RCTrep: An R Package for the Validation of Estimates of the Average Treatment Effect

Lingjie Shen (L.Shen@uvt.nl)
Tilburg University https://orcid.org/0000-0002-9354-8088
Gijs Geleijnse
IKNL
Maurits Kaptein
Tilburg University https://orcid.org/0000-0002-6316-7524

Method Article

Keywords: observational data, randomized controlled trial data, the average treatment effect, validation

Posted Date: August 9th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2559287/v2

Additional Declarations:
Competing interests: The authors declare no competing interests.

RCTrep: An R Package for the Validation of Estimates of the Average Treatment Effect

Lingjie Shen
Tilburg University

Gijs Geleijnse
IKNL

Maurits Kaptein
JADS

Abstract

Despite the recent development of numerous methods aiming to estimate individual-level treatment effects based on observational data, assessing the validity of these estimates remains challenging. It is often unclear whether the observational data meet the assumptions imposed by a method. Additionally, there is often great flexibility in model choice when implementing a given method. This article introduces the R package **RCTrep**, designed for easy assessment of the validity of estimates of the average treatment effect obtained from observational data. This is achieved by a) making it easy to obtain and visualize estimates derived using a large variety of methods, and b) ensuring that these estimates are easily compared to a gold standard on population and subpopulation levels. **RCTrep** outlines a four-step workflow, namely, set-selection, estimation, diagnosis, and validation. The package provides a simple dashboard to review the obtained results. This article serves as a user guide for researchers aiming to leverage the potential of observational data to inform personalized treatment.

Keywords: observational data, randomized controlled trial data, the average treatment effect, validation.

1. Introduction

There is a growing interest in estimating the average treatment effect (ATE) using observational data (Bica *et al.* 2021; Colnet *et al.* 2020; Stuart 2010). Numerous methods have been proposed, capitalizing on ideas such as the G-computation method (Hill 2011; Hitsch and Misra 2018; Atan *et al.* 2018; Wager and Athey 2018), the propensity score-based method (Xie *et al.* 2012; Rosenbaum and Rubin 1983; Austin 2011), the doubly robust method (Bang and Robins 2005; Funk *et al.* 2011), and the representation learning method (Yao *et al.* 2018; Johansson *et al.* 2020), etc.. For a more detailed overview of related literature, see the recent survey by Jiang *et al.* (2021). Despite this large contemporary literature, there is no "single

best" method that can consistently provide the most accurate estimates of the ATE on a variety of observational datasets (Dorie *et al.* 2019). Hence, given an observational dataset at hand, in the absence of a ground-truth, it is challenging to assess the validity and select the most appropriate method.

In this paper, we present the **RCTrep** package, an R package designed to easily implement a large number of methods for the ATE estimation using observational data. Next, we allow for the assessment of the validity of these estimates by enabling easy comparison to unbiased estimates obtained from randomized controlled trials (RCTs).

We formulate core elements of the approach taken in **RCTrep** as follows: Consider a target population $\mathcal{P}_{\boldsymbol{\theta}}$ defined by a true data generation mechanism, from which two samples are drawn: $\mathcal{S}^{rct}$ (an RCT sample) and $\mathcal{S}^{obs}$ (an observational sample). For the RCT sample, we assume, without loss of generality, the simple random sampling and the randomized treatment assignment. For the observational sample, we assume a known sampling mechanism and an unknown treatment assignment mechanism. Let $\boldsymbol{X} = (X_1, ..., X_d) \in \mathbb{R}^d$ denote a $d-$ dimensional vector of pre-treatment outcome predictors; let $\boldsymbol{X}_s \subseteq \boldsymbol{X}$ denote a vector of selection predictors of the observational sample; furthermore, let $\hat{\mathbb{T}} = \{\hat{\tau}_0, \hat{\tau}_1, ..., \hat{\tau}_n\}$ denote a set of estimators of the conditional average treatment effect (CATE). In this setting, first of all, **RCTrep** makes it easy to compute $\hat{\tau}(\boldsymbol{X})$. Next, **RCTrep** makes it easy to validate and select the most appropriate estimates according to the following metric:

$$\mathbb{L}(\hat{\tau}_0^{\mathcal{S}^{rct}}; \hat{\tau}^{\mathcal{S}^{obs}}) = \mathbb{L}\left(\hat{\tau}_0^{\mathcal{S}^{rct}}, \sum_{i \in \mathcal{S}^{obs}} \hat{w}(\boldsymbol{x}_{si})\hat{\tau}(\boldsymbol{x}_i)\right), \quad s.t. \;\; \hat{p}(\boldsymbol{x}_s) = \hat{w}(\boldsymbol{x}_s)\hat{q}(\boldsymbol{x}_s), \; \sum_{i \in \mathcal{S}^{obs}} \hat{w}(\boldsymbol{x}_{si}) = 1 \tag{1}$$

where $\hat{\tau}_0^{\mathcal{S}^{rct}}$ is an unbiased estimate of the ATE of the target population obtained from $\mathcal{S}^{rct}$ using the estimator $\hat{\tau}_0$ (the difference in means of outcomes between groups), $\hat{p}(\boldsymbol{x}_s)$ and $\hat{q}(\boldsymbol{x}_s)$ are the empirical densities of $\boldsymbol{x}_s$ in $\mathcal{S}^{rct}$ and $\mathcal{S}^{obs}$ respectively, $\hat{w}(\boldsymbol{x}_{si})$ is a normalized weight for $i \in \mathcal{S}^{obs}$, and $\hat{w}$ is an estimator of the weight. Thus, the **RCTrep** package allows for the comparison of estimates obtained from an observational dataset to those obtained from an RCT dataset by adjusting for the treatment assignment mechanism and the sampling mechanism of the observational dataset. **RCTrep** outlines a four-step workflow to implement the validation:

- **Step 1: Set-selection.** Users select two sets of covariates $\boldsymbol{X}$ and $\boldsymbol{X}_s$. These covariates are used to model $\hat{\tau}(\boldsymbol{X})$ and $\hat{w}(\boldsymbol{X}_s)$ respectively.

- **Step 2: Estimation.** Users specify two estimators, $\hat{\tau}$ and $\hat{w}$, and initiate two objects of the class `TEstimator` and `SEstimator` accordingly. By specification, users provide a method for estimating the ATE of population and subpopulations stratified by $\boldsymbol{X}$ and a method for estimating the weight for each individual.

- **Step 3: Diagnosis.** The **RCTrep** package provides a number of statistics to diagnose assumptions for these specified methods (i.e., for the choice of `TEstimator` and `SEstimator`).

- **Step 4: Validation.** Finally, users initiate an object of the class `Fusion`. This object integrates estimates of the ATE of population and subpopulations obtained from $\mathcal{S}^{rct}$ and $\mathcal{S}^{obs}$ and computes metrics $\mathbb{L}$.

For more elaboration of these four steps, see section 5. To the best of our knowledge, **RCTrep** is the only package that allows users to estimate the ATE using observational data and assess the validity of these estimates using RCT data.

The remaining part of the paper proceeds as follows: after a brief review of the related literature and an illustrating example, section 2 formulates the problem setup for the validation of estimates of the ATE. Next, section 3 details our approach. Section 4 provides an overview of the R package **RCTrep** and introduces core classes and functions. Section 5 outlines a four-step workflow of **RCTrep** package using an example. Section 6 demonstrates three additional examples, i.e., validation at scale, validation using aggregate data, and validation using synthetic RCT data. Finally, we provide suggestions for future study in section 7.

1.1. Related work

Currently, although there are a number of software for the treatment effect estimation using observational data, e.g., Python libraries **CausalML** (Zhao and Liu 2023), **EconML** (Research 2019), **DoWhy** (Sharma *et al.* 2019), and R package **causaleffect** (Tikka and Karvanen 2017), software for assessing the validity of estimates of the ATE obtained from observational dataset by comparison to RCT data are, to our best knowledge, non-existent (Mayer *et al.* 2022). Earlier work by Wendling *et al.* (2018), Alaa and Van Der Schaar (2019), Schuler *et al.* (2017), Powers *et al.* (2018), Franklin *et al.* (2014), and Cheng *et al.* (2022), and existing software packages such as the R package **MethodEvaluation** (Schuemie *et al.* 2020), the Python package **Causality-Benchmark** (Shimoni *et al.* 2018), and the Python package **JustCause** (Franz 2020), do approximate a data generation mechanism for a given observational dataset, and use the simulated truth of treatment effects for the validation. These methods implicitly assume no unmeasured confounders. An overview of existing software for treatment effects validation is provided in Table 1. The table shows that **RCTrep** is the only package using unbiased estimates from an RCT as a surrogate of truth. In addition, **RCTrep** provides both the regulatory agreement and the estimate agreement as evaluation metrics (Franklin *et al.* 2020).

On the other hand, there is a growing body of studies focusing on generalization or transportation of estimates of the ATE of a population to another population (Dahabreh *et al.* 2020; Ackerman *et al.* 2021; Dong *et al.* 2020; Cinelli and Pearl 2021; Rudolph *et al.* 2018). Approaches used in these studies are closely related to that of **RCTrep**, however, **RCTrep** is different from them with respect to the motivation - validating estimates of the ATE obtained from an observational dataset and selecting the most appropriate one accordingly. **RCTrep** serves as a tool for people who want to leverage the potential of observational data to inform personalized treatment.

1.2. Strengths and limitations of our work

RCTrep makes several contributions to the methodology and software design. First, unlike existing studies and relevant packages which validate estimates of the ATE using simulated data (Wendling *et al.* 2018; Alaa and Van Der Schaar 2019; Schuler *et al.* 2017; Franklin *et al.* 2014; Schuemie *et al.* 2020; Shimoni *et al.* 2018), **RCTrep** is the only package that compares to unbiased estimates of the ATE obtained from a real dataset. Second, **RCTrep** validates estimates on both population and subpopulation levels, providing a deeper understanding of the error of a method. For instance, a high-bias method may have a relatively low bias at a population level but may have a high bias at subpopulation levels. Third, **RCTrep** can

	Task	Package			
		MethodEvaluation	**CausalityBenchmark**	**JustCause**	**RCTrep**
Methods	propensity score	✓		✓	✓
	G_computation	✓			✓
	Doubly robust	✓		✓	✓
Sample space	population	✓	✓	✓	✓
	subpopulation		✓	✓	✓
Metrics	(R)MSE	✓	✓	✓	✓
	PEHE				
	Bias		✓		
	confidence interval		✓		✓
	coverage	✓	✓		
	AUC	✓			
	mean precision	✓			
	type 1 error	✓			
	type 2 error	✓			
	Regulatory agreement				✓
	Estimate agreement				✓
Truth	simulated value	✓	✓	✓	
	unbiased estimate				✓

Table 1: Comparisons of packages for the validation of estimates of the ATE with a focus on the provided options of methods, the sample space based on which estimates of the ATE are to validate, evaluation metrics, and the truth.

validate estimates using aggregate data of subpopulations, which can generate the approximately same results as those using an individual-level dataset. **RCTrep** also provides functions to generate synthetic RCT datasets based on available marginal distributions of covariates. Fourth, **RCTrep** provides a structured way to implement the validation. For instance, in the set-selection step, users can select different adjustment sets; in the estimation step, users can select different methods and modeling techniques for the estimation of the ATE and weights. Results from different settings can be easily assessed. Lastly, the design structure of **RCTrep** has advantages over other packages and can be easily extended for other motivations. For instance, **RCTrep** can be used to compare estimates of the ATE from multiple data sources by aligning the four-step workflow with data partners.

1.3. Demonstration of usage

Codes below demonstrate how to implement the validation. The results are presented in Figure 1.

```
R> library("RCTrep")
R> output <- RCTREP(TEstimator = "G_computation", SEstimator = "Exact",
+    outcome_method = "BART",
+    source.data = RCTrep::source.data,
+    target.data = RCTrep::target.data,
+    vars_name = list(outcome_predictors =
+    c("x1","x2","x3","x4","x5","x6"),
+    treatment_name = c('z'),
+    outcome_name = c('y')),
```

```
+     selection_predictors = c("x2","x6"),
+     stratification = c("x1","x3","x4","x5"),
+     stratification_joint = TRUE)
R> fusion <- Fusion$new(output$target.obj,
+     output$source.obj,
+     output$source.rep.obj)
R> fusion$plot()
```

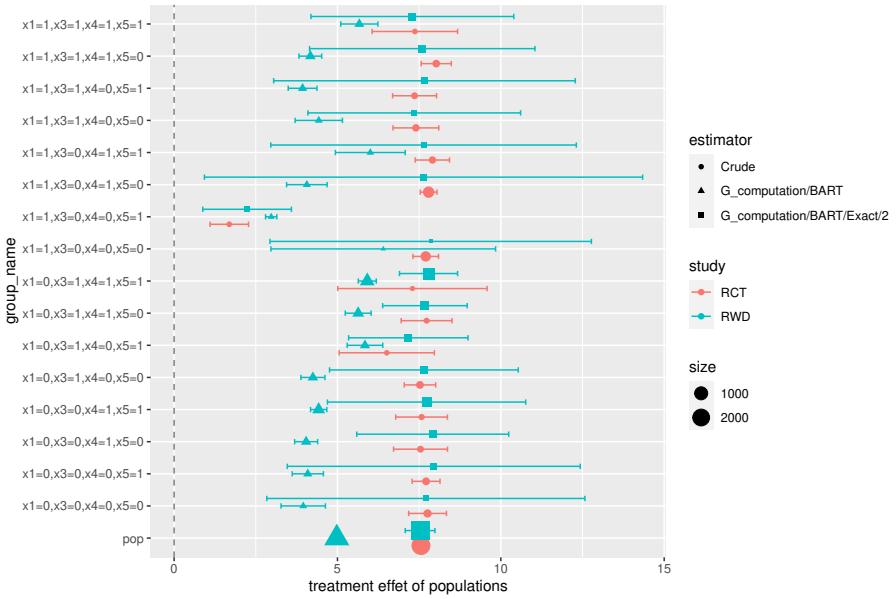


Figure 1: The validation of estimates of the ATE on population and subpopulation levels using **RCTrep**.

Descriptions of the input arguments in the function `RCTREP()` are as follows:

- `TEstimator` specifies a method to adjust for the treatment assignment mechanism;
- `SEstimator` specifies a method to adjust for the sampling mechanism;
- `outcome_method` specifies a modeling approach for the method `TEstimator`;
- `target.data` and `source.data` specify an RCT dataset and an observational dataset;
- `vars_name` specifies covariate names of the treatment, the outcome, and pre-treatment outcome predictors which are used to adjust for the treatment assignment mechanism;
- `selection_predictors` specifies covariate names of sample selection predictors of the observational data, which are used to adjust for the sampling mechanism;
- `stratification` and `stratification_joint` specify the selection of subpopulations based on levels of individual or joint covariates indicated in `stratification`.

In the above example, we use `G_computation` method to adjust for the treatment assignment mechanism and we use the `Exact` matching method to adjust for the sampling mechanism. We use Bayesian additive regression trees (BART) to model the outcome indicated by `outcome_method = "BART"`. We specify `outcome_predictors = c("x1","x2","x3","x4", "x5","x6")` and `selection_predictors = c("x2","x6")`. In this example, since `x2,x6` are the only set of selection predictors that can lead to a discrepancy of estimates of the ATE between two datasets, they are the minimal set of `selection_predictors` for the estimation of the weights. The results in Figure 1 show that estimates from the observational data (indicated by `G_computation/BART/Exact/2`) are close to the unbiased estimates from the RCT data (indicated by `Crude`), and hence these estimates from the observational data are arguably valid. Without properly adjusting for the sampling mechanism, a large discrepancy in estimates between an RCT dataset and an observational dataset can be observed, as shown by the large discrepancy in estimates between `Crude` and `G_computation/BART`, which might be wrongly attributed to unadjusted confounders in the observational dataset. See section 6 for more working examples.

2. Problem setup

In this section, we formulate the problem setup for validating estimates of the ATE. An overview of the notation used throughout this paper is provided in Appendix A.

2.1. Estimators for the ATE

We consider potential outcomes framework for estimating the ATE (Imbens and Rubin 2015). Let $\boldsymbol{X}$ denote a d-dimensional vector of all pre-treatment outcome predictors; $T \in \{0, 1\}$ denote a binary treatment indicator where 1 and 0 denote the treatment and the control, respectively; Y denote outcomes of interest, $Y(t)$ denote the potential outcome had the individual received $T = t$. The observed outcome of individual i under the received T_i is denoted as $Y_i = T_i Y_i(1) + (1 - T_i)Y_i(0)$. The individual-level treatment effect is defined as the simple difference $\tau_i = Y_i(1) - Y_i(0)$, the CATE is defined as $\tau(\boldsymbol{X}) = \mathbb{E}[Y(1) - Y(0) \mid \boldsymbol{X}]$, and the ATE is defined as $\tau = \mathbb{E}[\tau(\boldsymbol{X})]$, where $(\boldsymbol{X}, Y(1), Y(0)) \sim \mathcal{P}_{\boldsymbol{\theta}}$, and $\mathcal{P}_{\boldsymbol{\theta}}$ is a target population with a data generation mechanism parameterized by $\boldsymbol{\theta}$. A simple random sample is drawn from the target population. The treatment is assigned for each individual in the sample and the corresponding outcome is observed. The sample with observed data is denoted as $\mathcal{S} = \{(\boldsymbol{X}_i, T_i, Y_i); i = 1, ..., n\}$

2.2. Validation of estimates of the ATE

We now consider a set of candidate estimators of the CATE $\hat{\mathrm{T}} = \{\hat{\tau}_0, \hat{\tau}_1, ..., \hat{\tau}_n\}$, where $\hat{\tau}(\boldsymbol{X}) : \mathcal{X} \mapsto \mathbb{R}$. These may include, for example, different methods (the G-computation method, the inverse propensity-score weighting (IPW) method, the doubly robust (DR) method, the difference in means) combined with different modeling choices (e.g., BART, gaussian process, causal forest), and different hyper-parameter settings of one model, etc.. The accuracy of an estimator $\hat{\tau}$ for the estimation of the ATE is characterized by a distance measure $\mathbb{L}$ as an

evaluation metric, and the most accurate estimate of the ATE is derived based on:

$$\hat{\tau}^* = \underset{\hat{\tau}\in\mathrm{T}}{\arg\min}\, \mathbb{L}\left(\tau, \hat{\tau}\right) = \underset{\hat{\tau}\in\mathrm{T}}{\arg\min}\, \mathbb{L}\left(\tau, \sum_{\boldsymbol{x}\in\mathcal{S}} \hat{p}(\boldsymbol{x})\hat{\tau}(\boldsymbol{x})\right) \tag{2}$$

Since τ is not observed, the metric in Equation 2 can not be measured, hindering the direct validation of $\hat{\tau}$ using $\mathcal{S}$. In the following section, we provide our validation approach.

3. Validating estimates using RCT data

In this section, we elaborate our approach to validating estimates of the ATE. In section 3.1, we start by elaborating why an estimate of the ATE using an RCT dataset can be regarded as an unbiased estimate of the ATE of a target population. Next, in section 3.2 we elaborate how to use these estimates obtained from the RCT dataset to validate estimates obtained from an observational dataset.

3.1. An RCT provides unbiased estimates of the ATE

By definition, the treatment effect for each individual is not observed and can only be estimated. The following two assumptions allow for an unbiased estimate of the ATE:

Assumption 1 T-ignorability: $Y(1), Y(0) \perp\!\!\!\perp T \mid \boldsymbol{X}_t$

Assumption 2 T-overlap: $0 < P(T = 1 \mid \boldsymbol{X}_t) < 1$

where $\boldsymbol{X}_t \subseteq \boldsymbol{X}$ is a set of confounders that isolate dependence between covariates and the treatment. The assumption of $T-$ignorability implies that conditional on $\boldsymbol{X}_t$, the treatment is independent of potential outcomes, hence the change in observed outcomes between treatment and control groups is only attributed to the treatment. The assumption of $T-$overlap guarantees that there is a sufficient number of individuals with characteristics $\boldsymbol{X}_t = \boldsymbol{x}_t$ in both groups. Given these two assumptions, the causal relationship between the treatment and the outcome can be identified and an unbiased estimate can be derived. Three classes of methods can be used to derive estimates of the ATE under these assumptions: the G-computation method, the IPW method, and the DR method. Since the treatment is randomized in (sub-)population of an RCT, these assumptions hold given an empty set in (sub-)population, and the simple difference in means between groups in (sub-)population is an unbiased estimate of the ATE of (sub-)population. See appendix C for more detailed descriptions. In practice, all outcome predictors can be adjusted in these methods because $\boldsymbol{X}$ is a sufficient set of measured confounders and may improve the precision of estimates (Chatton *et al.* 2020).

3.2. We can use estimates derived from the RCT to validate estimates from an observational dataset

Once we have unbiased estimates of the ATE obtained from an RCT dataset, how to use these estimates to validate estimates obtained from an observational dataset? In this section, we introduce assumptions and methods that allow for the validation. We assume an RCT dataset $\mathcal{S}^{rct}$ and an observational dataset $\mathcal{S}^{obs}$ are drawn from the same target population $\mathcal{P}_{\boldsymbol{\theta}}$; $\mathcal{S}^{rct}$ is a simple random sample from $\mathcal{P}_{\boldsymbol{\theta}}$ while $\mathcal{S}^{obs}$ is drawn from $\mathcal{P}_{\boldsymbol{\theta}}$ via a sampling mechanism. Let $S \in \{0, 1\}$ denote a binary selection indicator where 1 and 0 denote selection

to $\mathcal{S}^{rct}$ and $\mathcal{S}^{obs}$. Analogous to assumptions and methods in section 3.1, we can use similar assumptions of the sampling mechanism to allow for the comparison of estimates between $\mathcal{S}^{rct}$ and $\mathcal{S}^{obs}$:

Assumption 3 S-ignorability: $Y(1), Y(0) \perp\!\!\!\perp S \mid \boldsymbol{X}_s$

Assumption 4 S-overlap: $0 < P(S = 1 \mid \boldsymbol{X}_s) < 1$

Assumption 3 demonstrates that conditioning on $\boldsymbol{X}_s \subseteq \boldsymbol{X}$, potential outcomes are exchangeable between samples. Assumption 4 guarantees that there is a sufficient number of individuals with characteristics $\boldsymbol{X}_s = \boldsymbol{x}_s$ in both samples. Given these two assumptions, within a subpopulation $\boldsymbol{X}_s = \boldsymbol{x}_s$, there is no unobserved covariate varying between samples, and hence estimates of the ATE conditioning on $\boldsymbol{X}_s$ are comparable.

Given these two assumptions, we can use weighting methods to adjust for the sampling mechanism of $\mathcal{S}^{obs}$. These methods aim to balance $\boldsymbol{X}_s$ between samples. Three weighting methods are provided: 1) inverse selection probability weighting (ISW); 2) exact matching; 3) sub-classification based on strata of the selection probability of $\mathcal{S}^{rct}$. In general, all weighting methods require estimation of either a selection probability or density of $\boldsymbol{X}_s$. See appendix D for an elaboration of the weighting methods in **RCTrep**. In practice, only covariates that are predictive of treatment effects *and* the sample selection can lead to the discrepancy of treatment effects between samples while adjusting other covariates may inflate the variance of weighted estimates (Egami and Hartman 2021; Dahabreh *et al.* 2020).

3.3. Putting all together

Given above four assumptions, we can replace $\hat{p}(\boldsymbol{x})\hat{\tau}(\boldsymbol{x})$ in Equation 2 with $\hat{w}(\boldsymbol{x}_s)\hat{\tau}(\boldsymbol{x})$, and replace τ with $\hat{\tau}_0^{\mathcal{S}^{rct}}$, where $\hat{\tau}_0^{\mathcal{S}^{rct}}$ is an unbiased estimate of the ATE of $\mathcal{P}_\theta$ obtained from the estimator $\hat{\tau}_0$, $\hat{\tau}_0$ is the simple difference in sample means of outcomes between groups, and $\hat{\tau}(\boldsymbol{x})$ is an estimate of the ATE of a subpopulation with $\boldsymbol{X} = \boldsymbol{x}$ in $\mathcal{S}^{obs}$. The proposed evaluation metric is as follows:

$$\mathbb{L}\left(\hat{\tau}_0^{\mathcal{S}^{rct}}, \sum_{i \in \mathcal{S}^{obs}} \hat{w}(\boldsymbol{x}_{si})\hat{\tau}(\boldsymbol{x}_i)\right), \; s.t. \;\; \hat{p}(\boldsymbol{x}_s) = \hat{q}(\boldsymbol{x}_s)\hat{w}(\boldsymbol{x}_s), \sum_{i \in \mathcal{S}^{obs}} \hat{w}(\boldsymbol{x}_{si}) = 1 \tag{3}$$

where $\hat{w}(\boldsymbol{x}_{si})$ is the weight for individual $i \in \mathcal{S}^{obs}$, $\hat{w}(\boldsymbol{x}_s) = \sum_{\boldsymbol{x}_{si} = \boldsymbol{x}_s} \hat{w}(\boldsymbol{x}_{si})$ is the weight for a subpopulation with $\boldsymbol{X}_s = \boldsymbol{x}_s$ in $\mathcal{S}^{obs}$, $\boldsymbol{x}_s$ in weighted $\mathcal{S}^{obs}$ and $\mathcal{S}^{rct}$ are approximately equally distributed. A variety of distance measurements can be applied to $\mathbb{L}$. We also validate estimates on subsets of the target population to quantify the ability of $\hat{\tau}(\boldsymbol{X})$ to capture the variation in treatment effects across subpopulations. The validation on subpopulation levels can help us evaluate the flexibility of $\hat{\tau}(\boldsymbol{X})$. In the following, we will move from math to code, we will first have an overview of the package **RCTrep**, and then demonstrate the usage of **RCTrep**.

4. Overview of software

The current section introduces the **RCTrep** implementation and core classes. The section first presents an overview of core classes that form the building blocks of **RCTrep** and offers an overview of the implementation of **RCTrep** using these core classes. Then the section provides

a further introduction to these core classes and core functions. In the next section, we provide the basic structure of **RCTrep** and relations between each class.

4.1. Implementation

An overview of the implementation of **RCTrep** is provided in Figure 2. The figure demonstrates the role of three core classes in the implementation. The three classes are:

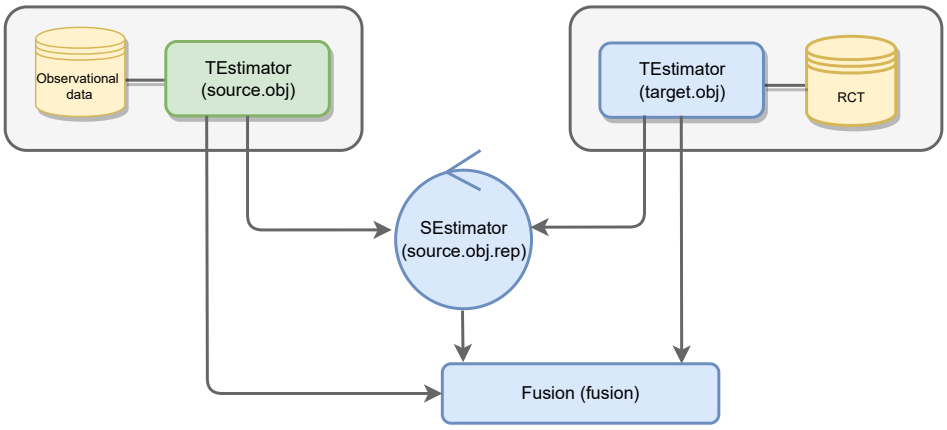


Figure 2: Diagram of **RCTrep** basic structure.

1. `TEstimator`: R6 class `TEstimator` is the parent class of all **RCTrep** `TEstimator` subclasses. It estimates the ATE of a population and subpopulations; it diagnoses the T-overlap assumption, and diagnoses the T-ignorability assumption depending on an instantiated class, e.g., it diagnoses model assumptions for the `G_computation` subclass and diagnoses the distance of confounders between groups for the `IPW` subclass. **RCTrep** provides `TEstimator_wrapper()` to generate an object of this class. See Table 2 for more detailed descriptions of input arguments in this function.

2. `SEstimator`: R6 class `SEstimator` is the parent class of all **RCTrep** `SEstimator` subclasses. The class integrates data from `source.obj` and `target.obj`, and regards data in `target.obj` as a simple random sample from a target population. It computes weights for `source.obj`, so that the weighted covariates in `source.obj` and these covariates in `target.obj` are balanced. It diagnoses the S-overlap assumption and diagnoses the S-ignorability assumption by measuring the distance of weighted covariates in `source.obj` and `target.obj`. **RCTrep** provides `SEstimator_wrapper()` to generate an object of this class. See Table 3 for more detailed descriptions of input arguments in this function.

3. `Fusion`: R6 class `Fusion` integrates estimates from objects of the class `TEstimator` and objects of the class `SEstimator`, computes evaluation metrics on population and subpopulation levels, and ranks estimates accordingly. The number of objects of the class `TEstimator` or `SEstimator` passed to its initialize function is not limited.

A main loop that relates one to one to the implementation is illustrated as follows:

1. users call `TEstimator_wrapper()` to initialize a `TEstimator` subclass for an observational dataset as `source.obj` and to initialize a `TEstimator` subclass for an RCT dataset as `target.obj`. These objects fit a model for the treatment or the outcome conditional on specified covariates, and estimate the ATE of population and subpopulations. The RCT data is regarded as a simple random sample from a target population;

2. users call `SEstimator_wrapper()` to initialize a `SEstimator` subclass as `source.obj.rep` by assigning `source.obj` and `target.obj` to the function. `source.obj.rep` estimates weights using specified covariates in `source.obj` and `target.obj`;

3. users call `source.obj.rep$EstimateRep()`, specifying two arguments `stratification` and `stratification_joint` to the function. The function estimates the weighted ATE of population and subpopulations stratified by levels of individual (`stratification_joint=FALSE`) or joint (`stratification_joint=TRUE`) covariates specified in `stratification`.

4. users initialize a `Fusion` class as `fusion` by assigning `source.obj`, `target.obj`, and `source.obj.rep` to its initialize function. `fusion` aggregates, ranks, plots, and prints estimates of the ATE of population and subpopulations. The object validates estimates of the ATE of the target population and subpopulations by calling `fusion$evaluate()`, prints evaluation metrics on population and subpopulation levels, and ranks these estimates according to the pseudo mean squared error.

5. (Optional) Then repeat step 3) and step 4) to validate estimates on subsets of the target population selected by different `stratification` and `stratification_joint`.

We provide an overview of the basic usage in section 5 where four main steps to validate estimates of the ATE using **RCTrep** are summarized. For more implementation details and infrastructure of design, see Appendix F.

4.2. Core classes

RCTrep provides two core classes, i.e., `TEstimator` and `SEstimator`, which are responsible for adjusting for the treatment assignment mechanism and the sampling mechanism, respectively. **RCTrep** offers four main subclasses of `TEstimator` and three main subclasses of `SEstimator`. The four subclasses of `TEstimator` are `Crude`, `G_computation`, `IPW`, and `DR`. The three subclasses of `SEstimator` are `SEexact`, `SEisw`, and `SEsubclass`. The description of key public attributes and key public methods of `TEstimator` and `SEstimator` are provided in Table 4. Note that input arguments of functions listed in Table 4 are `stratification` and `stratification_joint` with default values `private$outcome_predictors` and `TRUE`, respectively. By specifying these two arguments, these functions in Table 4 get outputs of subpopulations stratified by levels of covariates in `stratification`. More elaboration of these core classes is provided in Appendix B.

In case full data sets of `target.obj` and `source.obj` are not allowed to share to estimate weights, **RCTrep** provides a subclass `TEstimator_pp` and a subclass `SEstimator_pp`. The `TEstimator_wrapper()` returns an object of the class `TEstimator_pp` when the input argument `data.public=FALSE` is indicated. `SEstimator_wrapper()` returns an object of the class `SEstimator_pp` when the classes of input arguments are `TEstimator_pp`. The public

Arguments	Description	Default
`Estimator`	A character specifying a method for the ATE estimation. Allowable options are `"G_computation"`, `"IPW"`, `"DR"`.	-
`vars_name`	A list with three named characters, i.e., `outcome_predictors`, `treatment_name`, and `outcome_name`, which specifies covariate names of outcome predictors, the treatment, and the outcome.	-
`data`	A data.frame with n rows and p columns, each row contains covariates in `vars_name`. **RCTrep** supports the binary treatment and the binary/continuous outcome.	-
`name`	A character specifying a name of an returned object	`NULL`
`outcome_method`	A character specifying a method for modeling the outcome when `Estimator` is set to `"G_computation"` or `"DR"`. For more available methods, see a model list of the function `train()` in the R package **caret**	`"glm"`
`treatment_method`	A character specifying a method for modeling the propensity score when `Estimator` is set to `"IPW"` or `"DR"`. For more available methods, see a model list of the function `train()` in the R package **caret**	`"glm"`
`two_models`	Logical value indicating whether the outcome should be modeled separately when `Estimator` is set to `"DR"`	`FALSE`
`outcome_formula`	A formula specifying an outcome regression model when `Estimator` is set to `"G_computation"` or `"DR"`	`NULL`
`treatment_formula`	A formula specifying a propensity score model when `Estimator` is set to `"IPW"` or `"DR"`	`NULL`
`data.public`	Logical value indicating whether the full dataset `data` should be a public attribute of a returned object. If `FALSE`, the function returns an object of class `TEstimator_pp`	`TRUE`
`is.Trial`	Logical value indicating whether `data` is an RCT dataset	`FALSE`
`strata_cut`	A list each of a component is a named list with two named vectors. The name of a list is a covariate name and the names of two vectors are `breaks` and `labels`. `strata_cut` calls the `cut` function to divide the range of the value of the covariate into intervals based on `break` and code the value according to `label`.	NULL
`...`	A number of additional arguments for fitting a model specified in `outcome_method` or `treatment_method`. See allowable arguments in the function `train()` in the R package **caret**, or `pbart` and `wbart` in the R package **BART**	-

Table 2: Descriptions of the input argument of the function `TEstimator_wrapper()`.

Arguments	Description	Default
`Estimator`	A character specifying a method for estimating weights. Allowable options are `"Exact"`, `"Subclass"`, and `"ISW"`.	-
`target.obj`	An object of the class `TEstimator` of which `estimates` are unbiased estimates	-
`source.obj`	An object of the class `TEstimator` of which `estimates` are to validate	-
`selection_predictors`	A vector of characters specifying covariate names for weighting	-
`method`	A character specifying a method for estimating the selection probability. See a model list of the function `train()` in the R package **caret**, and options for `distance` argument of the function `matchit()` in the R package **MatchIt** package.	`'glm'`
`sampling_formula`	A formula specifying a model of the selection probability	NULL
`...`	A number of additional arguments for fitting a model specified in `method` when `Estimator` is set to `"ISW"`. See allowable arguments of the function `train()` in the R package **caret**	-

Table 3: Descriptions of the input arguments of the function `SEstimator_wrapper()`.

attributes `data` of objects of the class `TEstimator_pp` are the aggregate data of subpopulations. An object of the class `SEstimator_pp` estimates weights based on the aggregate data of objects of the class `TEstimator_pp`. See Example 2 in section 6 for the usage of aggregate data for the validation.

RCTrep provides a subclass `TEstimator_Synthetic` of `TEstimator`. The subclass is to initialize an object using a synthetic dataset. `GenerateSyntheticData()` generates a synthetic dataset given marginal distributions of covariates and pair-wise correlations between these covariates. The function estimates the joint distribution of these covariates and generates a full dataset accordingly. See Example 3 in section 6 for more details.

5. Basic usage

In the current section, we demonstrate a four-step workflow to validate estimates of the ATE using **RCTrep**: set-selection, estimation, diagnosis, and validation. We demonstrate these four steps using an example, and we integrate relevant results generated from these four steps into a dashboard. In the following, we introduce the first step.

5.1. Step 1: Set-selection

In the set-selection step, we select two covariates sets: 1) $\boldsymbol{X}$ `outcome_predictors`, a set of covariates used to adjust for the treatment assignment mechanism; 2) $\boldsymbol{X}_s$ `selection_predictors`, a set of covariates used to adjust for the sampling mechanism. By default, `outcome_predictors`

Attributes/Methods	Description
Class `TEstimator`	
`estimates`	A list containing two elements, i.e., a data frame named `ATE` and a data frame named `CATE`
`get_CATE()`	Print a data frame of estimates of the CATE
`plot_CATE()`	Plot estimates of the CATE
`diagnosis_t_ignorability()`	Plot diagnosis results of the T-ignorability assumption
`diagnosis_t_overlap()`	Plot diagnosis results of the T-overlap assumption
`diagnosis_y_overlap()`	Plot the count of binary outcomes in treatment and control groups; plot the distribution of continuous outcomes in treatment and control groups
`plot_y1_y0()`	Plot the predicted outcomes under the treatment and the control
Class `SEstimator`	
`estimates`	A list containing two elements, i.e., a data frame named `ATE` and a data frame named `CATE`
`EstimateRep()`	Estimate the weighted ATE of the population and subpopulations in `source.obj` and pass these results to the public attributes `estimates$ATE` and `estimates$CATE`
`diagnosis_s_ignorability()`	Plot diagnosis results of the S-ignorability assumption
`diagnosis_s_overlap()`	Plot diagnosis results of the S-overlap assumption

Table 4: Descriptions of core public attributes and core public methods of the class `TEstimator` and the class `SEstimator`.

and `selection_predictors` are the same. To reduce the variance of estimates of the weighted ATE, we assign a set of covariates that are predictive of treatment effects *and* the sample selection to `selection_predictors`. Since we don't know the true treatment assignment mechanism, we assign all pre-treatment covariates to `outcome_predictors`.

```
R> library("RCTrep")
R> source.data <- RCTrep::source.data
R> target.data <- RCTrep::target.data
R> vars_name <- list(outcome_predictors =
+    c("x1","x2","x3","x4","x5","x6"),
+    treatment_name = c("z"),
+    outcome_name = c("y")
+    )
R> selection_predictors <- c("x2","x6")
```

To demonstrate the set-selection, we present a causal structural diagram of the data generation mechanism of the data used throughout the paper in Figure 9. The figure presents predictors of the treatment, predictors of the outcome, and predictors of the selection. Although in practice the true causal structural diagram of a dataset is unknown, a such diagram can help us select `outcome_predictors` and `selection_predictors` easily. [1]

5.2. Step 2: Estimation

In the estimation step, two sub-steps are summarized, namely, the estimation of the ATE in `TEstimator`, and the estimation of the weighted ATE in `SEstimator`. In the first sub-step, we use one method to adjust for the treatment assignment mechanism, namely, `G_computation` method, and one method to derive unbiased estimates of the ATE in an RCT dataset, namely, `Crude` method. In the second sub-step, we use one method to adjust for the sampling mechanism, namely, `Exact` matching. We first estimate the ATE using an observational dataset.

Step 2.1: Estimation of the ATE

In this step, we estimate the ATE in `TEstimator`. We start out by instantiating objects of the class `TEstimator` using an observational dataset and an RCT dataset. We call `TEstimator_wrapper()` function to initialize objects `source.obj` and `target.obj` using these two datasets respectively:

```
R> source.obj <- TEstimator_wrapper(
+    Estimator = "G_computation",
+    data = source.data,
+    name = "RWD",
+    vars_name = vars_name,
+    outcome_method = "glm",
+    outcome_formula = y ~ x1 + x2 + x3 + z + z:x1 + z:x2 +z:x3+ z:x6,
+    data.public = TRUE
```

[1] Note that users could use related causal discovery packages to select these two sets. The software include but are not limited to, e.g., R packages **dosearch** (Tikka *et al.* 2021), **causaleffect** (Tikka and Karvanen 2017), and a web-based software **causalfusion** (Bareinboim and Pearl 2016).

```
+    )
R> target.obj <- TEstimator_wrapper(
+    Estimator = "Crude",
+    data = target.data,
+    name = "RCT",
+    vars_name = vars_name,
+    data.public = TRUE,
+    isTrial = TRUE
+    )
```

We specify the following arguments to instantiate `source.obj` and `target.obj`:

1. `Estimator`: specifying a method for estimating the ATE. `TEstimator_wrapper()` will initialize a `TEstimator` subclass according to the specified method. For instance, if `Estimator="G_computation"`, `TEstimator_wrapper()` initializes a subclass `G_computation` and returns the initialized object;

2. `data`: a `data.frame` with n rows and p columns, each row contains covariates indicated in `vars_name`;

3. `name`: a character indicating an object name;

4. `vars_name`: a list containing three vectors with the first vector `outcome_predictors` indicating the covariate names of outcome predictors, the second vector `treatment_name` indicating the covariate name of the treatment, and the third vector `outcome_name` indicating the covariate name of the outcome.

Step 2.2: Estimation of the weighted ATE

In this step, we estimate the weighted ATE in `SEstimator`. We instantiate a `SEstimator` subclass `SEexact` as `source.obj.rep` by calling the function `SEstimator_wrapper()`:

```
R> source.obj.rep <- SEstimator_wrapper(Estimator = "Exact",
+    target.obj = target.obj,
+    source.obj = source.obj,
+    selection_predictors =
+    selection_predictors)
R> source.obj.rep$EstimateRep(stratification = c("x1","x3","x5"),TRUE)
```

The arguments list for the function `SEstimator_wrapper` is:

1. `Estimator`: a character indicating a method for estimating weights. The wrapper function initializes a `SEstimator` subclass accordingly;

2. `target.obj` and `source.obj`: `target.obj` indicates an object whose data is regarded as a simple random sample of a target population and estimates of the ATE are regarded as unbiased estimates of the truth; `source.obj` indicates an object whose estimates of the ATE are to validate.

3. `selection_predictors`: a character vector indicating covariate names of sample selection predictors of the observational dataset; the weighted joint distribution of these covariates in `source.obj` should be approximately equally distributed to that in `target.obj`.

Then we call `EstimateRep()` - a core function of the instantiated object `source.obj.rep`. The function is to estimate the weighted ATE of the target population and subpopulations using the observational dataset. The weighted distribution of `selection_predictors` in `source.obj` and the distribution of `selection_predictors` in `target.obj` should be balanced. Two optional arguments for the function `EstimateRep()` are specified:

1. `stratification`: a character vector indicating covariate names. `EstimateRep()` estimates the weighted ATE of subpopulations. The subpopulations are selected according to levels of covariates in `stratification`; the default value of `stratification` is `selection_predictors`;

2. `stratification_joint`: a logical value, if `TRUE`, then subsets are selected by levels of joint covariates in `stratification`; otherwise, subsets are selected by levels of individual covariates in `stratification`.

5.3. Step 3: Diagnosis

On completion of all class instantiations, we need to diagnose assumptions in the object `source.obj` of the class `TEstimator`, and we need to diagnose assumptions in the object `source.obj.rep` of the class `SEstimator`:

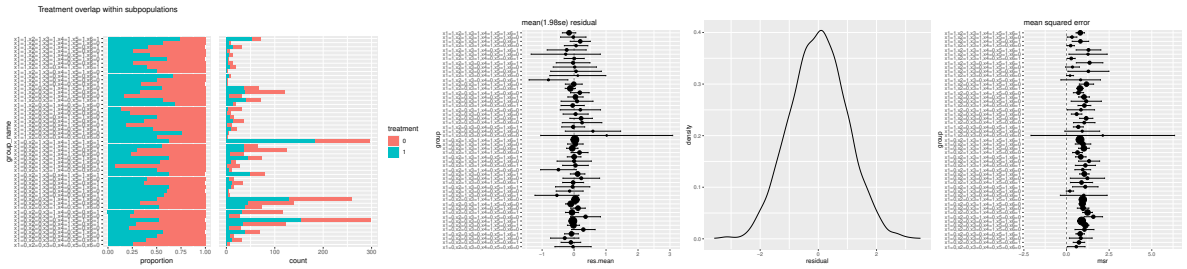
```
R> source.obj$diagnosis_t_overlap()
R> source.obj$diagnosis_t_ignorability()
R> source.obj.rep$diagnosis_s_overlap()
R> source.obj.rep$diagnosis_s_ignorability()
```

We call the above four lines to diagnose four assumptions, and the results show that:

1. Diagnosis of the T-overlap assumption: `source.obj` calls `diagnosis_t_overlap()`, and the result is presented in Figure 3 (a). The figure presents the proportion and the count of individuals receiving $T = 1$ and $T = 0$ within subpopulations stratified by `outcome_predictors`. The results show that there are sufficient individuals receiving the treatment and the control within the subpopulations.

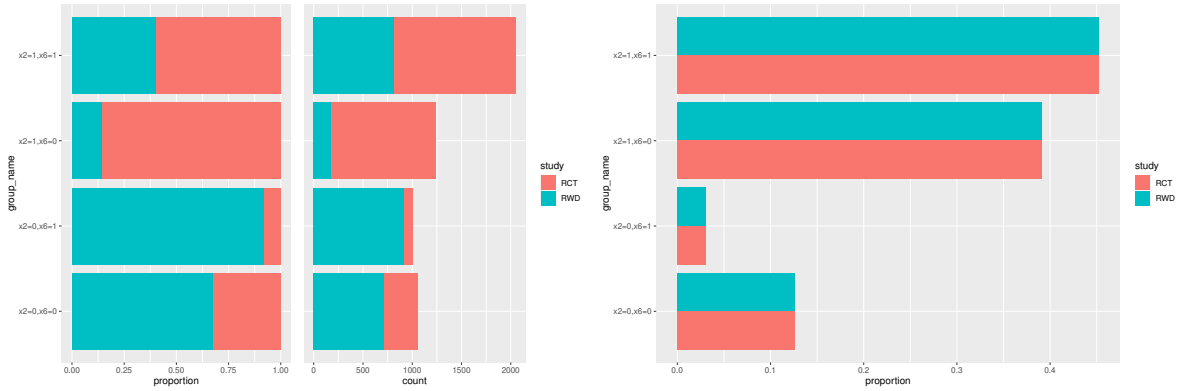
2. Diagnosis of the T-ignorability assupmtion: `source.obj` calls `diagnosis_t_ignorability()`, and the results are presented in Figure 3 (b). Since the class of `source.obj` is `G_computation`, the assumption of T-ignorability for the G-computation method indicates the assumption of no omitted variable bias in a regression model. Thus **RCTrep** diagnoses the T-ignorability assumption using the following three metrics:

   (a) residual mean ($\pm$ 1.98 standard error) of subpopulations stratified by `outcome_predictors`, which is presented in the left plot in Figure 3 (b). The result shows that means of residuals of subpopulations are all very close to zero;



(a) t-overlap in `source.obj`.

(b) t-ignorability in `source.obj`.



(c) s-overlap in `source.obj.rep`.

(d) s-ignorability diagnosis in `source.obj.rep`.

Figure 3: Diagnosis of assumptions in two objects.

(b) distribution of overall residuals, which is presented in the middle plot in Figure 3 (b). The result shows that the residual follows a standard normal distribution;

(c) mean squared error ($\pm$ 1.98 standard error) of subpopulations stratified by `outcome_predictors`, which is presented in the right plot in Figure 3 (b). The result shows that the mean squared error of each subpopulation is close to 1.

Overall, since the error term of the true data generation mechanism of the data in the example follows a standard normal distribution, the diagnosis results imply that the T-ignorability assumption plausibly holds. Thus the estimate of the ATE in `source.obj` is not biased. In addition, since the true variance of the error term is 1, the normal distribution of the residual (the middle plot in Figure 3 (b)) and the seemingly constant (i.e., 1) mean squared error over subpopulations (the right plot Figure 3 (b)) may imply that no other covariate can explain the residual variation. Diagnosis of the T-ignorability assumption depends on the class of `source.obj`. In case the class is `IPW`, an instantiated object diagnoses the assumption by presenting the inverse propensity score weighted distribution of `outcome_predictors` between treatment and control groups.

3. Diagnosis of the S-overlap assumption: `source.obj.rep` calls `diagnosis_s_overlap()`, and the results are presented in Figure 3 (c). The figure presents the proportion and the count of individuals in the observational dataset and the RCT dataset within combined subpopulations stratified by `selection_predictors` and the results show that there are sufficient individuals in the two samples.

4. Diagnosis of the S-ignorability assumption: `source.obj.rep` calls `diagnosis_s_ignorability()`, and the result is presented in Figure 3 (d). The figure presents the weighted distribution of `selection_predictors` in the observational dataset and the RCT dataset, indicating that `outcome_predictors` are balanced between the two samples and hence the sampling mechanism is properly adjusted.

In general, diagnosis of these four assumptions can help us understand the possible sources that may lead to a discrepancy of estimates between `source.obj.rep` and `target.obj`. For instance, near violation of the T-overlap assumption can lead to a high variance of estimates in the class `IPW` or a high bias of estimates in the class `G_computation`, and near violation of the S-overlap assumption can also lead to a high variance of weighted estimates in the class `SEstimator`.

5.4. Step 4: Validation

Lastly, we compute the evaluation metric in Equation 3 on population and subpopulation levels. We initialize a class `Fusion` as an object `fusion` and assign `source.obj`, `target.obj`, and `source.obj.rep` to `fusion`. `fusion` combines estimates from these objects and validates estimates of the ATE of the target population and subpopulations. Subsets are selected according to `stratification` and `stratification_joint` specified in `source.obj.rep$EstimateRep()`. `fusion` validates estimates in `source.obj` and `source.obj.rep` using four metrics, i.e., pseudo mean squared error (`mse`), length of confidence interval (`len_ci`), estimate agreement (`agg.est`), and regulatory agreement (`agg.reg`) (Franklin *et al.* 2020).

```
R> fusion <- Fusion$new(target.obj,
+    source.obj,
+    source.obj.rep)
R> fusion$evaluate()
```

```
# A tibble: 18 × 7
# Groups:   group_name [9]
   group_name     estimator                  size      mse len_ci agg.est agg.reg
   <chr>          <chr>                     <dbl>    <dbl>  <dbl> <lgl>   <lgl>
 1 pop            G_computation/glm/Exact/2  2622    0.038  0.92  TRUE    TRUE
 2 pop            G_computation/glm          2622  666.     0.239 FALSE   TRUE
 3 x1=0,x3=0,x5=0 G_computation/glm/Exact/2   230    0.197  4.03  TRUE    TRUE
 4 x1=0,x3=0,x5=0 G_computation/glm           230 1412.     0.625 FALSE   TRUE
 5 x1=0,x3=0,x5=1 G_computation/glm/Exact/2   496    4.82   4.69  TRUE    TRUE
 6 x1=0,x3=0,x5=1 G_computation/glm           496 1090.     0.444 FALSE   TRUE
 7 x1=0,x3=1,x5=0 G_computation/glm/Exact/2   481    0.091  2.49  TRUE    TRUE
 8 x1=0,x3=1,x5=0 G_computation/glm           481  642.     0.577 FALSE   TRUE
 9 x1=0,x3=1,x5=1 G_computation/glm           784   42.3    0.484 TRUE    TRUE
10 x1=0,x3=1,x5=1 G_computation/glm/Exact/2   784   66.1    1.68  TRUE    TRUE
11 x1=1,x3=0,x5=0 G_computation/glm/Exact/2    63    0.08   8.71  TRUE    TRUE
12 x1=1,x3=0,x5=0 G_computation/glm            63 1293.     1.30  FALSE   TRUE
13 x1=1,x3=0,x5=1 G_computation/glm/Exact/2    66    3.69   7.75  TRUE    TRUE
14 x1=1,x3=0,x5=1 G_computation/glm            66  226.     1.51  FALSE   TRUE
```

```
15 x1=1,x3=1,x5=0 G_computation/glm/Exact/2   246    6.35   4.18  TRUE    TRUE
16 x1=1,x3=1,x5=0 G_computation/glm           246 1285.     0.636 FALSE   TRUE
17 x1=1,x3=1,x5=1 G_computation/glm/Exact/2   256    6.04   5.34  TRUE    TRUE
18 x1=1,x3=1,x5=1 G_computation/glm           256  575.     0.788 FALSE   TRUE
```

```
R> fusion$plot()
```

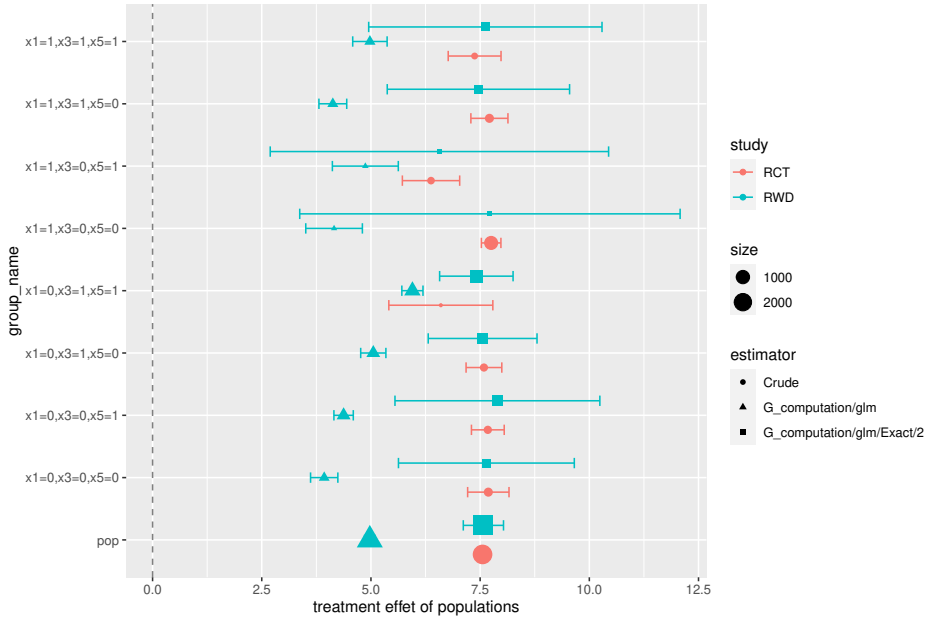


Figure 4: Results for validation of multiple estimates.

The result is presented in Figure 4, where `/2` indicates the number of covariates in `selection_predictors`. The result shows that

1. After adjusting for the treatment assignment mechanism and the sampling mechanism, point estimates obtained from the observational dataset (indicated by `G_computation/glm/Exact/2`) are very close to point estimates obtained from the RCT dataset (indicated by `Crude`), on both the population and the subpopulation levels. The result implies that the treatment assignment mechanism of the observational dataset is properly adjusted, and hence these estimates obtained from the observational data are valid.

2. The point estimates indicated by `G_computation` considerably differ from those indicated by `Crude`, implying that even though the treatment assignment mechanism of the observational dataset can be properly adjusted, there is a large difference in estimates of (sub-)populations between two datasets. Without considering the effect of the sampling mechanism on the difference in estimates, people may easily attribute the spurious difference to unmeasured confounders in the observational dataset, and question the validity of estimates obtained from the observational dataset.

3. The interval estimates of the weighted ATE of `G_computation/glm/Exact/2` (i.e., `len_ci` of `pop` = 0.92) is wider than those of `G_computation/glm` (`len_ci` of `pop`

=0.239), implying that weighting inflates the variance of weighted estimates. This result might be explained by the extreme imbalance of the proportion of subpopulations in the two datasets stratified by `selection_predictors`, as indicated in Figure 3 (c). The imbalance can lead to extreme weights, and hence inflates the variance of the weighted estimates.

4. The interval estimates of unweighted estimates as indicated by `G_computation/glm` vary across subpopulations, and may be influenced by multiple facts: 1) the sample size of subpopulations; 2) the imbalance of proportion of individuals in treatment and control groups within subpopulations; 3) the variance of an outcome predictor, and wide interval estimates of a subpopulation may indicate further stratification on the subpopulation or additional covariate adjustment to reduce the observed variation. The variation of covariates that are predictive of treatment effects amongst subpopulations can have impacts on interval estimates as well (Tipton 2021).

5.5. Easy visualization of results

RCTrep provides a dashboard that allows users to present all necessary results generated from these four steps and provides users with the flexibility to select subpopulation(s) for the validation. The dashboard can be launched by calling the function:

```
R> call_dashboard(source.obj = source.obj,
+    target.obj = target.obj,
+    source.obj.rep = source.obj.rep)
```

Once an interface is launched, users need to select covariates in checkboxes and click the "Go" buttons to generate related results. Figure 5 shows the dashboard and the generated results. The dashboard contains four panels, i.e., set-selection, estimation, diagnosis, and validation. Set-selection offers two sets of covariates used for adjusting for the treatment assignment mechanism and the sampling mechanism, and one additional set of covariates for selecting subpopulations; estimation provides point and interval estimates of the ATE of selected subpopulations; diagnosis provides diagnosis results of treatment- and sampling-related assumptions; validation presents and compares point and interval estimates of population and selected subpopulations. In the following, we introduce the basic workflow of the dashboard and the usage of each panel respectively:

1. The set-selection panel provides three boxes:

   - Outcome predictors: a set of outcome predictors used for adjusting the treatment assignment mechanism; by default, the selected covariates are `outcome_predictors` defined in `source.obj`; by clicking "Go" the boxes named T-overlap and T-ignorability will present the diagnosis results of the T-overlap assumption and the T-ignorability assumption, respectively;
   - Selection predictors: a set of sample selection predictors used for adjusting the sampling mechanism; by default, the selected covariates are `selection_predictors` defined in `source.obj.rep`; by clicking "Go" the boxes named S-overlap and S-ignorability will present diagnosis results of the S-overlap assumption and the S-ignorability assumption, respectively;

- Stratification: a set of all pre-treatment covariates. The box provides covariates to select subpopulations; no default values are selected. By clicking "Go" the estimation panel will present estimates of the ATE of the selected subpopulations, and the validation panel will present the validation results of the selected subpopulations. In Figure 5, we select `x1,x3,x4` for simplicity.

2. The estimation panel plots estimates of the ATE and estimates of potential outcomes of the selected subpopulations, and prints numeric values accordingly. Additional values `pt` and `py`, denoting the proportion of the treatment and the proportion of the positive outcome for binary outcomes (or mean of outcomes for continuous outcomes), are also printed.

3. The diagnosis panel diagnoses the T-overlap and the T-ignorability assumptions; the panel diagnoses S-overlap and S-ignorability assumptions.

4. The validation panel aggregates and plots estimates of the ATE of the target population and the selected subpopulations in `target.obj`, `source.obj` and `source.obj.rep`, and prints numeric results of the evaluation metrics.

6. Additional examples

In this section, we demonstrate three examples for validating estimates of the ATE using **RCTrep**. The first example demonstrates the validation of estimates derived from different settings. The second example demonstrates the validation in case only subpopulation-level data are available. The third example demonstrates the validation using synthetic RCT data. In the following, we first introduce using **RCTrep** to validate estimates from different settings.

6.1. Example 1: Validation at scale

In the following, we demonstrate how to validate estimates derived from different settings using **RCTrep**. We instantiated multiple objects, and combined these objects in one object of the class `Fusion`. Estimates of the ATE are compared to the unbiased estimates and results are shown in Figure 6:

```
R> library("RCTrep")
R> set.seed(123)
R> source.data <- RCTrep::source.data
R> target.data <- RCTrep::target.data
R> vars_name <- list(outcome_predictors =
+    c("x1","x2","x3","x4","x5","x6"),
+    treatment_name = c('z'),
+    outcome_name = c('y')
+    )
R> source.obj.gc <- TEstimator_wrapper(
+    Estimator = "G_computation",
+    data = source.data,
+    name = "RWD",
```

Figure 5: **RCTrep** dashboard to interactively visualize all results generated from the set-selection, estimation, diagnosis, and validation steps.

```
+    vars_name = vars_name,
+    outcome_method = "psBART",
+    data.public = TRUE
+    )
R> source.obj.ipw <- TEstimator_wrapper(
+    Estimator = "IPW",
+    data = source.data,
+    name = "RWD",
+    vars_name = vars_name,
+    treatment_method = "BART",
+    data.public = TRUE
+    )
R> source.obj.dr <- TEstimator_wrapper(
+    Estimator = "DR",
+    data = source.data,
+    name = "RWD",
+    vars_name = vars_name,
+    outcome_method = "BART",
+    treatment_method = "BART",
+    data.public = TRUE
+    )
R> target.obj <- TEstimator_wrapper(
+    Estimator = "Crude",
+    data = target.data,
+    name = "RCT",
+    vars_name = vars_name,
+    data.public = TRUE,
+    isTrial = TRUE
+    )
R> strata <- c("x1","x4")
R> selection_predictors <- c("x2","x6")
R> source.gc.exact <- SEstimator_wrapper(Estimator = "Exact",
+    target.obj = target.obj,
+    source.obj = source.obj.gc,
+    selection_predictors =
+    selection_predictors)
R> source.gc.exact$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> source.gc.isw <- SEstimator_wrapper(Estimator = "ISW",
+    target.obj = target.obj,
+    source.obj = source.obj.gc,
+    selection_predictors =
+    selection_predictors,
+    method = "glm")
R> source.gc.isw$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> source.gc.subclass <- SEstimator_wrapper(Estimator = "Subclass",
```

```
+    target.obj = target.obj,
+    source.obj = source.obj.gc,
+    selection_predictors =
+    selection_predictors)
R> source.gc.subclass$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> source.ipw.exact <- SEstimator_wrapper(Estimator = "Exact",
+    target.obj = target.obj,
+    source.obj = source.obj.ipw,
+    selection_predictors =
+    selection_predictors)
R> source.ipw.exact$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> source.ipw.isw <- SEstimator_wrapper(Estimator = "ISW",
+    target.obj = target.obj,
+    source.obj = source.obj.ipw,
+    selection_predictors =
+    selection_predictors,
+    method = "glm")
R> source.ipw.isw$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> source.ipw.subclass <- SEstimator_wrapper(Estimator = "Subclass",
+    target.obj = target.obj,
+    source.obj = source.obj.ipw,
+    selection_predictors =
+    selection_predictors)
R> source.ipw.subclass$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> source.dr.exact <- SEstimator_wrapper(Estimator = "Exact",
+    target.obj = target.obj,
+    source.obj = source.obj.dr,
+    selection_predictors =
+    selection_predictors)
R> source.dr.exact$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> source.dr.isw <- SEstimator_wrapper(Estimator = "ISW",
+    target.obj = target.obj,
+    source.obj = source.obj.dr,
+    selection_predictors =
+    selection_predictors,
+    method = "glm")
R> source.dr.isw$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> source.dr.subclass <- SEstimator_wrapper(Estimator = "Subclass",
+    target.obj = target.obj,
+    source.obj = source.obj.dr,
+    selection_predictors =
```

```
+    selection_predictors)
R> source.dr.subclass$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> fusion <- Fusion$new(target.obj,
+    source.gc.exact,
+    source.gc.isw,
+    source.gc.subclass,
+    source.ipw.exact,
+    source.ipw.isw,
+    source.ipw.subclass,
+    source.dr.exact,
+    source.dr.isw,
+    source.dr.subclass)
R> fusion$plot()
```

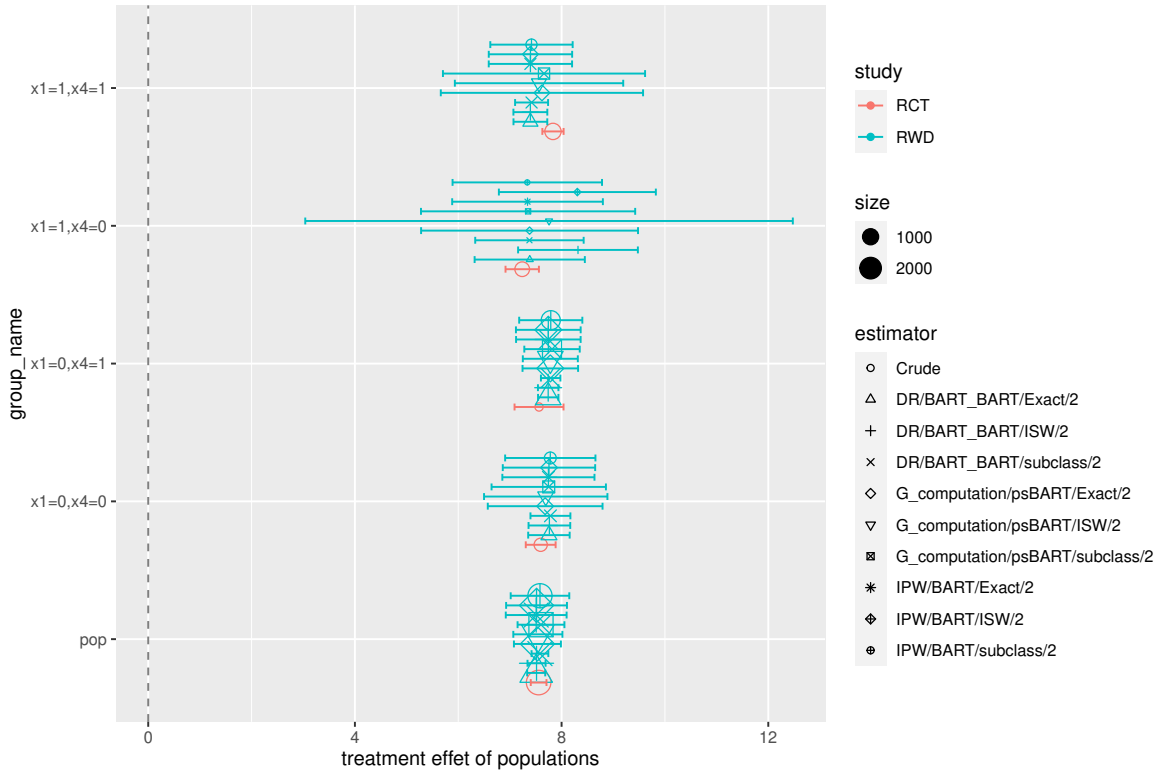


Figure 6: Comparisons of 9 estimates.

```
R> fusion$evaluate()

# A tibble: 45 × 7
# Groups:   group_name [5]
   group_name estimator                          size   mse len_ci agg.est agg.reg
   <chr>      <chr>                             <dbl> <dbl>  <dbl> <lgl>   <lgl>
 1 pop        G_computation/psBART/ISW/2         2622 0.021  0.95  TRUE    TRUE
```

```
 2 pop        G_computation/psBART/Exact/2     2622 0.053  0.91  TRUE    TRUE
 3 pop        DR/BART_BART/subclass/2          2622 0.066  0.324 TRUE    TRUE
 4 pop        IPW/BART/subclass/2              2622 0.07   1.13  TRUE    TRUE
 5 pop        IPW/BART/ISW/2                   2622 0.16   1.18  TRUE    TRUE
 6 pop        DR/BART_BART/ISW/2               2622 0.161  0.35  TRUE    TRUE
 7 pop        G_computation/psBART/subclass/2  2622 0.212  0.906 TRUE    TRUE
 8 pop        IPW/BART/Exact/2                 2622 0.219  1.18  TRUE    TRUE
 9 pop        DR/BART_BART/Exact/2             2622 0.22   0.35  TRUE    TRUE
10 x1=0,x4=0  G_computation/psBART/Exact/2      496 0.747  2.22  TRUE    TRUE
# i 35 more rows
# i Use `print(n = ...)` to see more rows
```

In general, the results show that the propensity-score adjusted `G_computation` indicated by `G_computation/psBART/` is the most accurate in terms of pseudo mean squared error, which is in line with results in existing literature (Chatton *et al.* 2020; Le Borgne *et al.* 2021; Loiseau *et al.* 2022; Dorie *et al.* 2019; Wendling *et al.* 2018; Hahn *et al.* 2020); `IPW` has the widest interval estimates, compared to `DR` and `G_computation`. More comparisons between model choices and adjustment sets can be implemented.

6.2. Example 2: Validation using aggregate data

RCTrep offers a solution to validating estimates of the ATE using aggregate data. We start out by instantiating an object `source.obj` using an observational dataset and an object `target.obj` using an RCT dataset [2]:

```
R> library("RCTrep")
R> source.data <- RCTrep::source.data
R> target.data <- RCTrep::target.data
R> vars_name <- list(outcome_predictors =
+    c("x1","x2","x3","x4","x5","x6"),
+    treatment_name = c('z'),
+    outcome_name = c('y')
+    )
R> selection_predictors <- c("x2","x6")
R> source.obj <- TEstimator_wrapper(
+    Estimator = "G_computation",
+    data = source.data,
+    vars_name = vars_name,
+    outcome_method = "glm",
+    outcome_form=y ~ x1 + x2 + x3 + z + z:x1 + z:x2 +z:x3+ z:x6,
+    name = "RWD",
+    data.public = FALSE
+    )
R> target.obj <- TEstimator_wrapper(
+    Estimator = "Crude",
```

[2] note that in Example 2, we have pre-processed data for instantiating two objects: the rows in `source.data` and `target.data` that have no match on the specified column `selection_predictors` are removed.

```
+    data = target.data,
+    vars_name = vars_name,
+    name = "RCT",
+    data.public = FALSE,
+    isTrial = TRUE
+    )
```

We specify `data.public=FALSE` to indicate that the full dataset is not allowed to output. `TEstimator_wrapper()` returns an object of the class `TEstimator_pp` of which the public field `data` is aggregate data of subpopulations stratified by levels of joint covariates in `outcome_predictors`:

```
R> print(head(source.obj$data), digits = 2)
```

```
  x1 x2 x3 x4 x5 x6 y1.hat y0.hat cate      se size   pt   py id
1  0  0  0  0  0  0    2.0  0.049  2.0 4.4e-09    8 0.25 0.51  1
2  0  0  0  0  0  1    3.1  0.049  3.1 8.0e-16   31 0.39 1.13  2
3  0  0  0  0  1  0    2.0  0.049  2.0 1.9e-08   16 0.50 0.75  3
4  0  0  0  0  1  1    3.1  0.049  3.1 1.1e-16   68 0.56 1.68  4
5  0  0  0  1  0  0    2.0  0.049  2.0 8.2e-17   29 0.21 0.75  5
6  0  0  0  1  0  1    3.1  0.049  3.1 2.4e-16  122 0.28 0.88  6
```

Then we instantiate an object `source.rep.obj` of the class `SEstimator_pp`, and we specify `stratification = strata` indicating subpopulations of which estimates of the ATE are to validate:

```
R> strata <- c("x1","x4")
R> source.rep.obj <- SEstimator_wrapper(Estimator = "Exact",
+    target.obj = target.obj,
+    source.obj = source.obj,
+    selection_predictors =
+    selection_predictors)
R> source.rep.obj$EstimateRep(stratification = strata,
+    stratification_joint = TRUE)
R> fusion <- Fusion$new(target.obj,
+    source.obj,
+    source.rep.obj)
R> fusion$plot()
```



```
R> fusion$evaluate()
```

```
# A tibble: 10 × 7
# Groups:   group_name [5]
   group_name estimator                   size      mse len_ci agg.est agg.reg
   <chr>      <chr>                      <dbl>    <dbl>  <dbl> <lgl>   <lgl>
 1 pop        G_computation/glm/Exact/2  2622    0.038  0.92  TRUE    TRUE
```

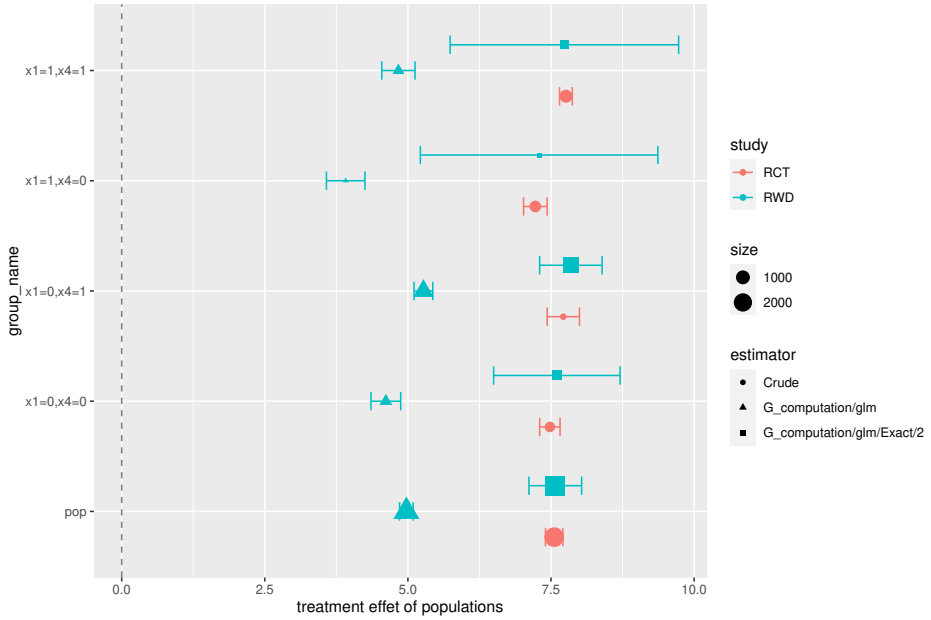


Figure 7: Validation results based on aggregate data of subpopulations in an observational dataset and an RCT dataset.

```
 2 pop        G_computation/glm          2622  666.     0.239 FALSE   TRUE
 3 x1=0,x4=0  G_computation/glm/Exact/2   496    1.50   2.21  TRUE    TRUE
 4 x1=0,x4=0  G_computation/glm           496  821.     0.519 FALSE   TRUE
 5 x1=0,x4=1  G_computation/glm/Exact/2  1495    1.73   1.09  TRUE    TRUE
 6 x1=0,x4=1  G_computation/glm          1495  598.     0.327 FALSE   TRUE
 7 x1=1,x4=0  G_computation/glm/Exact/2   193    0.428  4.15  TRUE    TRUE
 8 x1=1,x4=0  G_computation/glm           193 1098.     0.673 FALSE   TRUE
 9 x1=1,x4=1  G_computation/glm/Exact/2   438    0.076  3.99  TRUE    TRUE
10 x1=1,x4=1  G_computation/glm           438  857.     0.581 FALSE   TRUE
```

6.3. Example 3: Validation using synthetic RCT data

In Example 2 we demonstrate the validation approach using aggregate data from two datasets. However, in practice, we rarely have access to such data. In most cases, we only have aggregate data of each covariate and estimates of the ATE of subpopulations stratified by levels of these covariates individually. In Example 3 we demonstrate how to generate synthetic RCT data in this case using `GenerateSyntheticData()`. First, for a demonstrative purpose, we instantiate an object of the class `Crude` using an RCT dataset. We derive the marginal distributions of covariates `x1,x2,x3,x4,x5,x6` of the RCT data, and derive estimates of the ATE of subpopulations stratified by levels of these covariates individually:

```
R> library("dplyr")
R> library("gdata")
R> set.seed(123)
R> source.data <- RCTrep::source.data
```

```
R> target.data <- RCTrep::target.data
R> vars_name <- list(outcome_predictors =
+    c("x1","x2","x3","x4","x5","x6"),
+    treatment_name = c('z'),
+    outcome_name = c('y')
+    )
R> target.obj <- TEstimator_wrapper(
+    Estimator = "Crude",
+    data = target.data,
+    vars_name = vars_name,
+    name = "RCT",
+    data.public = FALSE,
+    isTrial = TRUE
+    )
R> vars_rct <- c("x1","x2","x3","x4","x5","x6")
R> RCT.estimates <- list(ATE_mean = target.obj$estimates$ATE$est,
+    ATE_se = target.obj$estimates$ATE$se,
+    CATE_mean_se = target.obj$get_CATE(vars_rct,FALSE))
```

Then we generate a synthetic RCT dataset `synthetic.data` by calling the **RCTrep** function `GenerateSyntheticData()`. In the function, we specify a marginal distribution of each covariate and pairwise correlations between these covariates. The function generates a synthetic dataset of the RCT accordingly:

```
R> emp.p1 <- mean(target.data$x1)
R> emp.p2 <- mean(target.data$x2)
R> emp.p3 <- mean(target.data$x3)
R> emp.p4 <- mean(target.data$x4)
R> emp.p5 <- mean(target.data$x5)
R> emp.p6 <- mean(target.data$x6)
R> t.d <- target.data[,vars_rct]
R> n <- dim(source.data)[1]
R> pw.cor <- gdata::upperTriangle(cor(t.d), diag = FALSE, byrow = TRUE)
R> synthetic.data <- RCTrep::GenerateSyntheticData(
+    margin_dis="bernoulli",
+    N = n,
+    margin = list(emp.p1, emp.p2, emp.p3, emp.p4, emp.p5, emp.p6),
+    var_name = vars_rct,
+    pw.cor = pw.cor)
R> head(synthetic.data)

  x1 x2 x3 x4 x5 x6
1  0  1  1  0  0  1
2  1  0  0  0  1  1
3  1  1  0  1  0  0
4  1  1  0  0  0  0
5  0  0  1  0  0  1
6  1  1  1  1  0  1
```

The rows in `source.data` and `synthetic.data` that have no match on the specified columns in `vars_rct` are removed. Then we instantiate `target.obj` of class `TEstimator_Synthetic` and `source.obj` of the class `G_computation`. For instantiation of an object `target.obj`, we initialize the public field `data` using `synthetic.data` and initialize the public field `estimates` using `RCT.estimates`. The weighted estimates of the ATE in `source.obj.rep` are compared to the unbiased estimates in `target.obj`, and the validation results are presented in Figure 8:

```
R> synthetic.data <- semi_join(synthetic.data, source.data, by = vars_rct)
R> source.data <- semi_join(source.data, synthetic.data, by = vars_rct)
R> target.obj <- RCTrep:::TEstimator_Synthetic$new(data = synthetic.data,
+    estimates=RCT.estimates,
+    vars_name = vars_name,
+    name = "RCT",
+    isTrial = TRUE,
+    data.public = TRUE)
R> source.obj <- TEstimator_wrapper(
+    Estimator = "G_computation",
+    data = source.data,
+    vars_name = vars_name,
+    outcome_method = "glm",
+    outcome_form=y ~ x1 + x2 + x3 + z + z:x1 + z:x2 +z:x3+ z:x6,
+    name = "RWD",
+    data.public = TRUE
+    )
R> source.rep.obj <- SEstimator_wrapper(Estimator="Exact",
+    target.obj=target.obj,
+    source.obj=source.obj,
+    selection_predictors=
+    c("x2","x6"))
R> source.rep.obj$EstimateRep(stratification = vars_rct,
+    stratification_joint = FALSE)
R> fusion <- Fusion$new(target.obj,
+    source.obj,
+    source.rep.obj)
R> fusion$plot()
```

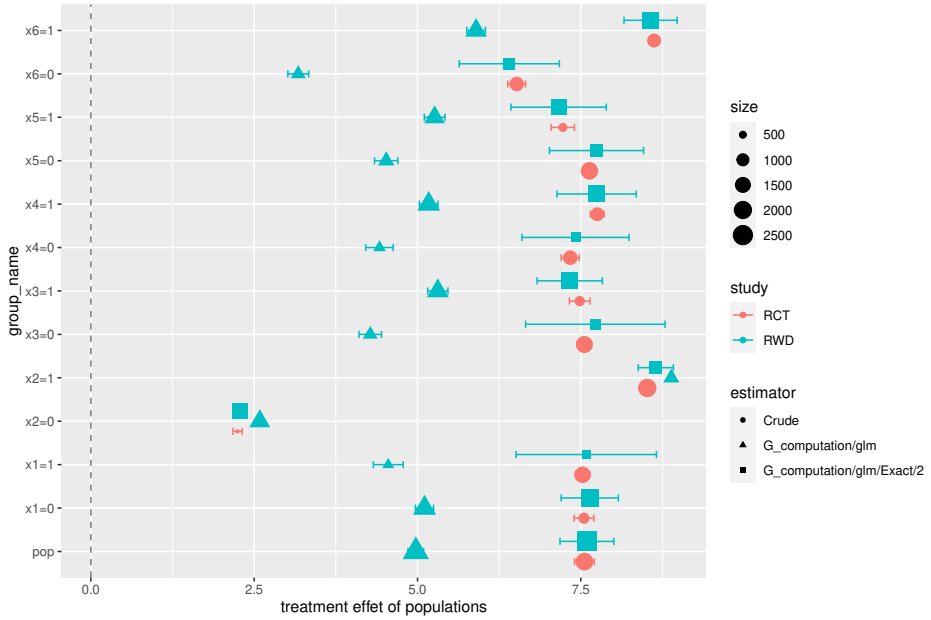


Figure 8: Validation results where the weights of `source.obj` are estimated based on the synthetic RCT data.

```
R> fusion$evaluate()

# A tibble: 26 × 7
# Groups:   group_name [13]
   group_name estimator                 size     mse len_ci agg.est agg.reg
   <chr>      <chr>                    <dbl>   <dbl>  <dbl> <lgl>   <lgl>
 1 pop        G_computation/glm/Exact/2  2622   0.139  0.824 TRUE    TRUE
 2 pop        G_computation/glm          2622 666.     0.239 FALSE   TRUE
 3 x1=0       G_computation/glm/Exact/2  1991   0.785  0.876 TRUE    TRUE
 4 x1=0       G_computation/glm          1991 596.     0.279 FALSE   TRUE
 5 x1=1       G_computation/glm/Exact/2   631   0.307  2.15  TRUE    TRUE
 6 x1=1       G_computation/glm           631 885.     0.458 FALSE   TRUE
 7 x2=0       G_computation/glm/Exact/2  1628   0.112  0.084 TRUE    TRUE
 8 x2=0       G_computation/glm          1628  11.5    0.053 FALSE   TRUE
 9 x2=1       G_computation/glm/Exact/2   994   1.70   0.538 FALSE   TRUE
10 x2=1       G_computation/glm           994  13.5    0.052 FALSE   TRUE
# i 16 more rows
# i Use `print(n = ...)` to see more rows
```

Results in Figure 8 show that even though we don't have individual-level RCT data, the weighted estimates of the ATE (indicated by `G_computation/glm/Exact/2`) can be closer to the unbiased estimates (indicated by `Crude`) compared to unweighted estimates (indicated by `G_computation/glm`). Hence we can still validate estimates of the ATE to some extent and obtain qualitative results, e.g., the direction of effects. Covariates that are predictive of the ATE *and* the sample selection (i.e., `x2,x6`), which can lead to a large discrepancy in estimates between samples, should be weighted.

7. Discussion

The package **RCTrep** aims to help researchers to validate estimates of the ATE of (sub-)populations obtained from an observational dataset in case individual-level or aggregate randomized controlled trial data is accessible. **RCTrep** provides three classes of methods for the estimation of the ATE and three classes of methods for the estimation of weights, and provides a variety of modeling choices for the outcome, the treatment, and the sample selection. **RCTrep** validates estimates on both population and subpopulation levels, providing a deeper insight into the performance of methods.

RCTrep highlights the importance of making RCT data more accessible in order to allow the validation of estimates of the ATE obtained from observational data. We recognize the irreplaceable role of RCT data in fueling the power of observational data to drive more personalized treatment. Further development can include 1) enrich methods for estimating the ATE in the class `TEstimator`, for instance, balancing-based methods via optimization (Chattopadhyay *et al.* 2020; Dong *et al.* 2020) and bayesian networks (Pearl 2009); 2) enrich methods for estimating weights in the class `SEstimator`; 3) additional options for the uncertainty quantification of the (weighted) ATE, for instance, the delta method (Oehlert 1992), the bootstrap resampling (Efron and Tibshirani 1994), the double bootstrap (Ackerman *et al.* 2021), and the parametric simulation-based method (Chatton *et al.* 2020); 4) different estimands of treatment effects and the corresponding weighted estimands can be provided, for instance, relative risk, risk ratio, odds ratio, etc. (Colnet *et al.* 2023).

References

Aalen OO, Farewell VT, De Angelis D, Day NE, Nöel Gill O (1997). "A Markov Model for HIV Disease Progression Including the Effect of HIV Diagnosis and Treatment: Application to AIDS Prediction in England and Wales." *Statistics in Medicine*, **16**(19), 2191–2210.

Ackerman B, Lesko CR, Siddique J, Susukida R, Stuart EA (2021). "Generalizing Randomized Trial Findings to a Target Population Using Complex Survey Population Data." *Statistics in Medicine*, **40**(5), 1101–1120.

Alaa A, Van Der Schaar M (2019). "Validating Causal Inference Models via Influence Functions." In *International Conference on Machine Learning*, pp. 191–201. PMLR.

Atan O, Jordon J, Van der Schaar M (2018). "Deep-Treat: Learning Optimal Personalized Treatments from Observational Data Using Neural Networks." In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Austin PC (2011). "An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies." *Multivariate behavioral research*, **46**(3), 399–424.

Bang H, Robins JM (2005). "Doubly Robust Estimation in Missing Data and Causal Inference Models." *Biometrics*, **61**(4), 962–973.

Bareinboim E, Pearl J (2016). "Causal Inference and the Data-Fusion Problem." *Proceedings of the National Academy of Sciences*, **113**(27), 7345–7352.

Bica I, Alaa AM, Lambert C, Van Der Schaar M (2021). "From Real-World Patient Data to Individualized Treatment Effects Using Machine Learning: Current and Future Methods to Address Underlying Challenges." *Clinical Pharmacology & Therapeutics*, **109**(1), 87–100.

Buchanan AL, Hudgens MG, Cole SR, Mollan KR, Sax PE, Daar ES, Adimora AA, Eron JJ, Mugavero MJ (2018). "Generalizing Evidence from Randomized Trials Using Inverse Probability of Sampling Weights." *Journal of the Royal Statistical Society Series A: Statistics in Society*, **181**(4), 1193–1209.

Chang W (2019). "**R6**: Encapsulated Classes with Reference Semantics." *R Package Version*, **2**(0).

Chatton A, Le Borgne F, Leyrat C, Gillaizeau F, Rousseau C, Barbin L, Laplaud D, Léger M, Giraudeau B, Foucher Y (2020). "G-Computation, Propensity Score-Based methods, and Targeted Maximum Likelihood Estimator for Causal Inference with Different Covariates Sets: A Comparative Simulation Study." *Scientific Reports*, **10**(1), 1–13.

Chattopadhyay A, Hase CH, Zubizarreta JR (2020). "Balancing vs Modeling Approaches to Weighting in Practice." *Statistics in Medicine*, **39**(24), 3227–3254.

Cheng L, Guo R, Moraffah R, Sheth P, Candan KS, Liu H (2022). "Evaluation Methods and Measures for Causal Learning Algorithms." *IEEE Transactions on Artificial Intelligence*.

Cinelli C, Pearl J (2021). "Generalizing Experimental Results by Leveraging Knowledge of Mechanisms." *European Journal of Epidemiology*, **36**(2), 149–164.

Colnet B, Josse J, Varoquaux G, Scornet E (2023). "Risk Ratio, Odds Ratio, Risk Difference... Which Causal Measure is Easier to Generalize?" *arXiv preprint arXiv:2303.16008*.

Colnet B, Mayer I, Chen G, Dieng A, Li R, Varoquaux G, Vert JP, Josse J, Yang S (2020). "Causal Inference Methods for Combining Randomized Trials and Observational Studies: A Review." *arXiv preprint arXiv:2011.08047*.

Dahabreh IJ, Robertson SE, Steingrimsson JA, Stuart EA, Hernan MA (2020). "Extending Inferences from a Randomized Trial to a New Target Population." *Statistics in Medicine*, **39**(14), 1999–2014.

Dong L, Yang S, Wang X, Zeng D, Cai J (2020). "Integrative Analysis of Randomized Clinical Trials with Real World Evidence Studies." *arXiv preprint arXiv:2003.01242*.

Dorie V, Hill J, Shalit U, Scott M, Cervone D (2019). "Automated Versus Do-It-Yourself Methods for Causal Inference: Lessons Learned from a Data Analysis Competition." *Statistical Science*, **34**(1), 43–68.

Efron B, Tibshirani RJ (1994). *An Introduction to the Bootstrap*. CRC press.

Egami N, Hartman E (2021). "Covariate Selection for Generalizing Experimental Results: Application to a Large-scale Development Program in Uganda." *Journal of the Royal Statistical Society Series A: Statistics in Society*.

Franklin JM, Pawar A, Martin D, Glynn RJ, Levenson M, Temple R, Schneeweiss S (2020). "Nonrandomized Real-World Evidence to Support Regulatory Decision making: Process for a Randomized Trial Replication Project." *Clinical Pharmacology & Therapeutics*, **107**(4), 817–826.

Franklin JM, Schneeweiss S, Polinski JM, Rassen JA (2014). "Plasmode Simulation for the Evaluation of Pharmacoepidemiologic Methods in Complex Healthcare Databases." *Computational Statistics and Data Analysis*, **72**, 219–226.

Franz M (2020). "**JustCause**: Comparing Methods for Causality Analysis in a Fair and Just Way." https://justcause.readthedocs.io/en/latest/#.

Funk MJ, Westreich D, Wiesen C, Stürmer T, Brookhart MA, Davidian M (2011). "Doubly Robust Estimation of Causal Effects." *American journal of epidemiology*, **173**(7), 761–767.

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

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

Hitsch GJ, Misra S (2018). "Heterogeneous Treatment Effects and Optimal Targeting Policy Evaluation." *Available at SSRN 3111957*.

Ho DE, Imai K, King G, Stuart EA (2011). "**MatchIt**: Nonparametric Preprocessing for Parametric Causal Inference." *Journal of Statistical Software*, **42**(8), 1–28. doi:10.18637/jss.v042.i08.

Hosmer Jr DW, Lemeshow S, Sturdivant RX (2013). *Applied Logistic Regression*, volume 398. John Wiley & Sons.

Imbens GW, Rubin DB (2015). *Causal Inference in Statistics, Social, and Biomedical Sciences*. Cambridge University Press.

Jiang H, Qi P, Zhou J, Zhou J, Rao S (2021). "A Short Survey on Forest Based Heterogeneous Treatment Effect Estimation Methods: Meta-Learners and Specific Models." In *2021 IEEE International Conference on Big Data (Big Data)*, pp. 3006–3012. IEEE.

Johansson FD, Shalit U, Kallus N, Sontag D (2020). "Generalization Bounds and Representation Learning for Estimation of Potential Outcomes and Causal Effects." *arXiv preprint arXiv:2001.07426*.

Kang JD, Schafer JL (2007). "Demystifying Double Robustness: A Comparison of Alternative Strategies for Estimating a Population Mean from Incomplete Data." *Statistical Science*, pp. 523–539.

Kuhn, Max (2008). "Building Predictive Models in R Using the **caret** Package." *Journal of Statistical Software*, **28**(5), 1–26. doi:10.18637/jss.v028.i05. URL https://www.jstatsoft.org/index.php/jss/article/view/v028i05.

Le Borgne F, Chatton A, Léger M, Lenain R, Foucher Y (2021). "G-Computation and Machine Learning for Estimating the Causal Effects of Binary Exposure Statuses on Binary Outcomes." *Scientific Reports*, **11**(1), 1–12.

Loiseau N, Trichelair P, He M, Andreux M, Zaslavskiy M, Wainrib G, Blum MG (2022). "External Control Arm Analysis: An Evaluation of Propensity Score Approaches, G-computation, and Doubly Debiased Machine Learning." *medRxiv*. doi:10.1101/2022.01.28.22269591. https://www.medrxiv.org/content/early/2022/01/30/2022.01.28.22269591.full.pdf, URL https://www.medrxiv.org/content/early/2022/01/30/2022.01.28.22269591.

Lunceford JK, Davidian M (2004). "Stratification and Weighting via the Propensity Score in Estimation of Causal Treatment Effects: A Comparative Study." *Statistics in Medicine*, **23**(19), 2937–2960.

Mayer I, Zhao P, Greifer N, Huntington-Klein N, Josse J (2022). "CRAN Task View: Causal Inference." Version 2022-12-07, URL https://cran.r-project.org/web/views/CausalInference.html.

Oehlert GW (1992). "A Note on the Delta Method." *The American Statistician*, **46**(1), 27–29.

Pearl J (2009). *Causality*. Cambridge University Press.

Powers S, Qian J, Jung K, Schuler A, Shah NH, Hastie T, Tibshirani R (2018). "Some Methods for Heterogeneous Treatment Effect Estimation in High Dimensions." *Statistics in Medicine*, **37**(11), 1767–1787.

Research M (2019). "**EconML**: A Python Package for ML-Based Heterogeneous Treatment Effects Estimation." https://github.com/microsoft/EconML. Version 0.x.

Rosenbaum PR, Rubin DB (1983). "The Central Role of the Propensity Score in Observational Studies for Causal Effects." *Biometrika*, **70**(1), 41–55. doi:10.1093/BIOMET/70.1.41.

Rudolph KE, Schmidt NM, Glymour MM, Crowder R, Galin J, Ahern J, Osypuk TL (2018). "Composition or Context: Using Transportability to Understand Drivers of Site Differences in a Large-scale Housing Experiment." *Epidemiology (Cambridge, Mass.)*, **29**(2), 199.

Saul BC, Hudgens MG (2020). "The Calculus of M-Estimation in R with **geex**." *Journal of Statistical Software*, **92**(2).

Schuemie MJ, Cepeda MS, Suchard MA, Yang J, Tian Y, Schuler A, Ryan PB, Madigan D, Hripcsak G (2020). "How Confident are We about Observational Findings in Healthcare: A Benchmark Study." *Harvard Data Science Review*, **2**(1).

Schuler A, Jung K, Tibshirani R, Hastie T, Shah N (2017). "Synth-Validation: Selecting the Best Causal Inference Method for a Given Dataset." *arXiv preprint arXiv:1711.00083*.

Sharma A, Kiciman E, *et al.* (2019). "**DoWhy**: A Python Package for Causal Inference." https://github.com/microsoft/dowhy.

Shimoni Y, Yanover C, Karavani E, Goldschmnidt Y (2018). "Benchmarking Framework for Performance-Evaluation of Causal Inference Analysis." *arXiv preprint arXiv:1802.05046*.

Stuart EA (2010). "Matching Methods for Causal Inference: A Review and a Look Forward." *Statistical Science: A Review Journal of the Institute of Mathematical Statistics*, **25**(1), 1. doi:10.1214/09-STS313.

Swaminathan A, Joachims T (2015). "The Self-Normalized Estimator for Counterfactual Learning." *Advances in Neural Information Processing Systems*, **28**.

Tikka S, Hyttinen A, Karvanen J (2021). "Causal Effect Identification from Multiple Incomplete Data Sources: A General Search-Based Approach." *Journal of Statistical Software*, **99**(5), 1–40. doi:10.18637/jss.v099.i05.

Tikka S, Karvanen J (2017). "Identifying Causal Effects with the R Package **causaleffect**." *Journal of Statistical Software*, **76**(12), 1–30. doi:10.18637/jss.v076.i12.

Tipton E (2021). "Beyond Generalization of the ATE: Designing Randomized Trials to Understand Treatment Effect Heterogeneity." *Journal of the Royal Statistical Society Series A: Statistics in Society*, **184**(2), 504–521.

Wager S, Athey S (2018). "Estimation and Inference of Heterogeneous Treatment Effects Using Random Forests." *Journal of the American Statistical Association*, **113**(523), 1228–1242.

Wendling T, Jung K, Callahan A, Schuler A, Shah NH, Gallego B (2018). "Comparing Methods for Estimation of Heterogeneous Treatment Effects Using Observational Data from Health Care Databases." *Statistics in Medicine*, **37**(23), 3309–3324.

Xie Y, Brand JE, Jann B (2012). "Estimating Heterogeneous Treatment Effects with Observational Data." *Sociological Methodology*, **42**(1), 314–347.

Yao L, Li S, Li Y, Huai M, Gao J, Zhang A (2018). "Representation Learning for Treatment Effect Estimation from Observational Data." *Advances in Neural Information Processing Systems*, **31**.

Zeng S, Li F, Wang R, Li F (2021). "Propensity Score Weighting for Covariate Adjustment in Randomized Clinical Trials." *Statistics in medicine*, **40**(4), 842–858.

Zhao Y, Liu Q (2023). "**Causal ML**: Python Package for Causal Inference Machine Learning." *SoftwareX*, **21**, 101294.

A. Notation used throughout the paper

Notation	Description
$\boldsymbol{X}$	random vector of length d of covariates, containing all pre-treatment outcome predictors.
$\boldsymbol{X}_t \subseteq \boldsymbol{X}$	random vector of length q, indicating confounders.
$\boldsymbol{X}_s \subseteq \boldsymbol{X}$	random vector of length p, indicating sample selection predictors of an observational dataset
T	treatment indicator ($T = 1$ for the treatment, $T = 0$ for the control)
Y	outcomes of interest. **RCTrep** supports binary outcomes and continuous outcomes
S	selection indicator ($S = 1$ indicates selection to a sample of an RCT, $S = 0$ indicates selection to a sample of an observational study)
$\mathcal{S}^{rct}$	$\mathcal{S}^{rct} = \{(\boldsymbol{X}_i, Y_i, T_i); S_i = 1\}$, an RCT sample
$\mathcal{S}^{obs}$	$\mathcal{S}^{obs} = \{(\boldsymbol{X}_i, Y_i, T_i); S_i = 0\}$, an observational sample
$\mathcal{P}_{\boldsymbol{\theta}}$	a target population parameterized by $\boldsymbol{\theta}$ that $\mathcal{S}^{rct}$ represents for
$\pi_t(\boldsymbol{x})$	the propensity score of an individual with characteristics $\boldsymbol{X} = \boldsymbol{x}$ being selected to treatment $T = 1$
$\pi_s(\boldsymbol{x})$	the probability of an individual with the characteristics $\boldsymbol{X} = \boldsymbol{x}$ being selected to an RCT $S = 1$
τ	the ATE of the target population $\mathcal{P}_{\boldsymbol{\theta}}$
$\tau(\boldsymbol{x})$	the CATE, denoted as $\tau(\boldsymbol{x}) = \mathbb{E}[Y(1) - Y(0) \mid \boldsymbol{X} = \boldsymbol{x}]$
σ_1^2, σ_0^2	the variance of potential outcomes $Y(1)$, $Y(0)$
$\sigma_t^2(\boldsymbol{x})$	the conditional variance of $Y(t)$, denoted as $\mathbb{V}(Y(t) \mid \boldsymbol{x})$
$p(\boldsymbol{x}_s), q(\boldsymbol{x}_s)$	the density of covariates $\boldsymbol{X}_s$ in $\mathcal{S}^{rct}$ and $\mathcal{S}^{obs}$
$w(\boldsymbol{x}_s)$	the density ratio of covariates $\boldsymbol{x}_s$ defined as $\frac{p(\boldsymbol{x}_s)}{q(\boldsymbol{x}_s)}$
$\pi_t(\boldsymbol{X}; \hat{\boldsymbol{\alpha}})$	an estimator for the propensity score
$\pi_s(\boldsymbol{X}; \hat{\boldsymbol{\gamma}})$	an estimator for the selection probability
$p(\boldsymbol{X}, t; \hat{\boldsymbol{\beta}})$	an estimator for the conditional mean of potential outcomes $\mathbb{E}[Y(t) \mid \boldsymbol{x}]$ parameterized by $\hat{\boldsymbol{\beta}}$ using the G-computation method
$\hat{\tau}(\boldsymbol{X})$	an estimator for the CATE $\tau(\boldsymbol{X})$
$\hat{\sigma}_1, \hat{\sigma}_0$	estimators for the variance of $Y(1), Y(0)$
$\hat{p}(\boldsymbol{X}), \hat{q}(\boldsymbol{X})$	estimators for the density of $\boldsymbol{X}$ in $\mathcal{S}^{rct}$ and $\mathcal{S}^{obs}$
ϵ_i^t	the residual of an estimator $p_t(\boldsymbol{X}, t_i; \hat{\boldsymbol{\beta}})$, defined as $\epsilon_i = Y_i - p(\boldsymbol{X}_i, t_i; \hat{\boldsymbol{\beta}})$
$\hat{\sigma}_t^2(\boldsymbol{X})$	an estimator of the conditional variance of $Y(t)$, denoted as $\hat{\mathbb{V}}(Y(t) \mid \boldsymbol{X})$

Table 5: List of notations.

B. Core classes

The current section offers additional background information on **RCTrep**'s classes structures - both on R6 class system (Chang 2019) and on each of the three previously introduced core **RCTrep** classes. Together with the information in the next section, on `TEstimator` and `SEstimator` implementation, this should be able to get users up and running with developing users own custom `TEstimator` and `SEstimator` subclasses.

B.1. Choice for the R6 class system

Though widely used as a procedural language, R offers several Object Oriented (OO) systems, which can significantly help in structuring the development of more complex packages. Out of the OO systems available (S3, S4, R5, and R6), we settled on R6, as it offers several advantages over other options. Firstly, it implements a mature object-oriented design compared to S3 and S4, hence is easier for developers with a background in programming languages such as C++ and Java to maintain. Secondly, when compared to the older R5 reference class systems, R6 classes are much lighter-weight, as they do not use S4 classes, and do not require the **methods** package.

B.2. Core classes: TEstimator, SEstimator, Fusion

In this section, we go over the three core classes on more detail - with an emphasis on the `TEstimator` and `SEstimator` classes. We illustrate the structure of classes, and enumerate core public functions of each class.

TEstimator

The `TEstimator` class is responsible for fitting a model and estimating treatment effects. The following skeleton code gives an overview of how the above is implemented in **RCTrep**'s `TEstimator` class:

```
TEstimator <- R6::R6Class(
  "TEstimator",
  public = list(
    id = NA,
    name = character(),
    statistics = list(n=numeric(),
                      density_confounders=data.frame()),
    data = NULL,
    estimates = list(ATE=data.frame(y1.hat=NA,
                                    y0.hat=NA,
                                    est=NA,
                                    se=NA),
                     CATE = data.frame()),
    model = list(),
    initialize = function(df, vars_name, name) {
      self$name <- name
      self$data <- df
```

```
      self$data$id <- seq(dim(df)[1])
      private$outcome_predictors <-
      vars_name$outcome_predictors
      private$treatment_name <- vars_name$treatment_name
      private$outcome_name <- vars_name$outcome_name
      self$statistics <- list(n=dim(df)[1],
                              density_confounders=
                              private$est_joint_denstiy())
    },
    get_CATE = function(stratification, stratification_joint=TRUE) {},
    plot_CATE = function(stratification = private$outcome_predictors,
                         stratification_joint = TRUE) {},
    plot_y1_y0 = function(stratification, stratification_joint = TRUE,
                          seperate = FALSE){},
    diagnosis_t_overlap = function(stratification,
                                   stratification_joint=TRUE){},
    diagnosis_t_ignorability = function(){},
    diagnosis_y_overlap = function(stratification,
                                   stratification_joint=TRUE){}
  ),
  private = list(
    outcome_predictors = NA,
    treatment_name = NA,
    outcome_name = NA,
    var_method = "sandwitch",
    isTrial = FALSE,

    set_ATE = function(){},
    set_CATE = function(stratification, stratification_joint){},
    est_joint_denstiy = function(){},
    est_CATEestimation4JointStratification = function(stratification) {},
    est_CATEestimation4SeperateStratification = function(stratification) {},
    fit = function(){},
    est_ATE_SE = function(){},
    est_weighted_ATE_SE = function(){}
  )
)
```

Subclasses of `TEstimator` have their unique implementation of `diagnosis_t_ignorability()`, `fit()`, `est_ATE_SE()`, and `est_weighted_ATE_SE()`, and their unique private methods. The main `TEstimator` functions are:

1. `get_CATE(stratification, stratification_joint=TRUE)`
   (a) `stratification`: a character vector specifies covariates to select subpopulations.
   (b) `stratification_joint`: logical to indicate if subpopulations are selected based on

levels of individual covariate in `stratification` or levels of combined covariates in `stratification`.

The function returns a `data.frame` containing treatment effects estimation of selected subpopulations. If `stratification=TRUE`, then the function returns a `data.frame` with column names `c(stratification,"y1.hat","y0.hat","cate","se","size")`; if `strati fication_joint=FALSE`, then the function returns a `data.frame` with column names `c("name", "value","y1.hat","y0.hat","cate","se","size")`.

2. `diagnosis_t_overlap(stratification, stratification_joint)`: the function plots the proportion and the count of individuals receiving the treatment and the control in each subpopulation. Subpoulations are selected by `stratification` and `stratificatio n_joint`.

3. `diagnosis_t_ignorability()`: the function diagnoses the T-ignorability assumption. For the subclass `G_computation`, the function summarizes the model fit using the following evaluation metrics, i.e., means of residuals of subpopulations, distribution of overall residuals, mean squared errors of subpopulations for a continuous outcome, and mean of deviance of subpopulations for a binary outcome. For the subclass `IPW`, the function plots the weighted distribution of subpopulations in treatment and control groups. For the subclass `DR`, the function summarizes both the model fit and weighted distribution of subpoulations in treatment and control groups.

4. `diagnosis_y_overlap(stratification, stratification_joint)`: the function plots the count of each level of the outcome in treatment and control groups within each subpopulation selected by `stratification` and `stratification_joint`. For the binary outcome, the function plots the count of the positive outcome and the negative outcome; for continuous outcomes, the function plots the distribution of outcomes.

5. private method `set_ATE()`: the function implements the private method `est_ATE_SE(id)` and gets the point estimate of the ATE, the standard error of the point estimate, mean of estimates of the potential outcomes; the function assigns these estimates to the public fields `estimates$ATE$est`, `estimates$ATE$se`, `estimates$ATE$y1.hat`,`estimates$AT E$y0.hat` accordingly. The function is implemented in the initialize function of each `TEstimator` subclass.

6. private method `set_CATE(stratification, stratification_joint)`: the function implements the public method `get_CATE(stratification, stratification_joint)` which returns a `data.frame` (see below for details of the returned object from the function `get_CATE()`); then the function `set_CATE()` assigns the returned estimates from `get_CATE()` to the public field `estimates$CATE`. The function is implemented in the initialize function of each subclass of `TEstimator` by calling `private$set_CATE(private$o utcome_predictors,TRUE)`.

7. private method `est_CATEestimation4JointStratification(stratification)`: the function selects subpopulations defined by levels of combined covariates specified in `stratification`, gets the index of selected data, and estimates the ATE of each subpopulation by calling the private method `est_ATE_SE(index)`. The function returns a `data.frame` with the column name `c(stratification, "y1.hat", "y0.hat", "cate", "se", "size")`.

8. private method `est_CATEestimation4SeperateStratification(stratification)`: the function selects subpopulations defined by levels of individual covariate specified in `stratification`, gets the index of selected data, and estimates the ATE of each subpopulation by calling the private method `est_ATE_SE(index)`. The function returns a `data.frame` with the column name `c("name", "value", "y1.hat", "y0.hat", "cate", "se", "size")`.

9. private method `est_ATE_SE(index)`: the function estimates the ATE and its standard error. `index` indicates the index of data. A different subclass has unique implementation of the point estimation. **RCTrep** implements the sandwich estimator to estimate the standard error using R package **geex** (Saul and Hudgens 2020). **RCTrep** specifies an estimation function `estFUN`, and passes the function to `geex::m_estimate(data, estFUN, ...)`. `geex::m_estimate()` provides a consistent estimator for the asymptotic variance of the estimate of the ATE. **RCTrep** does not take the uncertainty of the estimation of parameters of models into account in order to speed up implementation, however, users can customize `estFUN` so the function can take account of the uncertainty of the estimation of parameters into the estimation of the variance of estimates of the ATE. For more details, see simulation codes in Dahabreh *et al.* (2020) and tutorials by Saul and Hudgens (2020). `est_ATE_SE(index)` function returns a `list` with named elements `y1.hat`, `y0.hat`, `est`, and `se`. An overview of estimators of the variance of estimates of the ATE is provided in Appendix C.

10. private method `est_weighted_ATE_SE(index, weight)`: the function estimates the weighted ATE and its standard error. The function selects estimates of potential outcomes from `self$data[index,]$y1.hat` and `self$data[index,]$y0.hat`, and assigns weights for the selected estimates. **RCTrep** implements the sandwich estimator using R package **geex** to estimate the standard error of estimates of the weighted ATE. The function returns a `list` with named elements `y1.hat`, `y0.hat`, `est`, and `se`.

11. private method `est_CATEestimation4JointStratification(stratification)`: the function estimates the ATE of subpopulations. The function selects a subpopulation based on levels of combined covariates in `stratification`, gets `id` of the selected subpopulation, and computes the ATE of the subpopulation by calling `private_ATE_SE(id)`. Loop this procedure until all subpopulations have been selected. The function returns a `data.frame` with column names `c(stratification, "y1.hat", "y0.hat", "cate", "se", "size")`.

12. private method `est_CATEestimation4SeperateStratification(stratification)`: the function estimates the ATE of subpopulations. The function selects a subpopulation based on levels of the individual covariate in `stratification`, gets `id` of the selected subpopulation, and computes the ATE of the subpopulation by calling `private$est_ATE_SE( id)`. Loop this procedure until all subpopulations have been selected. The function returns a `data.frame` with column names `c("name", "value", "y1.hat", "y0.hat", "cate", "se", "size")`.

SEstimator

The `SEstimator` class is responsible for balancing covariates in `selection_predictors` between two objects of the class `TEstimator`, and estimates the weighted ATE and the weighted

CATE. The following skeleton code gives an overview of how the weighted estimation is implemented in **RCTrep**'s `SEstimator` class:

```
SEstimator <- R6::R6Class(
  "SEstimator",
  public = list(
    name = character(),
    id = character(),
    statistics = list(),
    estimates = list(ATE = data.frame(y1.hat=NA,
                                      y0.hat=NA,
                                      est=NA,
                                      se=NA),
                     CATE = data.frame()),
    model = NA,
    selection_predictors = NA,
    weighting_method = character(),

    initialize = function(target.obj, source.obj, weighting_method=NULL,
                          selection_predictors){
      private$target.obj <- target.obj
      private$source.obj <- source.obj
      self$weighting_method <- weighting_method
      self$selection_predictors <- selection_predictors
      private$ispublic <- !c("TEstimator_pp") %in% class(source.obj)
      self$name <- source.obj$name
      self$statistics <- source.obj$statistics
      self$id <- paste(private$source.obj$id,
                       self$weighting_estimator,
                       length(self$selection_predictors),sep = '/')
      private$isTrial <- source.obj$.__enclos_env__$private$isTrial
    },
    EstimateRep = function(stratification=self$selection_predictors,
                           stratification_joint=TRUE) {},
    diagnosis_s_overlap = function(stratification=NULL,
                                   stratification_joint=TRUE){},
    diagnosis_s_ignorability = function(stratification=NULL,
                                        stratification_joint=TRUE){}
  ),

  private = list(
    source.obj = NA,
    target.obj = NA,
    ispublic = NA,
    isTrial = NA,

    get_weight = function(){source.data,target.data, vars_weighting},
```

```
    set_weighted_ATE_SE = function() {},
    set_weighted_CATE_SE = function(stratification, stratification_joint) {},
    est_WeightedCATEestimation4JointStratification =
    function(stratification) {},
    est_WeightedCATEestimation4SeperateStratification =
    function(stratification) {},
    est_statistics = function(){}
  )
)
```

The following are public and private functions in `SEstimator`:

1. public function `EstimateRep(stratification, stratification_joint)`: the core function which estimates the weighted ATE and the weighted CATE; `stratification` and `stratification_joint` specify a criteria to select subpopulations.

2. `diagnosis_s_overlap(stratification=NULL, stratification_joint=TRUE)`: the function selects subpopulations according to `stratification, stratification_joint`; the function plots the proportion and the count of individuals in each subpopulation from `source.obj` and `target.obj`. The default value of `stratification` is `selection_predictors`.

3. `diagnosis_s_ignorability(stratification=NULL, stratification_joint=TRUE)`: the function diagnoses the assumption of S-ignorability. The function selects subpopulations according to `stratification, stratification_joint`. It computes the weighted distribution of the subpopulations in `source.obj` and the distribution of the subpopulations in `target.obj`.

4. private method `get_weight(source.data, target.data, vars_weighting)`: the function estimates weights for each individual in `source.obj`. The weights are computed based on specified covariates `vars_weighting`. Each subclass of `SEstimator` has a unique implementation of the function:

    - `SEexact`: the class performs exact matching and computes the weight accordingly. The implementation of the weight estimation depends on R package **MatchIt** (Ho *et al.* 2011).
    - `SEisw`: weighting based on the inverse selection probability. Methods for estimating the selection probability is specified in `self$weighting_method` argument. Allowable options of `weighting_method` are inherent from values of the argument `method` in the function `train()` of R package **caret** (Kuhn and Max 2008).
    - `SEsubclass`: weighting based on subclassification on the selection probability of the data in `source.obj`. Methods for estimating the selection probability are specified in `self$weighting_method`. The default is `glm` for the selection probability using the logistic regression which regresses the selection indicator on `selection_predictors`. The main effects of covariates in `selection_predictors` are specified in the function specification. The observational dataset in `source.obj` and the RCT dataset in `target.obj` are placed into subclasses based on quantiles

of the selection probability of the RCT datasets. Then weights for individuals in the observational dataset are computed based on the proportion of individuals from the RCT dataset in each subclass.

- `SEstimator_pp`: weighting for two objects of the class `TEstimator_pp`. Weight is computed as $w(\boldsymbol{x}_{si}) = \frac{w'(\boldsymbol{x}_{si})}{\sum_{i \in S^{obs}} w'(\boldsymbol{x}_{si})}, w'(\boldsymbol{x}_{si}) = \frac{\hat{p}(\boldsymbol{x}_s)}{\hat{q}(\boldsymbol{x}_s)}$

5. private method `set_weighted_ATE_SE`: the function estimates the weighted ATE of `source.obj`. The function calls `private$get_weight(source.data=private$source.obj$data, target.data=private$target.obj$data, vars_weighting=self$selection_predictors)` to compute weights, then calls the private method `est_weighted_ATE_SE()` of `source.obj` to estimate the weighted ATE and gets the weighted estimates of `y1.hat`, `y0.hat`, `est`, and `se` accordingly, then assigns the estimates to `self$estimates$ATE$y1.hat`, `self$estimates$ATE$y0.hat`, `self$estimates$ATE$est`, `self$estimates$ATE$se`.

6. private method `set_weighted_CATE_SE(stratification, stratification_joint)`: the function estimates the weighted CATE; if `stratification_joint=TRUE`, then the function calls `private$est_WeightedCATEestimation4JointStratification(stratification)`; if `stratification_joint=FALSE`, then the function calls `private$est_WeightedCATEestimation4SeperateStratification(stratification)`. `Stratification` is a character vector that specifies covariates for the subpopulation selection.

7. private method `est_WeightedCATEestimation4JointStratification(stratification)`: the function estimates the weighted CATE. The function selects subpopulations from `private$source.obj$data` and `private$target.obj$data`, and calls `private$get_weight()` to estimate weights of each individual in `source.obj` so that covariates in `self$selection_predictors` are balanced between `source.obj` and `target.obj`. The function returns a `data.frame` in the same form as that returned from the private method `est_CATEestimation4JointStratification(stratification)` of the class `TEstimator`.

8. private method `est_WeightedCATEestimation4SeperateStratification(stratification)`: the same as the `est_WeightedCATEestimation4JointStratification(stratification)` except for the criteria to select subpopulations. The function returns a `data.frame` in the same form as that returned from the private method `est_CATEestimation4SeperateStratification(stratification)` of the class `TEstimator`.

Fusion

The `Fusion` class is responsible for aggregating estimates from objects of the classes `TEstimator` and `SEstimator`, evaluating methods for the treatment effect estimation implemented in class `TEstimator`, plotting and printing results. The following skeleton code gives an overview of the class `Fusion`:

```
Fusion <- R6::R6Class(
  "Fusion",
  public = list(
    objs.cate.data = data.frame(),
```

```
    objs.ate.data = data.frame(),
    stratification = NA,
    stratification_joint = NA,
    RCT.study.name = NA,
    RWD.study.name = NA,

    initialize = function(...){},
    plot = function(){},
    print = function(){},
    evaluate = function(){}
  ),

  private = list(
    aggregate_cate_estimates = function(...){},
    aggregate_ate_estimates = function(...){}
  )
)
```

The following are public and private methods in `Fusion`:

1. constructor `initialize(...)` initializes an object of the class `Fusion`; passes objects of the class `TEstimator` and `SEstimator` to the argument `...`. The number of objects passed to the function is not limited.

2. public function `plot()`, `print()` plots and prints estimates of the ATE of population and subpopulations from passed objects.

3. public function `evaluate()`: the function computes validation results using the following metrics:

   - pseudo mse `mse`;
   - length of the confidence interval `length_ci`;
   - estimate agreement `agg.est`;
   - and regulatory agreement `agg.reg`.

   The regulatory agreement is defined as the consistency of the direction and the statistical significance of estimates from two data sets, and the estimate agreement indicates whether an estimate using observational data lies within the 95% confidence interval of the estimate using RCT data (Franklin *et al.* 2020). The function computes these evaluation metrics on population and subpopulation levels. Subpopulations are selected according to `self$stratification` and `self$stratification_joint`, which are inherent from arguments passed to the function `EstimateRep()` of the object of the class `SEstimator`.

4. private method `aggregate_ate_estimates()` and private method `aggregate_cate_estimates()`: the functions aggregate estimates of the ATE and the CATE from all objects passed to `...` of `initialize()`.

B.3. Subclasses of TEstimator and SEstimator

Subclasses of `TEstimator` are mainly responsible for fitting models and estimating treatment effects using their unique methods `est_ATE_SE`. Users can override `est_ATE_SE` for a new subclass of `TEstimator`. Subclasses of `SEstimator` are responsible for estimating weights using their unique methods `get_weight`. Users can override the function for a new subclass of `SEstimator`.

Since the aim of combining data is to estimate weights, it is not necessary to have individual-level data. For instance, each object needs to share 1) the density of $\boldsymbol{X}_s$, 2) estimates $\hat{\tau}(\boldsymbol{X}_s)$ and the standard error of $\hat{\tau}(\boldsymbol{X}_s)$, and 3) the sample size for each subpopulation stratified by $\boldsymbol{X}_s$. The weighted treatment effects can be derived accordingly. Hence, in case full data is not allowed to share, **RCTrep** defines a subclass `TEstimator_pp` for `TEstimator` and `SEstimator_pp` for `SEstimator`. In `TEstimator_pp`, instead of assigning individual-level data to the public field `data`, the class assigns the density of covariates in `outcome_predictors` and the estimates of the treatment effect of subpopulations stratified by levels of covariates in `outcome_predictors` to the public field `data` of an object of the class `TEstimator_pp`. Two objects of the class `TEstimator_pp` are passed to an object of the class `SEstimator` that estimates weights $w(\boldsymbol{X}_s)$ based on the public field `data` of these two assigned objects. For different weighting approaches, users can share different aggregate data. For instance, weighting using balancing-based methods requires $p(B(\boldsymbol{x}_s))$ (Chatton *et al.* 2020), where $B(\boldsymbol{x}_s)$ is the basis function of $\boldsymbol{x}_s$, e.g., interaction between two random variables. Hence, in this case, the density of the basis function $B(\boldsymbol{x}_s)$ is needed. To conclude, users can override the public field `data` in a new class of `TEstimator_pp` and override `get_weight()` in a new class of `SEstimator_pp` accordingly.

C. The variance of estimates of the ATE using three methods

RCTrep has three methods for estimating the ATE, namely, the G-computation, the IPW, and the DR methods. The G-computation method is unbiased and consistent as long as a model for the outcome is correctly specified. IPW is unbiased as long as a model for the treatment, i.e., the propensity score, is correctly specified. The DR method is unbiased as long as either a model for the outcome or a model for the treatment is correctly specified and is more efficient than the IPW method. In this section we show the variance of three methods for illustrative purposes; we demonstrate the effect of three factors on the variance estimation, namely, weights, model assumptions, and sample size. In the following, we analyze the variance of these estimators.

C.1. The variance of estimates of the ATE using the G-computation method

The assumption of T-ignorability implies that conditioning on confounders, the treatment assignment can be assumed random and hence the treatment effect can be identified as a simple difference in means between two groups within each subpopulation stratified by these confounders. The estimate of the ATE using the G-computation method is defined as:

$$\hat{\tau} = \mathbb{E}[\hat{\tau}(\boldsymbol{X})] = \mathbb{E}[p(\boldsymbol{X}, 1; \hat{\boldsymbol{\beta}}) - p(\boldsymbol{X}, 0; \hat{\boldsymbol{\beta}})] \approx \frac{1}{n} \sum_{i \in \mathcal{S}} p(\boldsymbol{x}_i, 1; \hat{\boldsymbol{\beta}}) - p(\boldsymbol{x}_i, 0; \hat{\boldsymbol{\beta}}) \tag{4}$$

where $\boldsymbol{X}$ is a random vector of all pre-treatment outcome predictors containing all confounders, $p(\boldsymbol{X}, T; \hat{\boldsymbol{\beta}}) = \hat{\mathbb{E}}[Y \mid \boldsymbol{X}, T]$. We can use both parametric and non-parametric models to estimate the conditional mean of potential outcomes given $\boldsymbol{X}$, in other words, $\hat{\boldsymbol{\beta}} \subset \mathbb{R}^{\mathbb{R}}$. Here we use $\boldsymbol{\beta}$ to denote a set of parameters that describes the distribution of the conditional mean of potential outcomes. We assume the conditional mean is expressed as an equation linear in $\boldsymbol{X}$ and T, and hence can be described by a fixed length of parameters $\boldsymbol{\beta}$. We can also assume that the conditional mean is described by a flexible function parameterized by $\boldsymbol{\beta}$ of flexible length depending on a model constraint, regularization, and sample size. $p(\boldsymbol{x}, 1; \hat{\boldsymbol{\beta}})$ is the estimate of $\mathbb{E}[Y(1) \mid \boldsymbol{x}]$ parameterized by $\hat{\boldsymbol{\beta}}$, $\hat{\boldsymbol{\beta}}$ is estimated using a sample $\mathcal{S}$. Then the variance of the estimator $\hat{\tau}(\boldsymbol{X})$ is derived as:

$$
\begin{aligned}
\mathbb{V}(\hat{\tau}(\boldsymbol{X})) &= \mathbb{E}[\mathbb{V}(\hat{\tau}(\boldsymbol{X}) \mid \boldsymbol{X})] + \mathbb{V}\left(\mathbb{E}[\hat{\tau}(\boldsymbol{X}) \mid \boldsymbol{X}]\right) \quad \text{law of total variance} \\
&= \mathbb{E}\left[\mathbb{V}\left(p(\boldsymbol{X}, 1; \hat{\boldsymbol{\beta}}) - p(\boldsymbol{X}, 0; \hat{\boldsymbol{\beta}}) \mid \boldsymbol{X}\right)\right] + \mathbb{V}\left(\mathbb{E}[p(\boldsymbol{X}, 1; \hat{\boldsymbol{\beta}}) - p(\boldsymbol{X}, 0; \hat{\boldsymbol{\beta}}) \mid \boldsymbol{X}]\right) \\
&\approx \mathbb{E}\left[\mathbb{V}\left(p(\boldsymbol{X}, 1; \hat{\boldsymbol{\beta}}) \mid \boldsymbol{X}\right) + \mathbb{V}\left(p(\boldsymbol{X}, 0; \hat{\boldsymbol{\beta}}) \mid \boldsymbol{X}\right)\right] + \mathbb{V}\left(p(\boldsymbol{X}, 1; \bar{\hat{\boldsymbol{\beta}}}) - p(\boldsymbol{X}, 0; \bar{\hat{\boldsymbol{\beta}}})\right) \\
&\approx \frac{1}{n} \sum_i \left(\hat{\mathbb{V}}(p(\boldsymbol{x}_i, 1; \hat{\boldsymbol{\beta}})) + \hat{\mathbb{V}}(p(\boldsymbol{x}_i, 0; \hat{\boldsymbol{\beta}}))\right) + \hat{\mathbb{V}}(p(\boldsymbol{X}, 1; \bar{\hat{\boldsymbol{\beta}}}) - p(\boldsymbol{X}, 0; \bar{\hat{\boldsymbol{\beta}}}))
\end{aligned}
\tag{5}
$$

Note in the third line, the first term is a function of $\boldsymbol{X}$ and the variance of $\hat{\boldsymbol{\beta}}$ depending on a sample, and hence the variance of this term depends on the sample. In logistic regression, the variance of $\hat{\boldsymbol{\beta}}$ is well-developed, and most of software can provide the estimate of the variance of these parameters. In non-parametric methods, it is not trivial to write down the closed form of the variance of parameters; alternative approaches to estimating $\mathbb{V}(p(\boldsymbol{x}_i, t_i; \hat{\boldsymbol{\beta}}))$ are the delta method, bootstrap, etc. We introduce approaches for estimating the variance of $p(\boldsymbol{x}_i, t_i; \hat{\boldsymbol{\beta}})$ in the next section. The second term in the third line is the variance between groups $\mathbb{V}(\hat{\tau}(\boldsymbol{X}; \bar{\hat{\boldsymbol{\beta}}}))$, and only $\boldsymbol{X}$ is random, hence the variance of the second term can be estimated using the sample variance of $\hat{\tau}(\boldsymbol{X}; \bar{\hat{\beta}})$ where $\bar{\hat{\boldsymbol{\beta}}} = \mathbb{E}[\hat{\boldsymbol{\beta}}]$. We use an estimate of $\hat{\boldsymbol{\beta}}$ based on a sample as an estimate of $\bar{\hat{\boldsymbol{\beta}}}$, and estimate the sample variance of plugged-in $\hat{\tau}(\boldsymbol{X}; \bar{\hat{\boldsymbol{\beta}}})$.

Methods for estimating the variance of estimates of the ATE using the G-computation method

In this section, we illustrate five methods for estimating the variance of estimates of the ATE using the G-computation method first, i.e., $\mathbb{V}(p(\boldsymbol{x}, t; \hat{\boldsymbol{\beta}}))$. In the following, for demonstrative purposes, we use logistic regression to estimate the conditional mean of potential outcomes. $p(\boldsymbol{x}, t; \hat{\boldsymbol{\beta}}) = \sigma(\boldsymbol{x}, t; \hat{\boldsymbol{\beta}}) = \frac{1}{1+exp^{-(\boldsymbol{x},t)'\hat{\boldsymbol{\beta}}}}$, where $\mathbb{V}(p(\boldsymbol{x}, t; \hat{\boldsymbol{\beta}}))$ can be estimated by the following five methods:

1. The model-based method, where $\mathbb{V}(\boldsymbol{\beta}) = \mathbf{I}^{-1}(\boldsymbol{\beta})$, $\mathbf{I}(\boldsymbol{\beta})$ is the observed information

matrix. $\mathbb{V}(\boldsymbol{\beta})$ can be estimated at $\hat{\boldsymbol{\beta}}$, denoted as $\hat{\mathbb{V}}(\hat{\boldsymbol{\beta}}) = \left(\mathbf{X}'\hat{\mathbf{V}}\mathbf{X}\right)^{-1}$, where

$$\hat{\mathbf{V}} = \begin{bmatrix} \hat{p}_1\left(1-\hat{p}_1\right) & 0 & \cdots & 0 \\ 0 & \hat{p}_2\left(1-\hat{p}_2\right) & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & 0 & \hat{p}_n\left(1-\hat{p}_n\right) \end{bmatrix},$$

$\hat{p}_i$ is the predicted observed outcome, then

$$\hat{\mathbb{V}}(p(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}})) = \mathbf{x}_i'\hat{\mathbb{V}}(\hat{\boldsymbol{\beta}})\mathbf{x}_i = \sum_j x_{ij}^2\hat{\mathbb{V}}(\hat{\beta}_j) + 2\sum_{j=0}^{p}\sum_{k=j+1}^{p} x_{ij}x_{ik}\widehat{\mathrm{Cov}}(\hat{\beta}_j, \hat{\beta}_k). \tag{6}$$

where we regard $T_i = t$ as an element in the vector $\mathbf{x}_i$, i.e., $\mathbf{x}_i = (\boldsymbol{x}_i, t)'$, $\hat{\mathbb{V}}(\hat{\boldsymbol{\beta}}_j)$ is the jth diagonal element of the matrix $\hat{\mathbb{V}}(\hat{\boldsymbol{\beta}})$, and $\widehat{\mathrm{Cov}}(\hat{\boldsymbol{\beta}}_j, \hat{\boldsymbol{\beta}}_k)$ is an off-diagonal element in the matrix. Then we can estimate $\hat{\mathbb{V}}(p(\boldsymbol{x}, 1; \hat{\beta}))$ and $\hat{\mathbb{V}}(p(\boldsymbol{x}, 0; \hat{\beta}))$ for each individual i. We estimate the sample average of $\hat{\mathbb{V}}(p(\boldsymbol{x}, t; \hat{\beta}))$ as the estimate of expectation of the variance within groups, i.e., the first term in the last line of the variance decomposition in Equation 5. For the variance between groups, i.e., the second term in the equation, we estimate the sample variance of $\hat{\tau}(\boldsymbol{X}; \hat{\bar{\boldsymbol{\beta}}})$ at $\hat{\boldsymbol{\beta}}$. For more computation details, see Chapter 2.5 in Hosmer Jr *et al.* (2013). Note that for a continuous outcome, a linear regression assumes that the variance of the error term does not depend on the conditional mean. We can use heteroskedasticity-consistent standard errors in case the assumption does not hold. However, in logistic regression, we have binomial errors, and as a result, the variance is a function of the conditional mean thereof is heterogeneous by nature (Hosmer Jr *et al.* 2013).

2. The simulation approach (Chatton *et al.* 2020; Aalen *et al.* 1997), where $\hat{\beta} \sim \mathcal{N}(\hat{\boldsymbol{\beta}}, \hat{\mathbb{V}}(\hat{\boldsymbol{\beta}}))$. The method shows similar results to the bootstrap resampling but is much faster (Chatton *et al.* 2020; Aalen *et al.* 1997). We can simulate a set of parametric models from the distribution of $\hat{\boldsymbol{\beta}}$. Then the sample variance of predicted potential outcomes for each $\boldsymbol{x}_i$ from a set of simulated models is the estimated variance for $\mathbb{V}(p(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}}))$.

3. The Bayesian approach. We can use a Bayesian logistic regression to estimate the conditional mean of potential outcomes. Via the Bayesian approach, each parameter in a model is regarded as a random variable and follows a distribution. The posterior distribution of model parameters is approximated using a sampling approach, e.g., Markov chain monte carlo. The resulting predicted value of potential outcomes for each individual follows a similar distribution and the variance of the distribution can be estimated using the sample variance. In **RCTrep**, `G_computation_BART` and `G_computation_psBART` use the Bayesian approach to estimating the variance of estimates of the ATE.

4. Bootstrapping. Instead of resampling model parameters using the simulation approach, we can bootstrap a sample from a dataset, estimate $\hat{\boldsymbol{\beta}}$ based on the resampled data, repeat resampling multiple times, and compute the sample variance of predicted potential outcomes for each individual derived from the sampling distribution. The sample variance can be regarded as the estimation of the variance of $p(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}})$. This method, however, is of computational burden.

5. The sandwich style method using R package **geex**. The standard error of estimates of the ATE using the G-computation method can be computed directly by calling the function `geex::m_estimate(data, estFUN, ...)`. See Saul and Hudgens (2020) for more theoretical proof and implementation details. All `TEstimator` subclasses in **RCTrep** use this method to compute the variance of the ATE of population and subpopulations except for `G_computation_BART` and `G_computation_psBART`, and all `TEstimator` subclasses in **RCTrep** use this method to compute the variance of the weighted ATE of population and subpopulations.

The variance of estimates of the ATE is composed of the variance within groups (the first term in the third line of Equation 5) and the variance between groups (the second term in the third line of Equation 5). Via simulation approach, bayesian approach, and bootstrap approach, the variance of $p(\boldsymbol{x}, t; \hat{\boldsymbol{\beta}})$ within a group $\boldsymbol{X} = \boldsymbol{x}$ can be computed as follows:

$$\hat{\mathbb{V}}(p(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}})) = \frac{1}{D} \sum_{d=1}^{D} \left(p(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}}^d) - \bar{p}(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}}) \right)^2 \tag{7}$$

where D is the number of draws from the distribution of $\hat{\boldsymbol{\beta}}$, $\hat{\boldsymbol{\beta}}^d \sim \hat{p}(\hat{\boldsymbol{\beta}})$, $\bar{p}(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}}) = \frac{1}{D} \sum_{d=1} p(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}}^d)$, where $\hat{p}(\hat{\boldsymbol{\beta}})$ is the approximated empirical sampling distribution of $\hat{\boldsymbol{\beta}}$ using the simulation approach, the Bayesian approach, and the bootstrapping approach. Then

$$\mathbb{E}[\mathbb{V}(\hat{\tau}(\boldsymbol{X}) \mid \boldsymbol{X})] \approx \frac{1}{n} \sum_{i} \hat{\mathbb{V}}(p(\boldsymbol{x}_i, 1; \hat{\boldsymbol{\beta}})) + \hat{\mathbb{V}}(p(\boldsymbol{x}_i, 0; \hat{\boldsymbol{\beta}})) \tag{8}$$

by assuming $p(\boldsymbol{x}, 1; \hat{\boldsymbol{\beta}})$ is independent of $p(\boldsymbol{x}, 0; \hat{\boldsymbol{\beta}})$. Then we estimate the sample average of $\hat{\mathbb{V}}(\hat{\tau}(\boldsymbol{x}_i))$ as the estimate of the expectation of the variance of estimates of the ATE within groups. The variance of estimates of the ATE between groups (the second term in the last line of Equation 5) can be estimated as follows:

$$\mathbb{V}(\mathbb{E}[\hat{\tau}(\boldsymbol{X}) \mid \boldsymbol{X}]) \approx \frac{1}{n} \sum_{i=1} \left(p(\boldsymbol{x}_i, 1; \bar{\hat{\boldsymbol{\beta}}}) - p(\boldsymbol{x}_i, 0; \bar{\hat{\boldsymbol{\beta}}}) - \bar{p}(1; \bar{\hat{\boldsymbol{\beta}}}) - \bar{p}(0; \bar{\hat{\boldsymbol{\beta}}}) \right)^2 \tag{9}$$

where $p(\boldsymbol{x}_i, t; \bar{\hat{\boldsymbol{\beta}}}) = \frac{1}{D} \sum_{d=1} p(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}}^d)$ for the simulation approach, the Bayesian approach, and the bootstrapping approach, and $p(\boldsymbol{x}_i, t; \bar{\hat{\boldsymbol{\beta}}}) = p(\boldsymbol{x}_i, t; \hat{\boldsymbol{\beta}}), i \in \mathcal{S}$ for the model-based approach; $\bar{p}(t; \bar{\hat{\boldsymbol{\beta}}}) = \frac{1}{n} \sum_{i=1} p(\boldsymbol{x}_i, t; \bar{\hat{\beta}}), i \in \mathcal{S}$. Then the variance of estimates of the ATE in Equation 5 for the G-computation is the sum of the estimated variance of estimates of the ATE within groups in Equation 8 and the estimated variance of the estimate of the ATE between groups in Equation 9. The standard error of estimates of the ATE (i.e., the mean of $\hat{\tau}(\boldsymbol{X})$ of a sample) is $\frac{\hat{\mathbb{V}}(\hat{\tau}(\boldsymbol{X}))}{n}$ accordingly. Note that using the sandwich style standard error via **geex** can directly estimate the standard error of the estimate of the ATE without manually computing Equation 8 and 9.

C.2. The variance of estimates of the ATE using the IPW method

The propensity-score based method for the ATE estimation has a methodological advantage since it mimics a set-up of an RCT in which the treatment and control groups are balanced. The propensity score is defined as:

$$\pi_t(\boldsymbol{X}) = P(T = 1 \mid \boldsymbol{X}) \tag{10}$$

The IPW method weighs each individual by the inverse probability of receiving the observed treatment. In an RCT, the propensity score is known; in an observational study, the propensity score is unknown but can be estimated. The IPW method is defined as follows, where we use the self-normalized IPW estimator since it has a smaller variance (Swaminathan and Joachims 2015):

$$\hat{\tau} = \sum_{i:T_i=1} \hat{w}(\boldsymbol{x}_i) Y_i - \sum_{i:T_i=0} \hat{w}(\boldsymbol{x}_i) Y_i \tag{11}$$

where

$$\hat{w}(\boldsymbol{x}_i) = \begin{cases} \frac{\frac{1}{\pi_t(\boldsymbol{x}_i;\hat{\boldsymbol{\alpha}})}}{\sum_{i:T_i=1} \frac{1}{\pi_t(\boldsymbol{x}_i;\hat{\alpha})}} & T_i = 1 \\ \frac{\frac{1}{1-\pi_t(\boldsymbol{x}_i;\hat{\boldsymbol{\alpha}})}}{\sum_{i:T_i=0} \frac{1}{1-\pi_t(\boldsymbol{x}_i;\hat{\alpha})}} & T_i = 0. \end{cases}$$

The different modeling approaches can be used to model the propensity score, for instance, logistic regression, random forest, etc. The IPW method is unbiased and consistent as long as the propensity score model is correctly specified. The variance of the IPW method is approximated as:

$$\begin{aligned} \mathbb{V}(\hat{\tau}(\boldsymbol{X})) =& \mathbb{V}\left(\frac{YT}{\pi_t(\boldsymbol{X};\hat{\boldsymbol{\alpha}})} - \frac{Y(1-T)}{1-\pi_t(\boldsymbol{X};\hat{\boldsymbol{\alpha}})}\right) \\ =& \mathbb{E}\left[\mathbb{V}\left(\frac{YT}{\pi_t(\boldsymbol{X};\hat{\boldsymbol{\alpha}})} - \frac{Y(1-T)}{1-\pi_t(\boldsymbol{X};\hat{\boldsymbol{\alpha}})} \mid \boldsymbol{X}\right)\right] + \\ & \mathbb{V}\left(\mathbb{E}\left[\frac{YT}{\pi_t(\boldsymbol{X};\hat{\boldsymbol{\alpha}})} - \frac{Y(1-T)}{1-\pi_t(\boldsymbol{X};\hat{\boldsymbol{\alpha}})} \mid \boldsymbol{X}\right]\right) \\ \approx & \sum_{i:t_i=1}^{n} w_i^2 \hat{\sigma}_1^2(\boldsymbol{x}_i) + \sum_{i:t_i=0}^{n} w_i^2 \hat{\sigma}_0^2(\boldsymbol{x}_i) + \hat{\mathbb{V}}\left(\hat{\tau}(\boldsymbol{X};\bar{\hat{\boldsymbol{\alpha}}})\right) \end{aligned} \tag{12}$$

where $\sigma_1^2(\boldsymbol{x})$ and $\sigma_0^2(\boldsymbol{x})$ is the conditional variance of $Y(1)$ and $Y(0)$ given $\boldsymbol{x}$, which is unknown and estimable using the exact matching, and regression adjustment, etc., see Imbens and Rubin (2015) Chapter 19 for details. $\hat{\tau}(\boldsymbol{X};\bar{\hat{\alpha}}) \approx \hat{\tau}(\boldsymbol{X}_i;\hat{\alpha}) = \frac{Y_i T_i}{\pi_t(\boldsymbol{X}_i;\hat{\alpha})} - \frac{Y_i(1-T_i)}{1-\pi_t(\boldsymbol{X}_i;\hat{\alpha})}$, $\hat{\mathbb{V}}(\hat{\tau}(\boldsymbol{X};\bar{\hat{\boldsymbol{\alpha}}}))$ is the sample variance of $\hat{\tau}(\boldsymbol{X};\bar{\hat{\boldsymbol{\alpha}}})$.

The standard error of estimates of the ATE (i.e., the mean of $\hat{\tau}(\boldsymbol{X})$ of a sample) is $\frac{\hat{\mathbb{V}}(\hat{\tau}(\boldsymbol{X}))}{n}$ accordingly. **RCTrep** uses the sandwich style standard error via **geex** to estimate the variance of the estimate of the ATE using the IPW method. It is clear to see that the variance of estimates of the ATE using the IPW method depends on the variance of estimated weights, which can inflate the variance of the estimate of the ATE if there are extreme values of weights. Hence, the IPW method can suffer from near violation of the T-overlap assumption. To have a good estimation of the variance of the estimate of the ATE, we should try to keep the dependence of $w(\boldsymbol{x}_i)$ as mild as possible. On one hand, we can reduce the variability of weights using related approaches (Dong *et al.* 2020; Chattopadhyay *et al.* 2020; Zeng *et al.* 2021) through optimizing an objective function, which aims to minimize the variability of all weights while preserving balance in weighted covariates between groups; on the other hand, we can exclude covariates which are merely associated with the treatment assignment in a propensity score modeling, since balancing over these covariates will decrease the sample size in each subgroup hence can inflate the estimation of the variance. Beyond confounders, other covariates which are predictive of outcomes can be adjusted in propensity score models, which may improve the precision of estimates of the ATE(Chatton *et al.* 2020).

C.3. The variance of estimates of the ATE using the DR method

The DR method combines a propensity-score model and an outcome model such that the method is unbiased and consistent if one of these two models is correctly specified, hence it offers protection against missmodeling. The DR method gains in precision of the estimation over the IPW method, however, may not be as precise as the G-computation method when the outcome model is correctly specified (or has good approximation) (Lunceford and Davidian 2004). The study by Kang and Schafer (2007) indicates that when both models are incorrect but neither is grossly misspecified, many DR methods perform better than the IPW methods, however, none of the DR methods tried in the study outperformed an outcome regression model. Although the study does not represent all scenarios of the data generation mechanism, the study does demonstrate that, in at least some settings, two wrong models may not be better than one. The DR method for the ATE estimation is demonstrated as follows:

$$\mathbb{E}[\hat{\tau}(\boldsymbol{X})] = \frac{1}{n}\sum_i \left(p(\boldsymbol{x}_i, 1; \hat{\boldsymbol{\beta}}) + \frac{T_i}{\pi_t(\boldsymbol{x}_i; \hat{\alpha})}\epsilon_i^1 \right) - \frac{1}{n}\sum_i \left(p(\boldsymbol{x}_i, 0; \hat{\boldsymbol{\beta}}) + \frac{(1-T_i)}{1-\pi_t(\boldsymbol{x}_i; \hat{\alpha})}\epsilon_i^0 \right) \tag{13}$$

where $\epsilon_i^1 = Y_i - p(\boldsymbol{x}_i, 1; \hat{\boldsymbol{\beta}})$ and $\epsilon_i^0 = Y_i - p(\boldsymbol{x}_i, 0; \hat{\boldsymbol{\beta}})$. The variance of the DR method is derived as follows:

$$\begin{aligned}
\mathbb{V}(\hat{\tau}(\boldsymbol{X})) = {} & \mathbb{E}\left[\mathbb{V}\left(p(\boldsymbol{X}, 1; \hat{\boldsymbol{\beta}}) + \frac{Z}{\hat{\pi}_t(\boldsymbol{X})}\epsilon_i^1 - p(\boldsymbol{X}, 0; \hat{\boldsymbol{\beta}}) - \frac{1-T}{1-\hat{\pi}_t(\boldsymbol{X})}\epsilon_i^0 \mid \boldsymbol{X} \right) \right] + \\
& \mathbb{V}\left(\mathbb{E}\left[p(\boldsymbol{X}, 1; \hat{\boldsymbol{\beta}}) + \frac{T}{\hat{\pi}_t(\boldsymbol{X})}\epsilon_i^1 - p(\boldsymbol{X}, 0; \hat{\boldsymbol{\beta}}) - \frac{1-T}{1-\hat{\pi}_t(\boldsymbol{X})}\epsilon_i^0 \mid \boldsymbol{X} \right] \right) \\
\approx {} & \frac{1}{n}\sum_i^n \hat{\mathbb{V}}\left(p(\boldsymbol{x}_i, 1; \hat{\boldsymbol{\beta}}) \right) + \hat{\mathbb{V}}\left(p(\boldsymbol{x}_i, 0; \hat{\boldsymbol{\beta}}) \right) + \\
& \frac{1}{n_1}\sum_{i:T_i=1} w_i^2 \hat{\sigma}_1^2(\boldsymbol{x}_i) + \frac{1}{n_0}\sum_{i:T_i=0} w_i^2 \hat{\sigma}_0^2(\boldsymbol{x}_i) + \hat{\mathbb{V}}\left[\hat{\tau}(\boldsymbol{X}; \bar{\hat{\boldsymbol{\beta}}}, \bar{\hat{\boldsymbol{\alpha}}}) \right]
\end{aligned} \tag{14}$$

Similar to the variance of the IPW method and the variance of the G-computation method, $\hat{\mathbb{V}}(p(\boldsymbol{x}, t; \hat{\boldsymbol{\beta}}))$, can be estimated using the model-based, the simulation-based, the Bayesian, or the bootstrapping method, and $\hat{\sigma}_1^2(\boldsymbol{x}_i)$ and $\hat{\sigma}_0^2(\boldsymbol{x}_i)$ can be estimated using the exact matching, regression adjustment approaches, etc.. The standard error of estimates of the ATE is $\frac{\hat{\mathbb{V}}(\hat{\tau}(\boldsymbol{X}))}{n}$. In **RCTrep**, we use the sandwich style method in **geex** to estimate the standard error of estimates of the ATE using the DR method.

C.4. The variance of estimates of the ATE using the difference in means of outcomes between groups

In this section, we demonstrate the variance of estimates of the ATE using the crude estimator, i.e., the difference in means of outcomes between treatment and control groups. The variance is as follows:

$$\begin{aligned}
\mathbb{V}(\hat{\tau}(\boldsymbol{X})) &= \mathbb{V}\left(\frac{1}{n_1}\sum_{i:T_i=1} Y_i(1) - \frac{1}{n_0}\sum_{i:T_i=0} Y_i(0) \right) \\
&= \frac{1}{n_1^2}\sum_{i:T_i=1} \sigma_1^2(\boldsymbol{x}_i) + \frac{1}{n_0^2}\sum_{i:T_i=1} \sigma_0^2(\boldsymbol{x}_i) \\
&\approx \frac{\hat{\sigma}_1^2}{n_1} + \frac{\hat{\sigma}_0^2}{n_0}
\end{aligned} \tag{15}$$

Under simplifying the assumption of homoscedasticity, i.e., $\sigma_1^2(\boldsymbol{x}) = \sigma_1^2$ and $\sigma_0^2(\boldsymbol{x}) = \sigma_0^2$ are constants across individuals, σ_1^2 and σ_0^2 can be estimated by the sample variance of $Y(1)$ in the treatment group and the sample variance of $Y(0)$ in the control group. We also assume observed outcomes Y_i are mutually independent, namely, the observed outcome of each individual does not depend on the observed outcome of another individual. Since $\mathbb{V}(Y \mid \boldsymbol{X}) = \mathbb{V}(Y)(1-\rho)$ where ρ is the correlation between Y and $\boldsymbol{X}$, the estimated variance of estimates of the ATE in Equation 15 is conservative, and can gain efficiency if the variance of potential outcomes for each individual is estimated conditioning on covariates $\boldsymbol{X}$ that are predictive of outcomes. The standard error of estimates of the ATE is $\frac{\hat{\mathbb{V}}(\hat{\tau}(\boldsymbol{X}))}{n}$.

D. Methods for adjusting the sampling mechanism

In this section, we elaborate three methods used in **RCTrep** to adjust for the sampling mechanism, 1) exact matching; 2) inverse sampling score weighting; 3) subclassification.

D.1. Exact matching

In this section, we introduce weighting based on $\boldsymbol{X}_s$ where weights are estimated using the exact matching. This weighting approach is similar to importance sampling/transfer learning/domain adaption/covariate shift, which balances the distribution of $\boldsymbol{X}_s$ between two samples (see Stuart 2010, for more details). Given assumptions on the sampling mechanism, $\mathcal{S}^{obs}$ and $\mathcal{S}^{rct}$ can be regarded as two random samples from a target population $\mathcal{P}_{\boldsymbol{\theta}}$. Then weights are defined as:

$$w(\boldsymbol{x}_{si}) = \frac{w'(\boldsymbol{x}_{si})}{\sum_{i \in \mathcal{S}^{obs}} w'(\boldsymbol{x}_{si})}, \quad \sum_{i \in \mathcal{S}^{obs}} w(\boldsymbol{x}_{si}) = 1, w'(\boldsymbol{x}_{si}) = \frac{\hat{p}(\boldsymbol{x}_s)}{\hat{q}(\boldsymbol{x}_s)} \tag{16}$$

where $\hat{p}(\boldsymbol{x}_s)$ and $\hat{q}(\boldsymbol{x}_s)$ are empirical densities of $\boldsymbol{X}_s$ in $\mathcal{S}^{rct}$ and $\mathcal{S}^{obs}$, respectively.

D.2. The inverse selection probability weighting

The selection probability is the conditional probability of being selected to an RCT data given covariates $\boldsymbol{X}_s$, which is defined as follows:

$$\pi_s(\boldsymbol{X}_i) = P(S = 1 \mid \boldsymbol{X}_{si}) \tag{17}$$

where $S = \{0, 1\}$, 1 indicates selection to $\mathcal{S}^{rct}$ and 0 indicates selection to $\mathcal{S}^{obs}$. In most of cases, the selection probability is unknown but could be estimated from a combined dataset. In **RCTrep**, we consider an RCT dataset as a simple random sample from a target population $\mathcal{P}_\theta$ and we regard an observational dataset as a sample drawn from the target population via a selection probability. We weight individuals in $\mathcal{S}^{obs}$ according to the odds of their selection probabilities. The resulting weighted dataset of $\mathcal{S}^{obs}$ resembles a simple random sample from the $\mathcal{P}_\theta$. Hence the weight for each individual are:

$$w_i = \begin{cases} \frac{\pi_s(\boldsymbol{x}_{si})}{1-\pi_s(\boldsymbol{x}_{si})} & S_i = 0 \\ 1 & S_i = 1 \end{cases}$$

According to Rosenbaum and Rubin (1983), the ignorability assumption holds conditioning on a balance score. The selection probability is the "coarsest" balancing score, $\boldsymbol{X}_s$ is the

"finest" balancing score. Any balancing score finer than the selection probability can allow the ignorability assumption holds. A selection probability is a propensity score when we adjust for "confounding" due to an unknown sampling mechanism.

D.3. Subclassification

Individuals are assigned to a class according to a distance measure, for instance, the selection probability $p(S = 1 \mid \boldsymbol{X}_s)$. In **RCTrep**, $\mathcal{S}^{obs}$ and $\mathcal{S}^{rct}$ data are assigned into classes based on quantiles of the selection probability of $\mathcal{S}^{rct}$. Weights are computed based on the proportion of individuals in $\mathcal{S}^{rct}$ in each class. For more details, see the function `matchit()` in the R package **MatchIt**. Many modeling approaches are provided in **RCTrep** for estimating the selection probability, for instance, `glm`, `gbm`, `lasso`.

D.4. Variance of the weighted ATE

We can treat $w(\boldsymbol{x}_{si})$ as a fixed value for each individual, and use a standard sandwich style variance estimator via R packages **geex** or **survey**. However, it is important to consider that these weights are estimated and are unknown. Buchanan *et al.* (2018) derived a variance estimator that accounts for the variance of weights when these weights are unknown; Ackerman *et al.* (2021) used a double bootstrap method to estimate the variance of weighted estimates, where both RCT data and observational data are resampled with a replacement prior. This approach, however, is computationally intensive, and results are very similar to the sandwich style variance estimator.

E. A structual causal diagram of data used throughout the paper

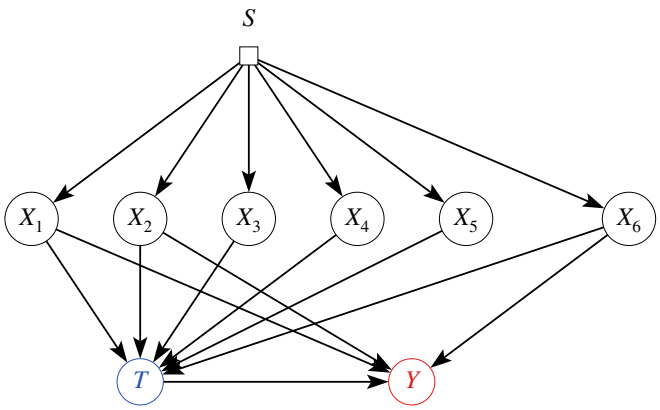


Figure 9: A structural causal diagram representing the treatment T, the outcome Y, the selection S and other predictors of the outcome. The diagram visualizes the data generation mechanism of the data used in the paper. The figure is generated using the software **causalfusion** (Bareinboim and Pearl 2016). The diagram shows that `x3,x4,x5` are not predictive of the outcome; and `x2` and `x6` are predictive of treatment effects based on the data generation mechanism. According to the back-door criteria, the minimal `outcome_predictors` and `selection_predictors` that allow the assumption of T-ignorability and the assumption of S-ignorability hold are `x1,x2,x6` and `x2,x6`. Adjusting `x3,x4,x5` can inflate the variance of estimates of the ATE and adjusting `x1,x3,x4,x5` can inflate the variance of weights.

.

F. Overview of the package

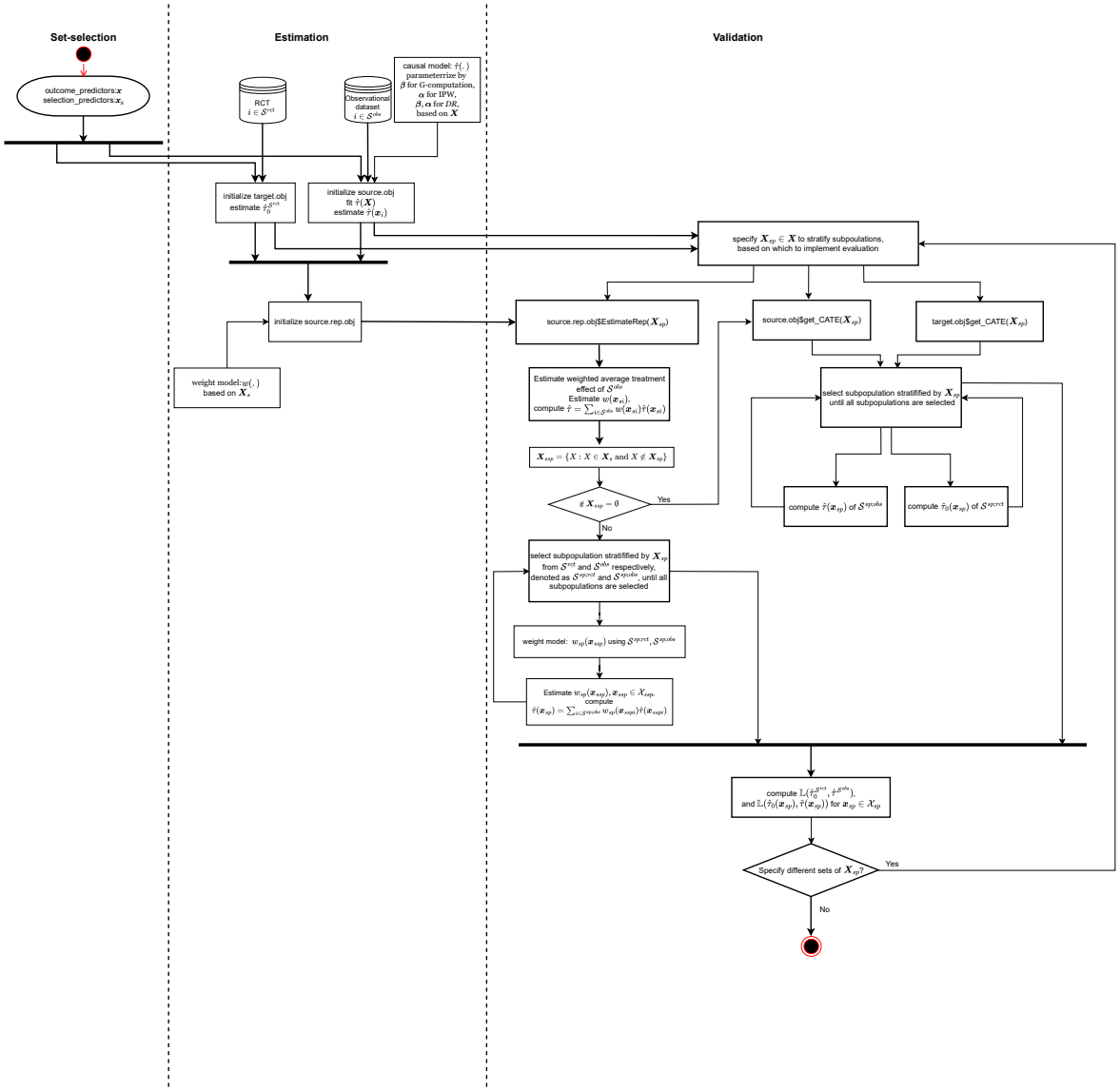


Figure 10: The set up of the approach to the assessment of the validity of estimates of the ATE, in which unbiased estimates of the ATE of population and subpopulations are obtained from an RCT dataset.

G. Descriptions of the function for generating synthetic RCT data

Arguments	Description
`margin_dis`	A character specifying the distribution of each covariate, allowable options are `"bernoulli_categorical"` and `"bernoulli"`. `"bernoulli_categorical"` indicates covariates with more than two categories; `"bernoulli"` indicates covariates with two categories.
`N`	A numeric value indicating the sample size for returned data.
`margin`	A list containing p named elements. The names of these elements are covariate names. If `margin_dis="bernoulli_categorical"`, each element is a vector with a character indicating a covariate name, the number of levels of the covariate, the value of each level, and the proportion of each level; if `margin_dis="bernoulli"`, each element is the proportion of the positive value of each covariate.
`var_name`	A character vector indicating names of covariates. These names should be in line with names of elements in `margin`.
`pw.cor=0`	A vector containing the pairwise correlations of specified covariates in `var_name`. When `margin_dis="bernoulli"`, `pw.cor` must be specified. The default value is 0.

Table 6: Descriptions of the input arguments of the function `GenerateSyntheticData()`.

Affiliation:

Lingjie Shen
Department of Methodology and Statistics
Tilburg University
5037 AB Tilburg, The Netherlands E-mail: L.Shen@uvt.nl