package. Below is the R code to set up and run the analysis:

```
set.seed(1234)
library(CTB)
library(grf)
library(ggplot2)

# Load the example data from the Santoro and Broockman (2022) study
data(santoro_broockman)

# Apply the CTB function
out <- CTB(data = santoro_broockman, seed = 1234, Y = "warmth", D = "informed", S = "complete
d",
           X = c("age", "gender", "education"), Pscore = "propensity_score", regression.splitt
ing = FALSE,
           cv_fold = 5, aggBounds = TRUE, cBounds = TRUE, X_moderator = NULL,
           direction = NULL, cond.mono = TRUE)

# Print the results
print(out)
```

In this example, we load the `CTB` package and the necessary libraries. We then load the data from the Santoro and Broockman (2022) study, which includes the variables `warmth` (the outcome), `informed` (the treatment indicator), `completed` (the response indicator), and additional covariates such as `age`, `gender`, and `education`.

We run the `CTB` function on this dataset, specifying the outcome variable `warmth`, the treatment indicator `informed`, and the response indicator `completed`. We include the covariates `age`, `gender`, and `education`, and provide the propensity score column `propensity_score`. We set `regression.splitting` to `FALSE` and use 5-fold cross-validation. The function calculates both aggregated and conditional bounds.

The output from the `CTB` function includes estimates of the lower and upper bounds for both aggregated and conditional trimming bounds. These bounds provide insights into the range within which the true treatment effect lies, accounting for the influence of covariates and potential biases due to endogenously missing data.

Here LeeBounds = 1 indicates we want Lee bounds and cond.mono = 0 indicates we want strict monotonicity for Lee bounds. For our CTTB, the conditional monotonicity is always set as default. Setting cond.mono = 1 will allow the Lee bounds to have conditional monotoncity.

```{r santoro2022 result2, message = FALSE, warning = FALSE}
result.lee <- CTB(data = dat, seed = seed,
                                    Y = Y, D = D, S = S, X = X, W = NULL, Pscore = "Ps",
                                    regression.splitting = FALSE, cv_fold = 5,
                                    aggBounds = 0,
                                    cBounds = 0, X_moderator = NULL,
                                    direction = NULL,
                                    cond.mono = 1)
tau_l_est_lee_condmono <- result.lee[["tau_l_est_lee"]]
tau_u_est_lee_condmono <- result.lee[["tau_u_est_lee"]]
se_tau_l_est_lee_condmono <- result.lee[["se_tau_l_est_lee"]]
se_tau_u_est_lee_condmono <- result.lee[["se_tau_u_est_lee"]]```

The average missing rate is  0.5756.

The results are plots as follows:
```