Improving Causal Inference from Observational Data: A Comparative Analysis of Confidence Interval Methods in Sequential Target Trial Emulation

Florian Metwaly, Oisín Ryan†, Wouter van Amsterdam†

Student Number: 778265.

12th May 2025.

†Supervisors

Abstract

Background: Sequential Target Trial Emulation (TTE) is a framework that is gaining popularity to estimate causal effects from observational data when a randomized control trial (RCT) is not feasible. As sequential TTE involves copying data from the observational dataset multiple times, the assumption of independence between observations is violated, and confidence intervals (CI) can no longer be estimated by standard estimation approaches. While sandwich-type estimators are commonly implemented in statistical software for CI estimation, the non-parametric bootstrap is the recommended method for CI estimation in the TTE literature. We compare the performance of the two approaches and present the `Julia` package `TargetTrialEmulation.jl`, for computationally efficient estimation of bootstrap CIs in the context of sequential TTE.

Methods: In a simulation study, we compare the performance of the sandwich-type against two types of non-parametric bootstrap CI estimation: the percentile and the empirical bootstrap. We test the estimators across 81 scenarios, varying the sample sizes, confounding strength, outcome prevalence, and treatment prevalence. We compare their performance regarding their coverage, width, bias-eliminated coverage, and power.

Results: Our simulation study finds that both bootstrap methods, percentile and empirical, more frequently achieved coverage closer to the nominal 95% level, particularly in small sample sizes, and produced narrower confidence intervals than the sandwich estimator. Although the sandwich-type estimator found consistently higher statistical power, this finding is most likely attributable to the

sandwich-type estimator being biased upwards. Bias-eliminated coverage analysis suggested that coverage differences were primarily driven by bias in the point estimates, particularly at low event rates and later follow-up times.
Conclusion: While this study confirms bootstrap's potential, limitations of the percentile and empirical approaches, such as sensitivity to skewed distributions and a biased estimator, suggest the need for more robust methods. Advanced techniques like the bias-corrected and accelerated (BCa) bootstrap may provide improved inference in complex TTE settings. Future work should focus on benchmarking these methods using the evaluation framework employed in this study, coverage, width, and bias-eliminated coverage. It should additionally incorporate assessments of power and type I error control, and further focus on computational efficiency.

Keywords: Sequential Target Trial Emulation, Bootstrap, Sandwich Estimator, Observational Data, Causal Inference

1 Introduction

Randomised Control Trials (RCT) are the gold standard of causal inference. However, due to practical and ethical constraints, conducting an RCT is not always feasible (Hernán and Robins 2016; Sanson-Fisher et al. 2007). Gaining causal insights from observational data has been a long prevailing topic of research (Wold 1956; Nichols 2007). In health sciences, large-scale observational datasets such as electronic health record databases are increasingly used for causal analyses when RCTs are not feasible (Bakker et al. 2021).

When randomization is not possible or available, confounding bias must be addressed, for example, by measuring and statistically adjusting for confounding variables. Common methods are using matching or weighting methods for observed confounding (Schafer and Kang 2008) or adjusting the design for unobserved confounding (Igelström et al. 2022; Listl et al. 2016).

Increasingly, researchers in the health sciences have recognized that while mimicking randomization and adjusting for confounding are necessary steps, they are not sufficient to draw valid causal inferences from observational data. The reason for this is that, when analyzing observational data, the researcher is free to make design choices that may accidentally introduce different forms of selection biases or lead to a mismatch between the target quantity being estimated and the quantity of clinical interest (Fu 2023).

Immortal time bias, a type of bias that occurs when a period of time during which the event of interest cannot occur is incorrectly classified in the analysis, is a specific form of selection bias often seen in observational studies (Yadav and Lewis 2021; Hernán et al. 2025b). For example, in a vaccination study, individuals are enrolled in the study and they are eligible for vaccination from this time point on. However, the vaccine administration often occurs a few days after enrollment. Suppose this period between enrollment and vaccination is misclassified as time during which the individual is unvaccinated and not accounted for in the analysis. In that case, it results in an

artificially lower infection rate for the vaccinated group. This is because during this interval, the patient cannot get infected with COVID-19, effectively rendering them "immortal" with respect to the outcome. Such misclassification is typically referred to as immortal time bias (Hernán et al. 2025b).

1.1 Target Trial Emulation

Target Trial Emulation (TTE) is a methodological framework for designing observational data analyses to enable valid causal inference. TTE has specifically been developed to prevent the introduction of biases through research design. To perform TTE, one must first specify the ideal randomized control trial, the so-called target trial, which one would perform if feasible. The design of the TTE then mimics this idealized RCT as closely as possible (Hernán et al. 2008, 2025a). For this, a detailed trial protocol is typically specified (Hernán et al. 2008, 2025a).

A key concept of TTE is that of t_0 alignment. The t_0 describes the moment at which eligibility is determined, treatment is assigned and started, and the follow-up period starts (Hernán et al. 2008, 2025a). Extending the COVID-19 vaccine example, EHR data could be used to study the effectiveness of the COVID-19 vaccine. Suppose a dataset that includes patient data from January 1st, 2020, to December 31st, 2022, including their vaccination status, an indicator of whether they are eligible to be vaccinated, possible confounders, and an indicator of whether COVID-19 occurred or not. To emulate the target trial, the data can be analyzed by choosing an enrollment date, for example, January 1st, 2020. Every eligible individual on January 1st, 2020, gets "enrolled" in the trial, and their follow-up time starts. Individuals vaccinated on this date now form the treatment group; unvaccinated individuals form the control group. After the previously defined end date, say December 31st, 2022, the resulting subset can be used to estimate causal effects from this dataset. Notably, the model still has to adjust for confounding, for example, by using inverse propensity score weighting, which is not discussed further at this point.

To align t_0, a critical design choice must be made (Fu et al. 2025). In the COVID-19 vaccine example, a single t_0 is selected as the starting point of the follow-up time. However, this potentially leads to losing a large number of observations, due to them not being eligible at the chosen t_0, and can result in very few observations in the treatment arm, as individuals who receive the vaccination at a later time point are excluded from the analysis. Alternatively, the researcher can select *all* time points. When opting for this option, we speak of sequential Target Trial Emulation.

1.2 Sequential Target Trial Emulation

The core idea behind sequential Target Trial Emulation is that the specific date at which the trial starts is not of primary interest. Therefore, it is possible to iterate through all observed time points and start a new trial with all eligible individuals and their follow-up observations at each time point. Returning to the vaccination example, the eligible individuals on January 1st will now define the first trial. The same process will now be repeated for January 2nd. All individuals who are eligible on January 2nd and their follow-up times are selected and enrolled in the second trial. This process

is repeated for all observation points, forming a large dataset containing all emulated trials, with any individual possibly being part of multiple trials. By reassigning t_0 at every eligible time point, each separate trial includes only the correctly aligned follow-up period, excluding any interval during which the outcome, like the COVID-19 infection, could not occur due to the individuals being in an "immortal" phase. This prevents the introduction of immortal time bias (Hernán et al. 2025b,a). To clearly illustrate the construction of the sequentially emulated dataset, pseudocode is provided in Appendix A.

1.3 Variance Estimation

Two problems arise for the variance estimation in a sequential TTE setting. The first issue arises from sequential TTE generating multiple trials over time by repeatedly applying the trial eligibility criteria at each time point. In practice, any individual may be subject to numerous trials, which violates the standard assumption of independence between observations, as any individual potentially contributes data to multiple emulated trials (Hernán and Robins 2016). This dependency is an inherent consequence of the sequential TTE framework.

Second, as mentioned earlier, a common method to account for confounding bias is using inverse probability of treatment weighting (IPTW). Additionally, time-to-event data typically contains censored observations. Those are also accounted for when performing an analysis using the sequential TTE framework through inverse probability of censor weighting (IPCW) (Hernán et al. 2008). Censoring itself is not further discussed in this work, as the specific form of weighting is not central to the methodological evaluation; rather, the focus lies on the general impact of using inverse probability weights (IPW). As IPWs are often unstable, they introduce variability into the estimates, especially when the probabilities of treatment or censoring are close to 0 or 1. This variability can lead to underestimated standard errors, resulting in overly narrow confidence intervals and overconfident conclusions if not appropriately addressed (Austin and Stuart 2015; Austin 2016). While this problem is not inherent to the sequential TTE design, inverse probability weighting is the recommended, and in practice most often applied approach, to adjust for confounding and censoring (Hernán et al. 2025a, 2008).

A common solution to these issues is using sandwich-type variance estimators, as proposed by Lin and Wei (1989), which provide robust standard error estimates that remain valid under such complexities, and are commonly used to construct confidence intervals (CI). Due to its common integration in statistical software (Hernan and Robins 2020), it is a frequently used method in the literature (see, e.g. Wu et al. (2025); Li et al. (2024); Scola et al. (2023)).

Although the sandwich-type variance estimator is commonly used in practice, the TTE literature typically recommends using a non-parametric bootstrap approach to estimate the variance and confidence intervals (Maringe et al. 2020; Hernán and Robins 2016; Bakker et al. 2021). This is due to previous findings, for example in the survival analysis literature, where bootstrapping CIs in comparison with unadjusted

approaches or sandwich-type estimators, were found to produce approximately correct standard errors and correct coverage rates for CIs, outperforming the unadjusted model and sandwich-type estimators (Austin 2016).

The non-parametric bootstrap is a powerful tool for inference in cases of violation of various assumptions. Bootstrapping is a resampling technique that involves repeatedly sampling with replacement from the observed data to generate new datasets, which are then used to estimate the sampling distribution of a statistic. Due to the repeated resampling, bootstrap confidence intervals can be computationally intensive, posing practical challenges in large-scale settings such as EHR data. While computational optimizations for the bootstrap exist at the cost of additional assumptions, we do not consider them further here (see, e.g. Li and Lawson (2024); Binder et al. (2004)).

In this paper, we systematically evaluate the properties of non-parametric bootstrap confidence intervals within the sequential TTE framework, comparing them to widely used sandwich-type estimators under realistic simulation settings. In Section 2 we describe the model on which the sequential TTE framework is applied and how confidence intervals can be estimated. Further in Section 2.3 we present a software implementation of bootstrapped confidence intervals. Section 2.4 describes the simulation study which we use to evaluate the performance of the non-parametric bootstrap compared to sandwich-type estimators. The simulation study results are presented in Section 3 and discussed in Section 4.

2 Methods

In the following section, we describe the methodological framework used to evaluate the performance of the non-parametric bootstrap in comparison to sandwich-type estimators for estimating confidence intervals in a sequential TTE setting. Further, we introduce the `TargetTrialEmulation.jl` package, a practical implementation of sequential TTE in the `Julia` programming language. Finally, we describe the simulation study designed to assess and compare the performance of both variance estimation approaches across a range of data-generating scenarios.

2.1 Defining the Estimand

Two types of analysis are common in clinical research: the intention-to-treat (ITT) and per-protocol (PP) analyses. In a PP analysis, individuals are analyzed according to the treatment they actually received, excluding those who deviated from the study protocol. In an ITT analysis, the individuals are analyzed according to their initially assigned treatment, regardless of subsequent treatment changes (Tripepi et al. 2020). As the validity of bootstrap confidence intervals has already been investigated in a PP analysis (Limozin et al. 2024), the following study will compare the sandwich-type estimator and bootstrap confidence intervals in the context of an ITT analysis.

We assume a setting in which a single individual is observed regularly over a set period of time K. The start of follow-up time is denoted by $k = 0$. Information about treatment status A_k, eligibility E_k, time-varying confounders L_k, and the binary outcome of interest Y_k is collected at each timepoint $k = 0, 1, 2, \ldots, K$. Time-invariant confounders are denoted by V, and time-varying confounders are fixed to their baseline

values and denoted by L_0, due to performing an ITT analysis. Treatment status A_k is binary, taking values $a \in \{0, 1\}$, representing treated and untreated, respectively. Once an individual experiences the event ($Y_k = 1$), they are no longer followed and are excluded from the dataset in subsequent time points.

The cumulative survival distribution under treatment strategy a, denoted by $S_a(k)$, represents the probability of surviving through time point k under the respective treatment strategy. This distribution is derived by estimating the hazard of the event at each time point using a pooled logistic regression model, and then computing the product of one minus the estimated hazards over time

$$S_a(k \mid V_i, L_{m,0,i}, \hat{\boldsymbol{\beta}}) = \prod_{j=0}^{k} \left\{ 1 - \text{logit}^{-1} \left[\mu \left(j, m, a, V_i, L_{m,0,i}; \hat{\boldsymbol{\beta}} \right) \right] \right\}, \tag{1}$$

where $\text{logit}^{-1}(\cdot) = \exp(\cdot)/\{1 + \exp(\cdot)\}$ denotes the logistic function and $\hat{\boldsymbol{\beta}}$ denotes the parameters estimated in Equation 3.

The linear predictor $\mu(\cdot)$ is defined as

$$\mu \left(j, m, a, L_{m,0}, V; \hat{\boldsymbol{\beta}} \right) = \hat{\beta}_0(m) + \hat{\beta}_1(j) + \hat{\beta}_2 \cdot a + \hat{\boldsymbol{\beta}}_3^{\mathrm{T}} V + \hat{\boldsymbol{\beta}}_4^{\mathrm{T}} L_{m,0}, \tag{2}$$

for $j = 0, ..., k$, where $\hat{\beta}_0(m)$ denotes the intercept term for trial m, accounting for trial-specific baseline risks, $\hat{\beta}_1(j)$ denotes the effect of time $j = 0, ..., k$, $\hat{\beta}_2$ denotes the average treatment effect and $\hat{\boldsymbol{\beta}}_{\mathbf{3}}$ and $\hat{\boldsymbol{\beta}}_{\mathbf{4}}$ denote coefficients associated with covariates V_i and $L_{m,0,i}$.

The pooled logistic regression model to estimate the parameters $\hat{\boldsymbol{\beta}}$ takes the form

$$\begin{aligned} &\text{logit} \left(Pr(Y_{j,m,i} = 1 \mid Y_{j-1,m,i} = 0, a_i, L_{m,0,i}, V_i) \right) = \\ &\beta_0(m) + \beta_1(j) + \beta_2 \cdot a_i + \boldsymbol{\beta}_3^{\mathrm{T}} V_i + \boldsymbol{\beta}_4^{\mathrm{T}} L_{m,0,i}, \end{aligned} \tag{3}$$

where $Y_{j,m,i}$ denotes the outcome for patient i at time j in trial m, and all other terms are defined as above. This approach corresponds to a discrete-time hazard model estimated via pooled logistic regression (Zivich et al. 2025).

The causal estimand of interest is the average treatment effect (ATE) at each timepoint k

$$\Pr\left(Y_k^{a=1} = 1\right) - \Pr\left(Y_k^{a=0} = 1\right) \tag{4}$$

(Schafer and Kang 2008).

The difference between the estimated effect on the treated group and the untreated group constructs the ATE. This estimator is called the marginal risk difference (MRD) (Keogh et al. 2023; Hernán 2010). The MRD quantifies the average difference in outcome probabilities between treated and untreated groups, adjusted for covariates, and averaged across multiple sequential trials. The MRD per trial m for the model specified

in Equation 1 is specified by

$$\begin{aligned}\widehat{\mathrm{MRD}}_m(k) =& \frac{1}{n_m}\sum_{i=1}^{n} E_{m,i}\prod_{j=0}^{k}\left\{1-\mathrm{logit}^{-1}\left[\mu\left(j,m,a=0,V_i,L_{m,0,i};\hat{\boldsymbol{\beta}}\right)\right]\right\} \\ &- \frac{1}{n_m}\sum_{i=1}^{n} E_{m,i}\prod_{j=0}^{k}\left\{1-\mathrm{logit}^{-1}\left[\mu\left(j,m,a=1,V_i,L_{m,0,i};\hat{\boldsymbol{\beta}}\right)\right]\right\},\end{aligned} \tag{5}$$

where i is the patient index of the original observational dataset, $n_m = \sum_{i=1}^{n} E_{m,i}$ describes the total number of patients enrolled in trial m.

The final MRD for each time point k is then the unweighted average $\widehat{\mathrm{MRD}}$ over all m trials

$$\widehat{\mathrm{MRD}}(k) = \frac{1}{m}\sum_{i=1}^{m}\widehat{\mathrm{MRD}}_m(k) \tag{6}$$

The causal assumptions of the MRD estimator are discussed in Appendix B.

2.2 Variance Estimation Methods

As mentioned, the commonly implemented variance estimation method in statistical software is the sandwich-type estimator. The sandwich-type variance estimator, as implemented in the `R` package `TrialEmulation`, is defined as

$$\hat{\Sigma} = \left\{\sum_{i=1}^{n}\frac{\partial \boldsymbol{U}_i(\hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}^{\mathrm{T}}}\right\}^{-1}\left\{\sum_{i=1}^{n}\boldsymbol{U}_i(\hat{\boldsymbol{\beta}})\boldsymbol{U}_i(\hat{\boldsymbol{\beta}})^{\mathrm{T}}\right\}\left\{\sum_{i=1}^{n}\frac{\partial \boldsymbol{U}_i(\hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}^{\mathrm{T}}}\right\}^{-1}, \tag{7}$$

with $\hat{\boldsymbol{\beta}}$ being the estimates of the $\boldsymbol{\beta}$ parameter of the model defined in Equation 3, and $\boldsymbol{U}_i(\hat{\boldsymbol{\beta}})$ being the score function of the model evaluated at $\hat{\boldsymbol{\beta}}$. For further details, see Lin and Wei (1989). The confidence intervals are then obtained simulation-based, by drawing the model parameters from a multivariate normal distribution $\boldsymbol{\beta} \sim \mathcal{N}(\hat{\boldsymbol{\beta}}, \hat{\Sigma})$ and evaluating the $\widehat{\mathrm{MRD}}$ repeatedly based on the drawn parameters. For a detailed description, see Su et al. (2024) and Mandel (2013).

We evaluate a non-parametric bootstrap approach to estimate the confidence interval around the $\widehat{\mathrm{MRD}}$ estimator. In sequential TTE, bootstrap confidence intervals are obtained by resampling from the baseline observations to create a new observational dataset, which includes both the baseline observations and their corresponding follow-up times. Sequential TTE is then performed on this resampled dataset. The resampling procedure is outlined in Algorithm 1.

We estimate two types of confidence intervals from the empirical distribution returned by Algorithm 1. The *percentile bootstrap confidence interval* is defined for each trial visit k as

$$CI^{\mathrm{pct}}_{\mathrm{low}}(k) = \widehat{\mathrm{MRD}}^{*}_{(0.025)}, \tag{8}$$

$$CI^{\mathrm{pct}}_{\mathrm{high}}(k) = \widehat{\mathrm{MRD}}^{*}_{(0.975)}, \tag{9}$$

Algorithm 1 Non-Parametric Bootstrap Procedure (Carpenter and Bithell 2000)

1: **for** $b = 1$ to B **do**
2: Sample n observations with replacement from Y_0 to obtain $Y^{*(b)}$
3: Compute the bootstrap statistic $h_K^{*(b)} = h_K(Y^{*(b)})$
4: **end for**
5: **return** Empirical distribution of $\{h_K^{*(1)}, h_K^{*(2)}, \ldots, h_K^{*(B)}\}$

where $\widehat{\text{MRD}}^*_{(p)}$ denotes the p-th percentile of the bootstrap distribution.

The second type, which we refer to as the *empirical confidence interval*, also known as the non-Studentized pivotal or basic bootstrap interval, is evaluated at each trial visit k as

$$CI^{\text{emp}}_{\text{low}}(k) = 2\,\widehat{\text{MRD}}(k) - \widehat{\text{MRD}}^*_{(0.975)}\,, \tag{10}$$

$$CI^{\text{emp}}_{\text{high}}(k) = 2\,\widehat{\text{MRD}}(k) - \widehat{\text{MRD}}^*_{(0.025)}\,. \tag{11}$$

Both confidence interval methods follow Carpenter and Bithell (2000).

2.3 TargetTrialEmulation.jl

As, to our knowledge, there currently is no software implementation that allows researchers to estimate bootstrap confidence intervals easily in the context of sequential TTE, we present the `Julia` package `TargetTrialEmulation.jl` (Metwaly 2025).

The `TargetTrialEmulation.jl` package estimates confidence intervals using the bootstrapping approaches described in Section 2.2. Due to resampling, bootstrap confidence intervals impose a heavy computational constraint. This constraint adds to sequential TTE itself, which is already a computationally demanding task, by potentially copying millions of observations many times.

The `R` programming language is widely used for data analysis in the health sciences, especially due to its broad package ecosystem. However, its performance limitations, particularly in terms of memory management and execution speed, pose challenges in large-scale applications such as sequential TTE. These limitations can lead to excessive memory usage and long runtimes when processing large datasets.

In contrast, `Julia` is a modern programming language designed for efficient memory handling and fast compilation of code. `Julia` is built on the principles of running as fast as a low-level programming language, with an efficient memory handling system, while keeping the syntax easy and comprehensive (Karpinski et al. 2012). `Julia`'s just-in-time (JIT) compiler ensures that code is compiled and optimized at runtime, allowing it to execute as efficiently as compiled languages. Moreover, `Julia`'s efficient memory management system minimizes memory overhead and fragmentation (Bezanson et al. 2017).

These features motivate our choice to write a `Julia` package to perform sequential TTE, in addition to implementing and testing bootstrap confidence intervals. `Julia`'s potential performance improvements over `R` and other programming languages

have previously been shown, for example, by van Amsterdam (2024). For a speed comparison of `R` and `Julia` in the context of sequential TTE, see Appendix F.

2.4 Simulation Study

2.4.1 Data Generation

The data for the simulation is generated using an algorithm proposed by Young and Tchetgen Tchetgen (2014). The algorithm draws baseline values for (time-varying) confounders, treatment assignment, and survival outcomes, and then sequentially draws the survival sequence conditional on previous values. This process continues iteratively until an event occurs or the previously defined length of the period concludes. The algorithm uses parameters to indicate the outcome event rate, confounding strength, and treatment prevalence. For a comprehensive description of the algorithm and its theoretical underpinnings, we refer to Young and Tchetgen Tchetgen (2014).

To evaluate the performance of the different variance estimation methods, true values need to be known. While in simulation studies true parameter values normally are specified during the simulation design, it is not straightforward to simulate time-to-event data directly from specified true parameters (Young and Tchetgen Tchetgen 2014; Keogh et al. 2023). Therefore, Keogh et al. (2023) recommend generating the true reference values through a simulation-based approach. Two large-scale ($n = 1,000,000$) randomised control trials are simulated for each scenario by removing the effect of L_k on A_k by forcing the treatment strategy to either be 'always treated' or 'never treated'. The true MRD(k) is then estimated using the Kaplan-Meier estimator (Keogh et al. 2023).

2.4.2 Simulation Parameters

The properties of the sandwich-type estimator and the non-parametric bootstrap have previously been compared in a per-protocol analysis setting by Limozin et al. (2024). In the following simulation study, we attempt to replicate the conditions of Limozin et al. (2024) in an ITT analysis. The simulation parameters are chosen to allow for a low ($5-6\%$), medium ($10-14\%$), and high ($20-25\%$) percentage of individuals experiencing the outcome event during the follow-up time (Limozin et al. 2024). These scenarios are tested across different sample sizes, confounding strengths, and treatment prevalences. In total, 81 scenarios are tested with the parameters described in Table 1.

Sample size	Outcome Event Rate	Confounding strength	Treatment prevalence
200	-4.7	0.1	-1
1000	-3.8	0.5	0
5000	-3.0	0.9	1

Table 1 Conditions of the simulation study (Limozin et al. 2024). The outcome event rate and treatment prevalence are indicators that determine the resulting values through the algorithm described in Section 2.4.1. For a detailed description, see Young and Tchetgen Tchetgen (2014).

Multiple performance measures are used to evaluate and compare the properties of the sandwich-type estimators and the bootstrap confidence intervals. Specifically, the empirical coverage probability, bias-eliminated coverage probability, average confidence interval width, and statistical power are calculated for each simulation scenario (Morris et al. 2019). To assess the precision of the simulation results, the respective Monte Carlo Errors (MCE) are additionally estimated, providing a measure of the sampling variability of the performance metrics (Morris et al. 2019). To be able to understand the behaviour of the CIs, the $\widehat{\text{MRD}}$ point estimates calculated in `R` and `Julia` are compared, and their empirical bias and Mean Squared Errors (MSE) are computed. A detailed description of the performance measures and how to compute them and their MCEs can be found in Appendix C.

All simulations are conducted in `R` version `4.4.3` (R Core Team 2024) and `Julia` version `1.11.4` (Bezanson et al. 2017). The `TrialEmulation` package is used in version `0.0.4.2` (Su et al. 2024). All scripts and simulation results are available on GitHub[1].

3 Results

In the following section, we present the results of the simulation study comparing the performance of two bootstrap confidence interval estimation methods, percentile and empirical, with that of the commonly used sandwich-type estimator, applied in sequential Target Trial Emulation.

The coverage of the 95% confidence intervals is similar in all conditions for all three methods: empirical bootstrap CI, percentile bootstrap CI, and sandwich estimator CI. In some scenarios, mainly in low sample sizes ($n = 200$), the empirical bootstrap performs better than both the percentile bootstrap CI and the sandwich-type estimator CI, as can be seen in Figure D1. The overall coverage is close to 95% in most scenarios with a low sample size ($n = 200$; Figure D1). In medium sample sizes ($n = 1000$; Figure D2) the coverage starts to deteriorate in scenarios with a high outcome event rate. Finally, in large sample sizes ($n = 5000$; Figure D3), the coverage rate starts to deteriorate and is not close to 95% most of the time. Table 2 shows a direct comparison of which method shows the closest coverage to the nominal 95% over all simulations. In the direct comparison, both bootstrap methods reach the nominal 95% more often than the sandwich-type CIs.

	Sandwich	**Empirical**	**Percentile**
Sandwich	**24.7%**	40.5%	38.8%
Empirical	59.5%	**36.3%**	48.2%
Percentile	61.2%	51.8%	**39.0%**

Table 2 Comparison of **coverage** of the methods based on proximity to 0.95 coverage. Diagonal entries show the percentage of scenarios in which each method was closest to 0.95. Off-diagonal entries show pairwise win rates: the percentage of scenarios where the row method was closer to 0.95 than the column method.

[1] https://github.com/flo1met/thesis_TTE

The width of the empirical and the percentile bootstrap are exactly equal, as is to be expected, due to how they are computed (Equation 8-11). Therefore, when talking about the width, we will refer to them simply as bootstrap CIs. The bootstrap CIs are narrower in most scenarios with low sample sizes, except in scenarios with a high outcome event rate (outcome event rate indicator = −3; Figure D4). In those scenarios, the widths of the methods become approximately equal. This trend continues in scenarios with a low outcome event rate with a medium sample size (outcome event rate indicator = −3.8; Figure D5). In scenarios with a large sample size, the width of bootstrap and sandwich-type confidence intervals is almost the same (Figure D6).

The bias-eliminated coverage can be utilized to evaluate how the bias and width influence the coverage performance. In low sample sizes, the sandwich-type CI bias-eliminated coverage is higher than the percentile and the empirical bootstrap CIs in most cases, in which there is a medium (= 0) or high treatment prevalence (= 1; Figure D7). While the sandwich-type CIs continue to show a bias-eliminated coverage closer to 95% in some cases in medium sample sizes when there is a high treatment prevalence (Figure D8), overall the percentile bootstrap shows the closest coverage to 95% in most cases. In large sample sizes (Figure D9), the percentile and the empirical bootstrap consistently perform better than the sandwich-type CIs, with the percentile bootstrap CI often showing a slightly better coverage than the empirical bootstrap CI. Both bootstrap CIs show a closer coverage to 95% in more cases, when directly compared to the sandwich-type CIs, as seen in Table 3. Here, we must mention that the bias-eliminated bootstrap is not an indicator of the performance of the bootstrap at hand but rather an indicator of exploring why the bootstrap methods possibly show low coverage. This will be further discussed in Section 4.

	Sandwich	**Empirical**	**Percentile**
Sandwich	**19.0%**	30.1%	25.7%
Empirical	69.9%	**31.9%**	30.4%
Percentile	74.3%	69.6%	**49.1%**

Table 3 Comparison of **bias-eliminated coverage** of the methods based on proximity to 0.95 coverage. Diagonal entries show the percentage of scenarios in which each method was closest to 0.95. Off-diagonal entries show pairwise win rates: the percentage of scenarios where the row method was closer to 0.95 than the column method.

Next, we will look at the power to show the performance of the CI methods to find an effect when there is a true effect in the population. Over low (Figure D10) and medium sample sizes (Figure D11), all three CI methods show a consistent picture of the performance. The sandwich-type CIs consistently outperform the empirical bootstrap CI; in a few cases, the percentile bootstrap method reaches similar power. That being said, the overall power of all three methods is very low, with $< 50\%$ and in most cases even $< 25\%$. The overall power is higher in large sample size scenarios (Figure D12). While sandwich-type CIs still almost consistently outperform

the empirical bootstrap CIs, all three methods converge to the same power in some cases of a low treatment prevalence.

Next, we look at the $\widehat{\text{MRD}}$ point estimate to better understand how the bias possibly influences the estimated confidence intervals. When evaluating the results, over all 81 scenarios, with 5 follow-up timepoints each, and 1000 simulation replications, the expected number of results in `R` and `Julia` each is $81 \times 5 \times 1000 = 405,000$. While the `Julia` simulation does reach this value, in `R` only $390,955$ replications were possible. Reasons for this are discussed in Section 4. The resulting difference in point estimates is shown for low sample sizes in Figure E13, for medium sample sizes in Figure E14, and for large sample sizes in Figure E15. Low sample sizes show differences in $\widehat{\text{MRD}}$ point estimates in most cases, while differences only prevail in medium sample sizes for a few cases with high treatment prevalence and a low outcome event rate. There are no differences in the $\widehat{\text{MRD}}$ point estimates in large sample sizes.

To estimate bias and MSE, the point estimates from the `TargetTrialEmulation.jl` package have been used, as they are complete over all $405,000$ simulations. The bias of the $\widehat{\text{MRD}}$ point estimates is explored across sample sizes in Figure E16. The plot shows that the point estimate is slightly biased, especially in cases with a low outcome event rate. The bias is stronger in low sample sizes, while medium and large sample sizes are always very similar, with large sample sizes consistently being less biased. Further, bias occurs more strongly in later follow-up times ($k > 2$). As the bias increases with a reduction of the outcome event rate, this can most likely be explained by a low outcome event rate. The MSE shows the same pattern across all scenarios, visible in Figure E17.

Monte Carlo Errors (MCE) were estimated and examined but were consistently low, ranging from 0.2% to, in rare cases, 0.7%, for all estimated measures. A total of $1,000$ simulation repetitions is sufficient to detect a nominal coverage of 95% with a maximal Monte Carlo Error of 1%, as only $\frac{0.95 \times (1-0.95)}{0.01^2} = 475$ repetitions are theoretically required (Morris et al. 2019). Consequently, we do not discuss MCEs further here. Full details are available in the online appendix on GitHub[2].

4 Discussion

In this simulation study, we investigated the difference in performance of the sandwich-type estimator compared to percentile and empirical bootstrap methods in the context of sequential Target Trial Emulation. We found that bootstrapped CIs, both the percentile and the empirical method, more often achieve coverage closer to the nominal 95% level than the sandwich-type confidence intervals. Bootstrap CIs additionally appear to be narrower, especially in cases with a low sample size or high outcome event rates. This suggests bootstrapping may provide more reliable inference in finite samples, especially under complex data structures. Although the overall power was low in this simulation study, the sandwich-type CI intervals found the true effect consistently more often than both bootstrap CI methods. While this may seem problematic, this is explainable by the small effect sizes generated by the data-generating mechanism, which can be seen in Figures E13-E15. The sandwich-type estimator has been

[2] https://github.com/flo1met/thesis_TTE

found to be biased upwards in survival analysis (Austin 2016), which could explain its higher power in low effect size scenarios. Importantly, this study was not explicitly designed to evaluate power, which constitutes a study design limitation. Furthermore, assessing the type I error rate was impossible, as the data-generating mechanism outlined in Section 2.4.1 always introduced a non-zero, but small, effect. Future research should therefore aim to design a simulation study specifically targeting the evaluation of power and type I error rates.

The results show that there are differences between the $\widehat{\text{MRD}}$ point estimates in `TrialEmulation` and `TargetTrialEmulation.jl`. These differences happen in scenarios that show a combination of low sample sizes, a medium to high treatment prevalence, and a medium to low outcome event rate. Therefore, those problems are most likely traced back to convergence issues. Unfortunately, due to `R` suppressing warnings, which was not accounted for before running the simulations, the exact reasons can not be retraced at this moment, but are subject to future investigation.

We found that the outcome model is biased, mainly in scenarios with a low outcome event rate and in higher follow-up times. While small biases in early follow-up visits ($k \leq 2$) are negligible and can be explained by a finite sample bias, as previously found in the literature (Keogh et al. 2021), larger biases are most likely explained by only few outcome events occurring at later follow-up times. When removing the bias, by analyzing the bias-eliminated coverage rate, we showed that the bootstrap CI methods consistently performed better, except in cases with a high treatment prevalence. Given the similar CI widths across methods, except in high event rate scenarios, the observed undercoverage can be primarily attributed to bias in the point estimates (Morris et al. 2019). This result points to the bootstrap methods potentially yielding better results than sandwich-type estimators in sequential TTE. The findings additionally suggest that the empirical sampling distribution generated through the bootstrap procedure described in Section 2.2 is not symmetrically distributed. To illustrate this, Appendix G presents three example bootstrap distributions showing skewness. These examples confirm our suspicion of skewed bootstrap distributions. This explains the percentile and empirical bootstraps' performance difference and possible underperformance. The percentile bootstrap method can naturally handle skewed distributions but is sensitive to biases in the point estimate. The empirical bootstrap assumes symmetry and is, therefore, sensitive to skewness. This points out another limitation of this study, which should be addressed in future research by including more bootstrap methods. For future research, we recommend the implementation of the bias-corrected and accelerated bootstrap, as this method allows for asymmetric bootstrap distributions (Morris et al. 2019; Lei and Smith 2003).

The findings show evidence that they are in accordance with previous findings in the survival analysis literature. Bootstrap CIs have been shown to provide confidence intervals with a correct coverage rate, while other methods like the sandwich-type estimators yielded incorrect coverage rates (Austin 2016). For a conclusive answer, further research is necessary.

As mentioned in Section 2.4.2, we replicated parts of the simulation study by Limozin et al. (2024) in the context of an ITT analysis. While the relative performance of the sandwich-type and empirical bootstrap confidence interval methods is broadly

consistent with the findings of Limozin et al. (2024), notable discrepancies remain, especially concerning coverage and the bias-eliminated coverage probability. In the present study, both coverage and bias-eliminated coverage often approached the nominal 95% level, whereas Limozin et al. (2024) find that bootstrap confidence intervals remained substantially below nominal bias-eliminated coverage in most scenarios.

Further, Limozin et al. (2024) proposed the linearised equation function (LEF) bootstrap as an alternative to the non-parametric empirical bootstrap, to reduce computation time and prevent convergence issues when using a pooled logistic regression to estimate survival times. While we do find computation time as a practical limitation (Appendix F), we consider computational improvements through using more efficient programming languages like `Julia`, using parallelization of bootstrap samples, or implementing stochastic gradient descent (Christmann and Lei 2024) as more promising strategies. In trial runs, convergence issues did not seem to be a big problem, with only very few failed iterations with $B = 500$ bootstrap samples. During the estimation in Appendix G, none of the iterations failed. Therefore, concentrating on improving inference should be the focus of future research. As Limozin et al. (2024) did not investigate power or width, a comparison of those parameters is not possible.

In summary, while the bootstrap methods come with increased computational cost, and we do find lower power in the present scenarios, they show promise as a more suitable approach for estimating causal effects in the context of sequential TTE, owing to their more accurate coverage and consistently narrower interval widths. Nonetheless, further research is needed before bootstrap confidence intervals can reliably be employed in sequential Target Trial Emulation. Although this study shows limitations of the percentile and empirical bootstrap confidence intervals, we find evidence that bootstrapping methods that adjust for biased estimators and non-symmetric bootstrap sample distributions, like the bias-corrected and accelerated bootstrap, could outperform the commonly implemented sandwich-type confidence intervals consistently. In addition, power and type I error rates should be explicitly investigated. Future research should focus on improving inference and conducting systematic benchmarking to determine whether more advanced bootstrap techniques offer more reliable and efficient inference in sequential TTE analyses.

References

van Amsterdam W.: The Need for Speed, Performing Simulation Studies in R, JAX and Julia; 2024. https://vanamsterdam.github.io/posts/240308-jaxopt-vs-r-vs-julia/.

Austin PC. Variance Estimation When Using Inverse Probability of Treatment Weighting (IPTW) with Survival Analysis. Statistics in Medicine. 2016;35(30):5642–5655. https://doi.org/10.1002/sim.7084.

Austin PC, Stuart EA. Moving towards Best Practice When Using Inverse Probability of Treatment Weighting (IPTW) Using the Propensity Score to Estimate Causal Treatment Effects in Observational Studies. Statistics in Medicine. 2015;34(28):3661–3679. https://doi.org/10.1002/sim.6607.

Bakker LJ, Goossens LMA, O'Kane MJ, Uyl-de Groot CA, Redekop WK. Analysing Electronic Health Records: The Benefits of Target Trial Emulation. Health Policy and Technology. 2021 Sep;10(3):100545. https://doi.org/10.1016/j.hlpt.2021.100545.

Bezanson J, Edelman A, Karpinski S, Shah VB. Julia: A Fresh Approach to Numerical Computing. SIAM Review. 2017;59(1):65–98. 45109190.

Binder D, Kovacevic M, Roberts G. Design-based methods for survey data: Alternative uses of estimating functions. In: Proceedings of the Section on Survey Research Methods American Statistical Association Alexandria, VA; 2004. p. 3301–3312.

Carpenter J, Bithell J. Bootstrap Confidence Intervals: When, Which, What? A Practical Guide for Medical Statisticians. Statistics in Medicine. 2000 May;19(9):1141–1164. https://doi.org/10.1002/(SICI)1097-0258(20000515)19:9⟨1141::AID-SIM479⟩3.0.CO;2-F.

Christmann A, Lei Y.: Bootstrap SGD: Algorithmic Stability and Robustness. arXiv; 2024.

Fu EL. Target Trial Emulation to Improve Causal Inference from Observational Data: What, Why, and How? Journal of the American Society of Nephrology. 2023 Aug;34(8):1305. https://doi.org/10.1681/ASN.0000000000000152.

Fu EL, Harhay M, Schneeweiss S, Desai R, Hernán MA. Starting Right: Aligning Eligibility and Treatment Assignment at Time Zero When Emulating a Target Trial. Available at SSRN 5177135. 2025;.

Hernán MA. The Hazards of Hazard Ratios. Epidemiology. 2010 Jan;21(1):13. https://doi.org/10.1097/EDE.0b013e3181c1ea43.

Hernán MA, Alonso A, Logan R, Grodstein F, Michels KB, Willett WC, et al. Observational Studies Analyzed Like Randomized Experiments: An Application to

Postmenopausal Hormone Therapy and Coronary Heart Disease. Epidemiology. 2008 Nov;19(6):766–779. https://doi.org/10.1097/EDE.0b013e3181875e61.

Hernán MA, Dahabreh IJ, Dickerman BA, Swanson SA. The Target Trial Framework for Causal Inference From Observational Data: Why and When Is It Helpful? Annals of Internal Medicine. 2025 Mar;178(3):402–407. https://doi.org/10.7326/ANNALS-24-01871.

Hernán MA, Robins JM. Using Big Data to Emulate a Target Trial When a Randomized Trial Is Not Available: Table 1. American Journal of Epidemiology. 2016 Apr;183(8):758–764. https://doi.org/10.1093/aje/kwv254.

Hernan MA, Robins JM. Causal Inference: What If; 2020.

Hernán MA, Sterne JAC, Higgins JPT, Shrier I, Hernández-Díaz S. A Structural Description of Biases That Generate Immortal Time. Epidemiology. 2025 Jan;36(1):107–114. https://doi.org/10.1097/EDE.0000000000001808.

Igelström E, Craig P, Lewsey J, Lynch J, Pearce A, Katikireddi SV. Causal Inference and Effect Estimation Using Observational Data. Journal of Epidemiology and Community Health. 2022 Nov;76(11):960–966. https://doi.org/10.1136/jech-2022-219267.

Karpinski S, Shah V, Edelman A, Bezanson J.: Why We Created Julia; 2012. https://julialang.org/blog/2012/02/why-we-created-julia/.

Keogh RH, Gran JM, Seaman SR, Davies G, Vansteelandt S. Causal Inference in Survival Analysis Using Longitudinal Observational Data: Sequential Trials and Marginal Structural Models. Statistics in Medicine. 2023;42(13):2191–2225. https://doi.org/10.1002/sim.9718.

Keogh RH, Seaman SR, Gran JM, Vansteelandt S. Simulating Longitudinal Data from Marginal Structural Models Using the Additive Hazard Model. Biometrical Journal Biometrische Zeitschrift. 2021 Oct;63(7):1526–1541. https://doi.org/10.1002/bimj.202000040.

Lei S, Smith MR. Evaluation of Several Nonparametric Bootstrap Methods to Estimate Confidence Intervals for Software Metrics. IEEE Transactions on Software Engineering. 2003 Nov;29(11):996–1004. https://doi.org/10.1109/TSE.2003.1245301.

Li L, Zhu N, Zhang L, Kuja-Halkola R, D'Onofrio BM, Brikell I, et al. ADHD Pharmacotherapy and Mortality in Individuals With ADHD. JAMA. 2024 Mar;331(10):850–860. https://doi.org/10.1001/jama.2024.0851.

Li T, Lawson J. A Generalized Bootstrap Procedure of the Standard Error and Confidence Interval Estimation for Inverse Probability of Treatment Weighting.

Multivariate Behavioral Research. 2024 Mar;59(2):251–265. https://doi.org/10.1080/00273171.2023.2254541.

Limozin JM, Seaman SR, Su L.: Inference Procedures in Sequential Trial Emulation with Survival Outcomes: Comparing Confidence Intervals Based on the Sandwich Variance Estimator, Bootstrap and Jackknife. arXiv; 2024.

Lin DY, Wei LJ. The Robust Inference for the Cox Proportional Hazards Model. Journal of the American Statistical Association. 1989 Dec;84(408):1074–1078. https://doi.org/10.1080/01621459.1989.10478874.

Listl S, Jürges H, Watt RG. Causal Inference from Observational Data. Community Dentistry and Oral Epidemiology. 2016;44(5):409–415. https://doi.org/10.1111/cdoe.12231.

Mandel M. Simulation-Based Confidence Intervals for Functions With Complicated Derivatives. The American Statistician. 2013 May;67(2):76–81. https://doi.org/10.1080/00031305.2013.783880.

Maringe C, Benitez Majano S, Exarchakou A, Smith M, Rachet B, Belot A, et al. Reflection on Modern Methods: Trial Emulation in the Presence of Immortal-Time Bias. Assessing the Benefit of Major Surgery for Elderly Lung Cancer Patients Using Observational Data. International Journal of Epidemiology. 2020 Oct;49(5):1719–1729. https://doi.org/10.1093/ije/dyaa057.

Metwaly F.: TargetTrialEmulation.jl - pre-release; 2025.

Morris TP, White IR, Crowther MJ. Using Simulation Studies to Evaluate Statistical Methods. Statistics in Medicine. 2019;38(11):2074–2102. https://doi.org/10.1002/sim.8086.

Nichols A. Causal Inference with Observational Data. The Stata Journal. 2007;7(4).

R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria; 2024, https://www.R-project.org/.

Sanson-Fisher RW, Bonevski B, Green LW, D'Este C. Limitations of the Randomized Controlled Trial in Evaluating Population-Based Health Interventions. American Journal of Preventive Medicine. 2007 Aug;33(2):155–161. https://doi.org/10.1016/j.amepre.2007.04.007.

Schafer JL, Kang J. Average Causal Effects from Nonrandomized Studies: A Practical Guide and Simulated Example. Psychological Methods. 2008;13(4):279–313. https://doi.org/10.1037/a0014268.

Scola G, Chis Ster A, Bean D, Pareek N, Emsley R, Landau S. Implementation of the Trial Emulation Approach in Medical Research: A Scoping Review. BMC Medical Research Methodology. 2023 Aug;23(1):186. https://doi.org/10.1186/s12874-023-02000-9.

Su L, Rezvani R, Seaman SR, Starr C, Gravestock I.: TrialEmulation: An R Package to Emulate Target Trials for Causal Analysis of Observational Time-to-event Data. arXiv; 2024.

Tripepi G, Chesnaye NC, Dekker FW, Zoccali C, Jager KJ. Intention to Treat and per Protocol Analysis in Clinical Trials. Nephrology. 2020;25(7):513–517. https://doi.org/10.1111/nep.13709.

Wold H. Causal Inference from Observational Data: A Review of End and Means. Journal of the Royal Statistical Society Series A (General). 1956;119(1):28–61. https://doi.org/10.2307/2342961. 2342961.

Wu YH, Lin SY, Lin FJ, Tang SC, Wang CC. Outcomes of Reinitiating Direct Oral Anticoagulants After Intracranial Hemorrhage: A Sequential Target Trial Emulation Study. JACC: Asia. 2025 Mar;5(3, Part 1):361–370. https://doi.org/10.1016/j.jacasi.2024.11.008.

Yadav K, Lewis RJ. Immortal Time Bias in Observational Studies. JAMA. 2021 Feb;325(7):686–687. https://doi.org/10.1001/jama.2020.9151.

Young JG, Tchetgen Tchetgen EJ. Simulation from a Known Cox MSM Using Standard Parametric Models for the G-Formula. Statistics in Medicine. 2014 Mar;33(6):1001–1014. https://doi.org/10.1002/sim.5994.

Zivich PN, Cole SR, Shook-Sa BE, DeMonte JB, Edwards JK.: Estimating Equations for Survival Analysis with Pooled Logistic Regression. arXiv; 2025.

Data Availability Statement

All codes and results are publicly available on https://github.com/flo1met/thesis_TTE. The developed `TargetTrialEmulation.jl` package is available on https://github.com/flo1met/TargetTrialEmulation.jl. Package documentation is available on https://flo1met.github.io/TargetTrialEmulation.jl/stable/.

Ethical Approval

Ethical approval was granted by the Utrecht University FETC under the case Nr.: 24-2059

AI Statement

During the project and while writing this paper, generative AI models (ChatGPT, Grammarly) were solely used to improve the text's stylistic and grammatical structure and assist in coding.

Appendix A Pseudo Code for sequential Target Trial Emulation

Algorithm 2 sequential Target Trial Emulation (TTE)

Require: Longitudinal dataset `df`, including identifier variable `id`, treatment variable A_t, eligibility indicator E_t, time variable `T`, time-varying covariates L_t

for each time-point t in `df` **do**
  Identify individuals eligible at time-point t ($E_t = 1$)
  Subset data for eligible individuals from time-point t onward
  if no eligible observations exist **then**
    continue to next time-point
  end if
  Assign trial number t to all observations in the subset
  Compute variable `follow-up time` K for each individual starting from time-point t
  Sort subset by `id` and `T`
  Fix treatment A_t and covariates L_t to baseline ($k = 0$) for each individual
  Add trial t to the output list
end for
Combine all trials into a single emulated dataset
return Combined dataset

Appendix B Causal Assumptions of Marginal Risk Difference (MRD)

To identify the marginal risk difference (MRD) defined in Section 2 from observational data in a sequential TTE setting, the following causal assumptions must hold:

1. **Consistency**
   For each individual, the observed outcome under the actual treatment received equals the potential outcome under that treatment:

   $$Y = Y^a \quad \text{if } A = a.$$

   This assumes that treatment is well-defined and versions of treatment are consistent.
2. **Positivity**
   Every individual has a non-zero probability of receiving each treatment level, given their covariates:

   $$\Pr(A = a \mid V, L_0) > 0,$$

   where V time-invariant and L_0 are baseline covariates.
3. **Conditional Exchangeability (No Unmeasured Confounding)**
   Treatment assignment is independent of the potential outcomes, conditional on baseline covariates:

   $$Y^a \perp\!\!\!\perp A \mid V, L_0.$$

   This implies all confounding is captured by the measured covariates V and L_0.
4. **Correct Specification of Models**
   The models used for estimating treatment effects must be correctly specified.

These assumptions are standard in causal inference using the potential outcomes framework and must be justified or approximated for the MRD to be interpreted causally. For further references, see (Hernan and Robins 2020).

Appendix C Performance Measures Formulas

In the following Table C1 we provide the formulas for all performance measures and their Monte Carlo Errors (MCE), which were used in the simulation study, described in Section 2.4.2.

Performance Measure	**Definition**	**Estimate**	**MCE**
Coverage $\hat{C}$	$\Pr(\hat{\theta}_{low} \leq \theta \leq \hat{\theta}_{high})$	$\frac{\sum_{i=1}^{n_{\text{sim}}} \mathbb{I}\left(\hat{\theta}_{\text{low},i} \leq \theta \leq \hat{\theta}_{\text{high},i}\right)}{n_{\text{sim}}}$	$\sqrt{\frac{\hat{C} \times (1-\hat{C})}{n_{\text{sim}}}}$
Bias-Eliminated Coverage $\hat{C}_{BE}$	$\Pr(\hat{\theta}_{low} \leq \bar{\theta} \leq \hat{\theta}_{high})$	$\frac{\sum_{i=1}^{n_{\text{sim}}} \mathbb{I}\left(\hat{\theta}_{\text{low},i} \leq \bar{\theta} \leq \hat{\theta}_{\text{high},i}\right)}{n_{\text{sim}}}$	$\sqrt{\frac{\hat{C}_{BE} \times (1-\hat{C}_{BE})}{n_{\text{sim}}}}$
Power	$\Pr(p_i) \leq \alpha$	$\frac{\sum_{i=1}^{n_{\text{sim}}} \mathbb{I}(p_i \leq \alpha)}{n_{\text{sim}}}$	$\sqrt{\frac{\hat{P} \times (1-\hat{P})}{n_{\text{sim}}}}$
Bias	$E[\hat{\theta}] - \theta$	$\frac{\sum_{i=1}^{n_{sim}} \hat{\theta}_i - \theta}{n_{sim}}$	$\sqrt{\frac{\sum_{i=1}^{n_{sim}} (\hat{\theta}_i - \theta)^2}{n_{sim} \times (n_{sim}-1)}}$
Mean Squared Error $\widehat{\text{MSE}}$	$E[(\hat{\theta} - \theta)^2]$	$\frac{\sum_{i=1}^{n_{\text{sim}}} (\hat{\theta}_i - \theta)^2}{n_{\text{sim}}}$	$\sqrt{\frac{\sum_{i=1}^{n_{\text{sim}}} \left[(\hat{\theta}_i - \theta)^2 - \widehat{\text{MSE}}\right]^2}{n_{\text{sim}}(n_{\text{sim}}-1)}}$

Table C1 The table provides the formal definition and formulas for the performance measures used in the simulation study described in Section 2.4, as well as the formulas to their respective Monte Carlo Errors (MCE). $\hat{\theta}$ represents the individual estimate of the parameter, and $\bar{\theta}$ denotes the average of the estimates across all simulations. Formulas are taken from Table 6 of Morris et al. (2019).

Appendix D Results Simulation: Confidence Intervals

In Appendix D we present all plots regarding the performance of the three confidence interval estimation methods, sandwich-type, percentile, and empirical bootstrap estimators. The specific numbers to each plot can be found in the online supplementary materials on https://github.com/flo1met/thesis_TTE.

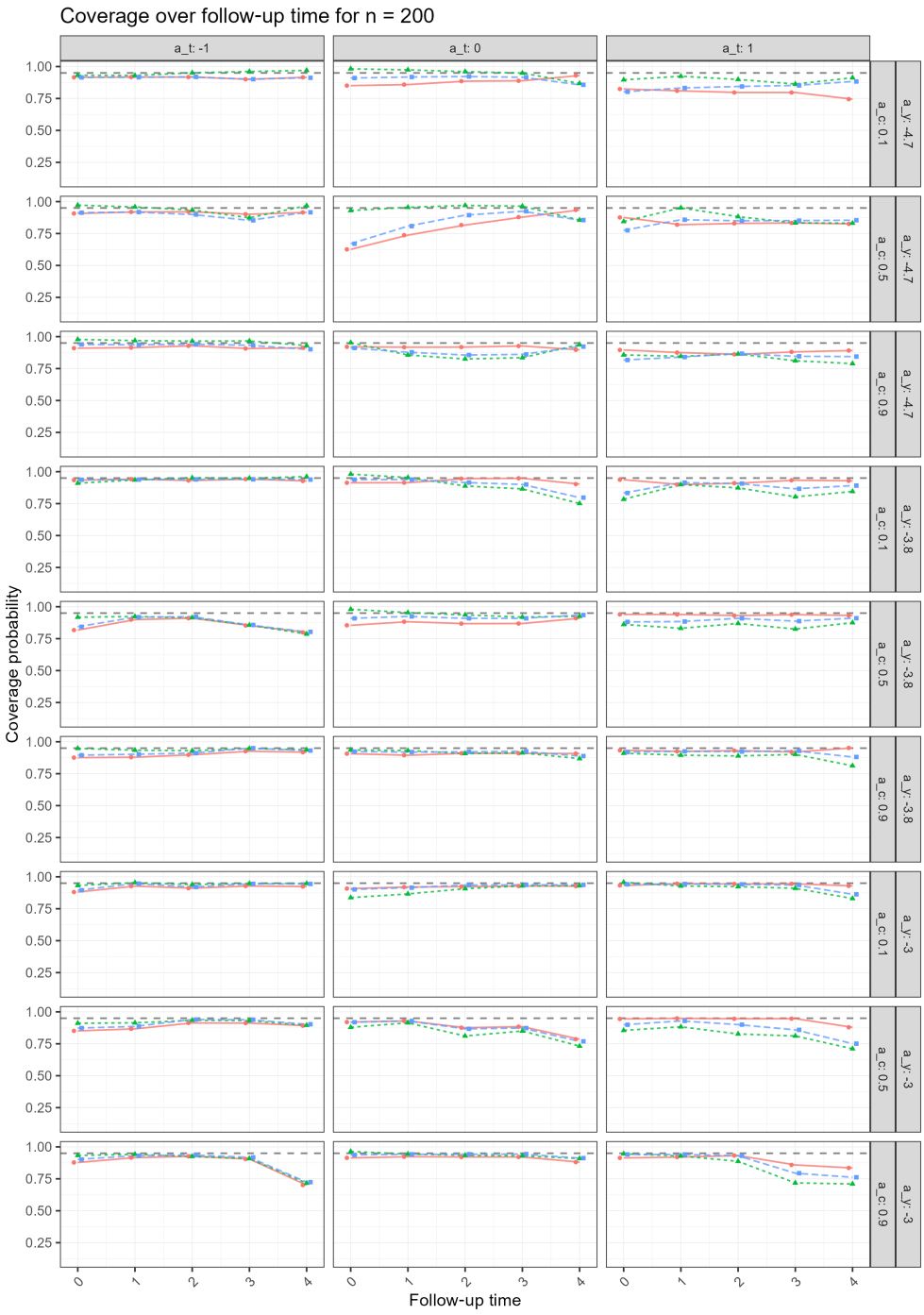

Fig. D1 Coverage of the 95% confidence interval over 1000 simulations with sample size $n = 200$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The dashed gray line marks 95%. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

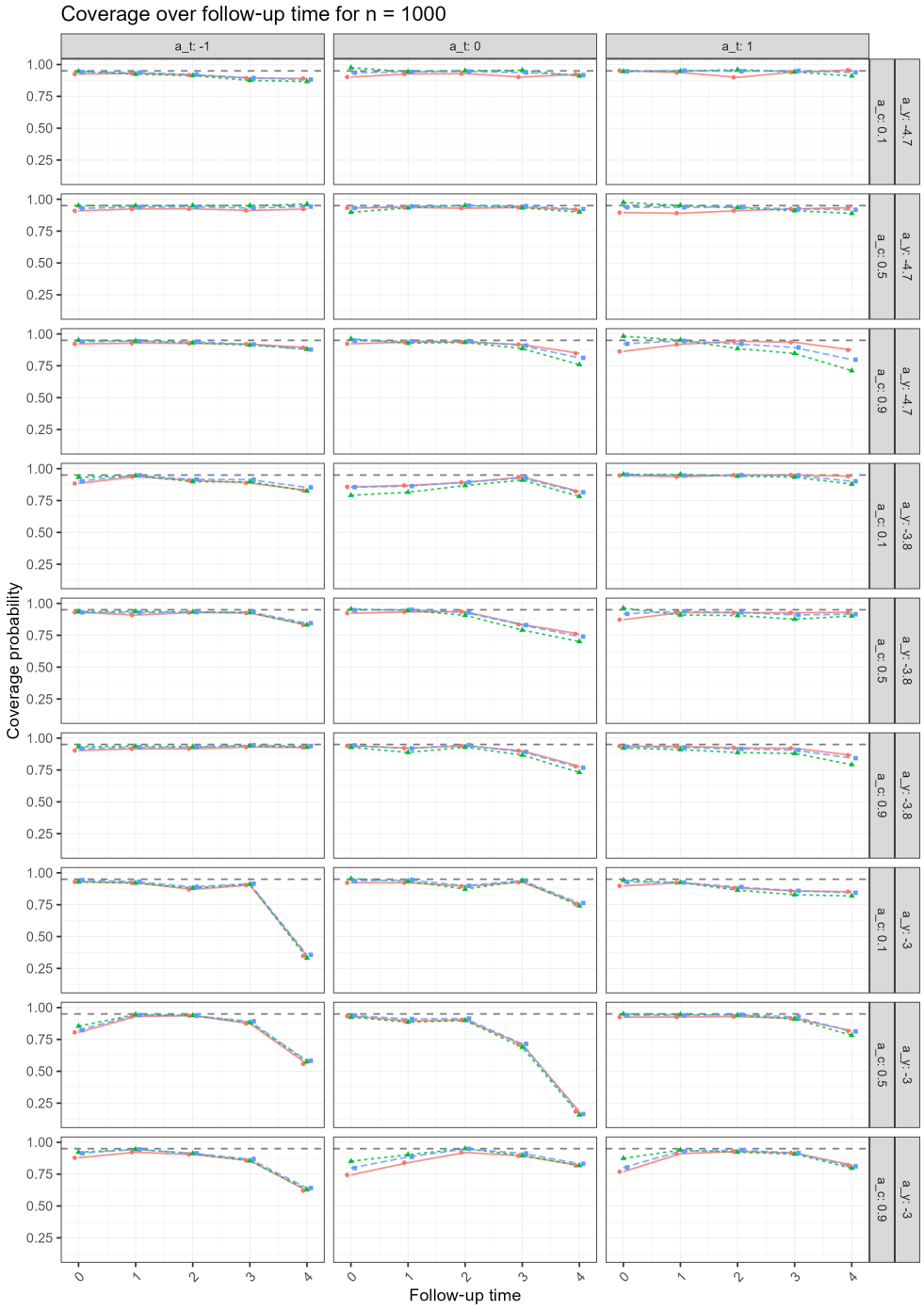

Fig. D2 Coverage of the 95% confidence interval over 1000 simulations with sample size $n = 1000$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The dashed gray line marks 95%. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

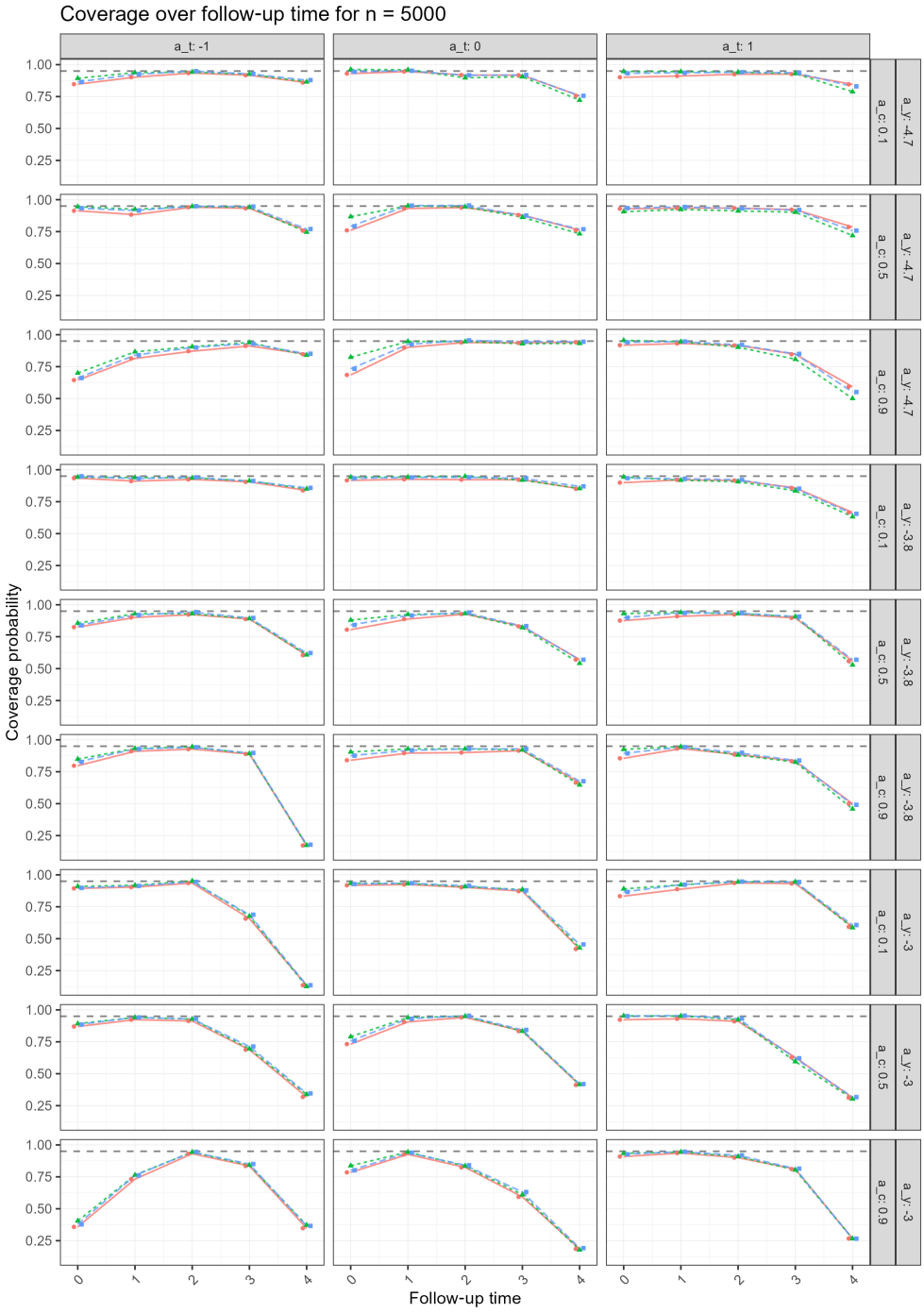

Fig. D3 Coverage of the 95% confidence interval over 1000 simulations with sample size $n = 5000$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The dashed gray line marks 95%. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

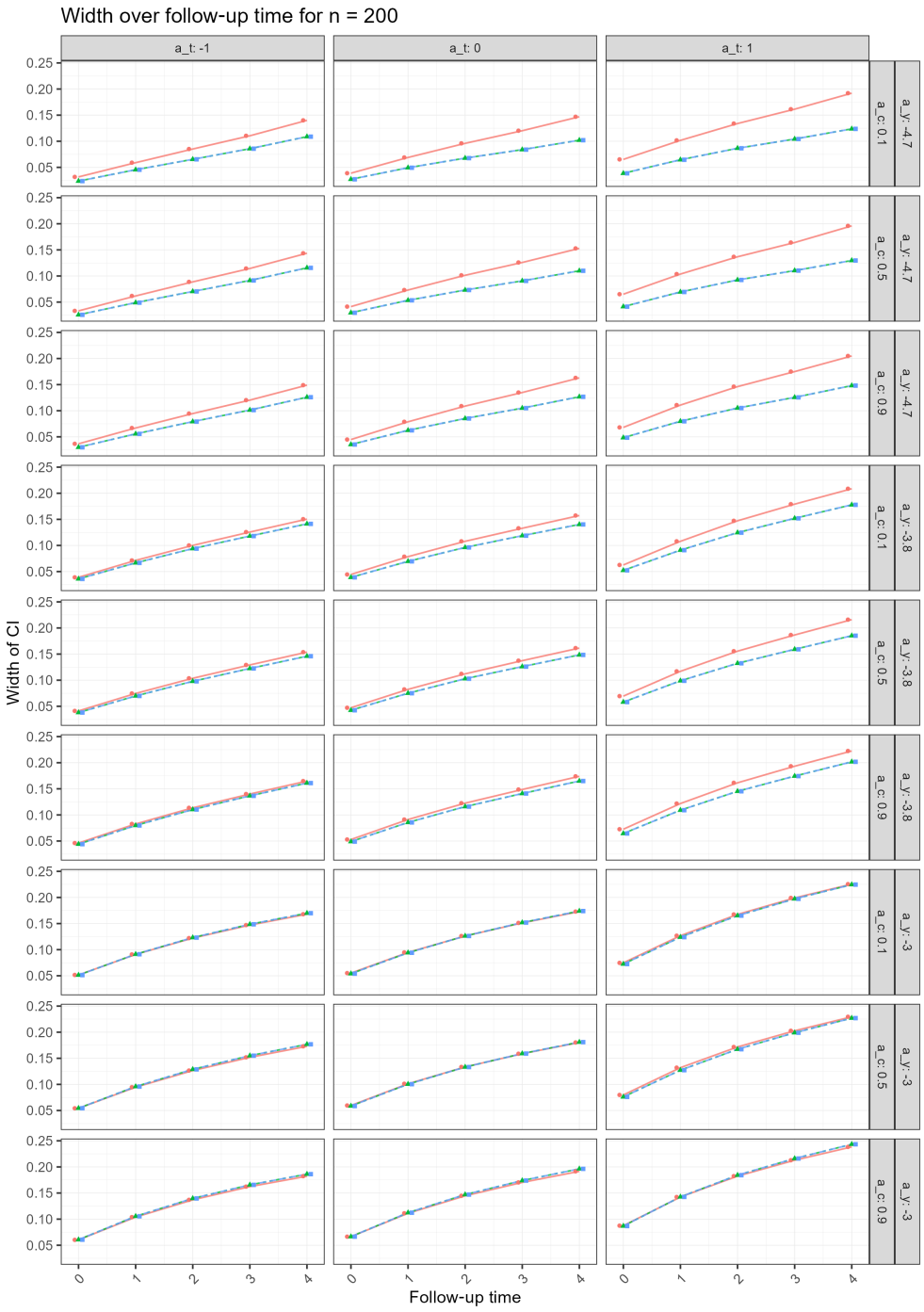

Fig. D4 Width of the 95% confidence interval over 1000 simulations with sample size $n = 200$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

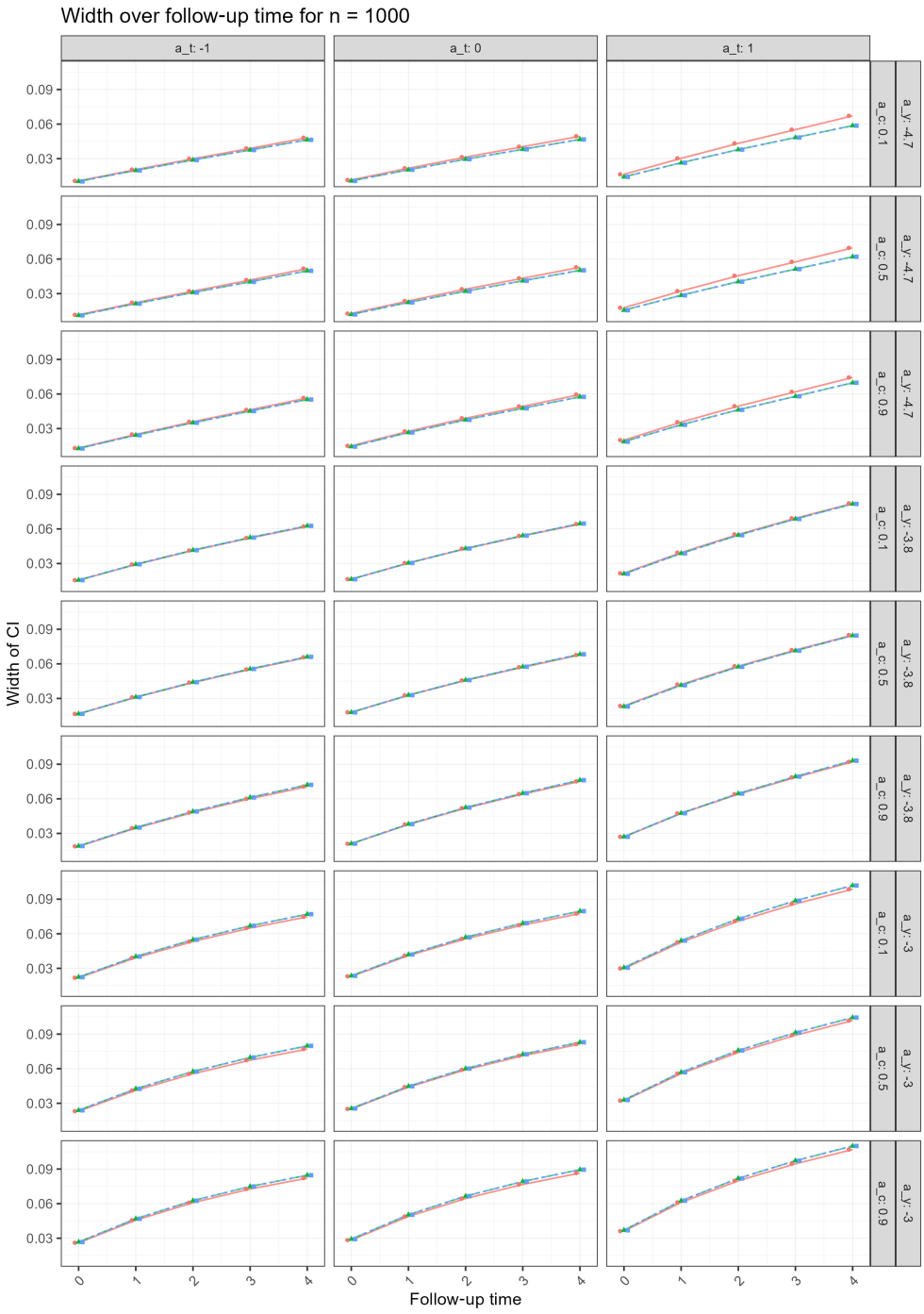

Fig. D5 Width of the 95% confidence interval over 1000 simulations with sample size $n = 1000$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

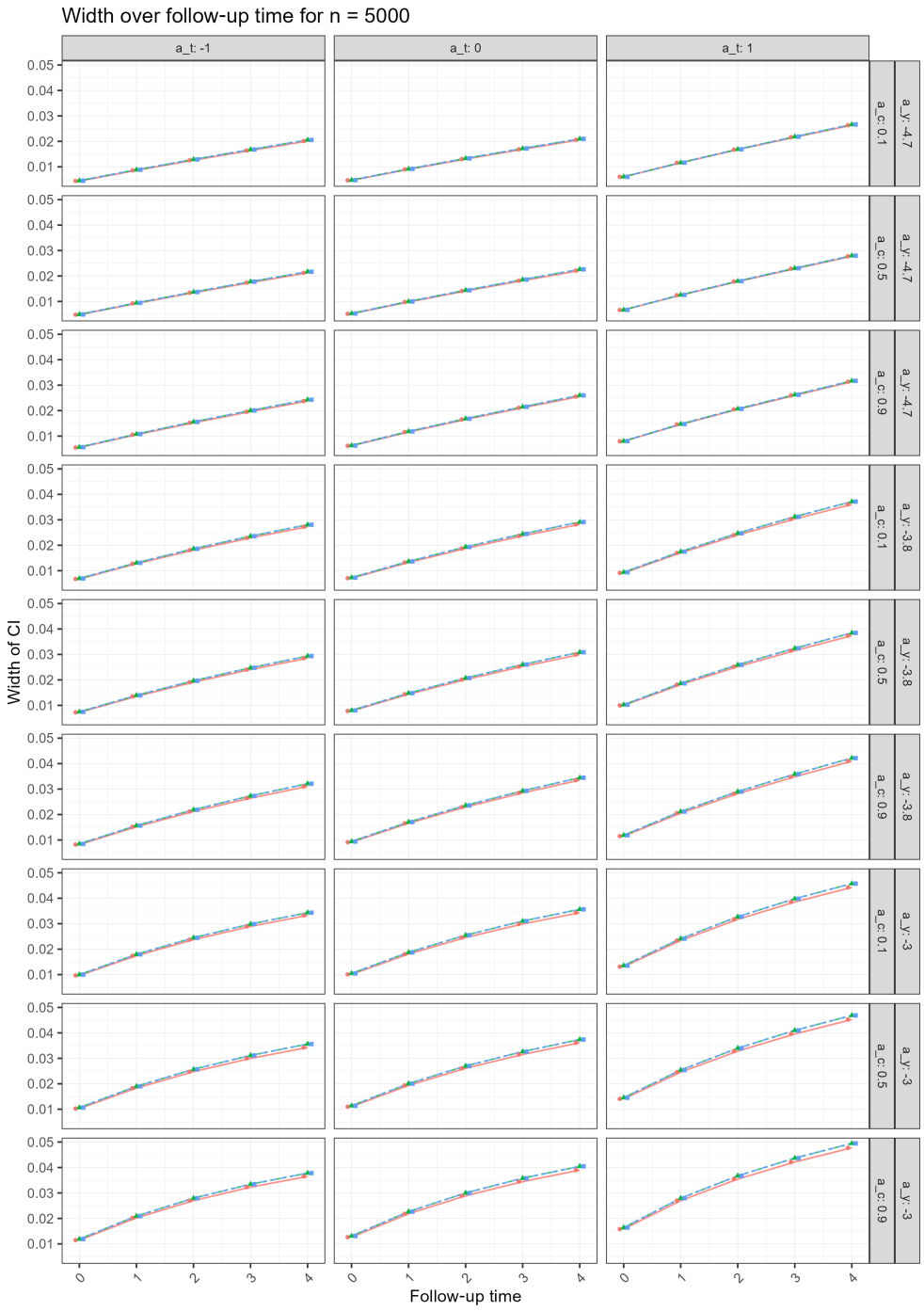

Fig. D6 Width of the 95% confidence interval over 1000 simulations with sample size $n = 5000$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

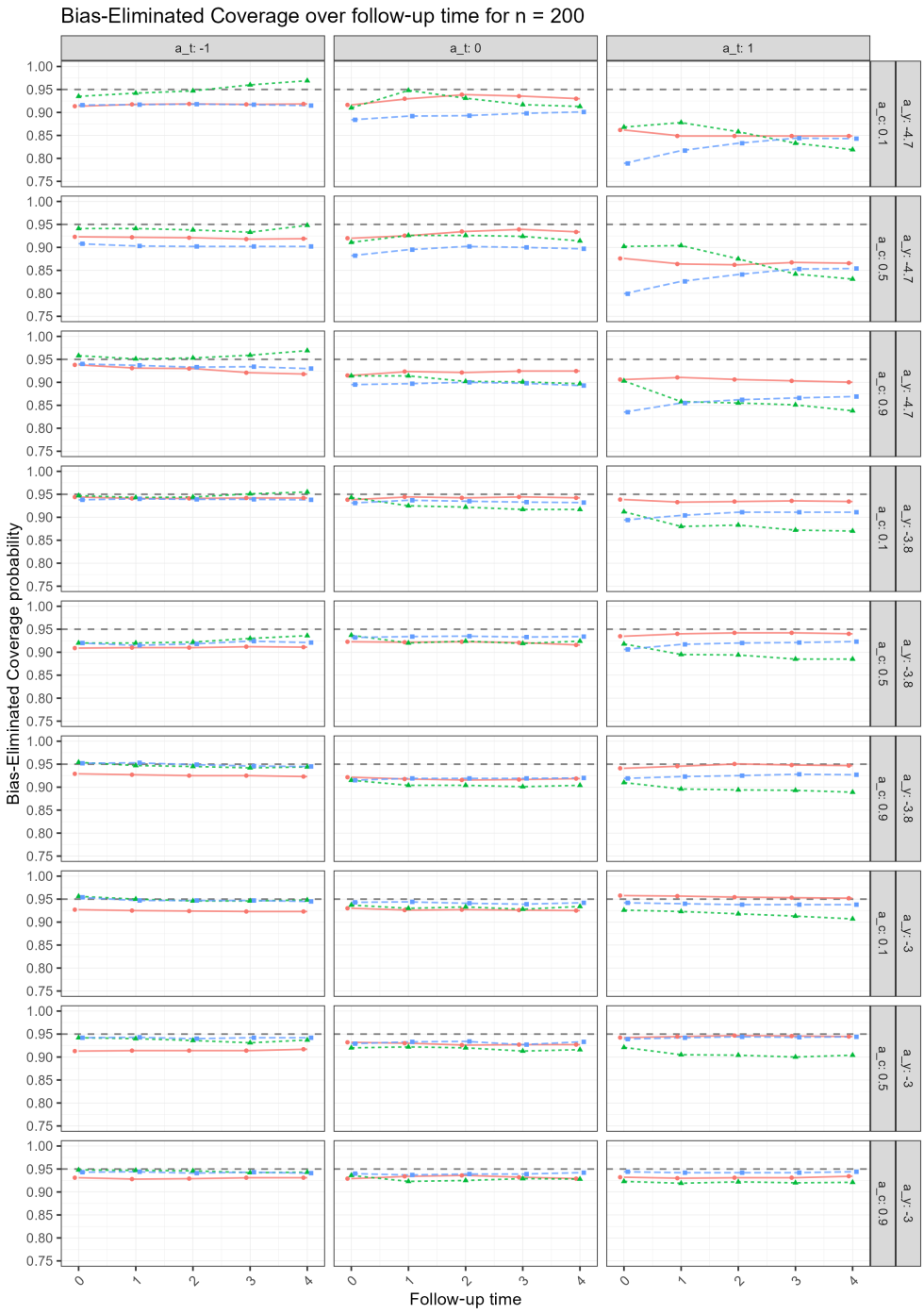

Fig. D7 Bias-eliminated coverage of the 95% confidence interval over 1000 simulations with sample size $n = 200$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The dashed gray line marks 95%. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

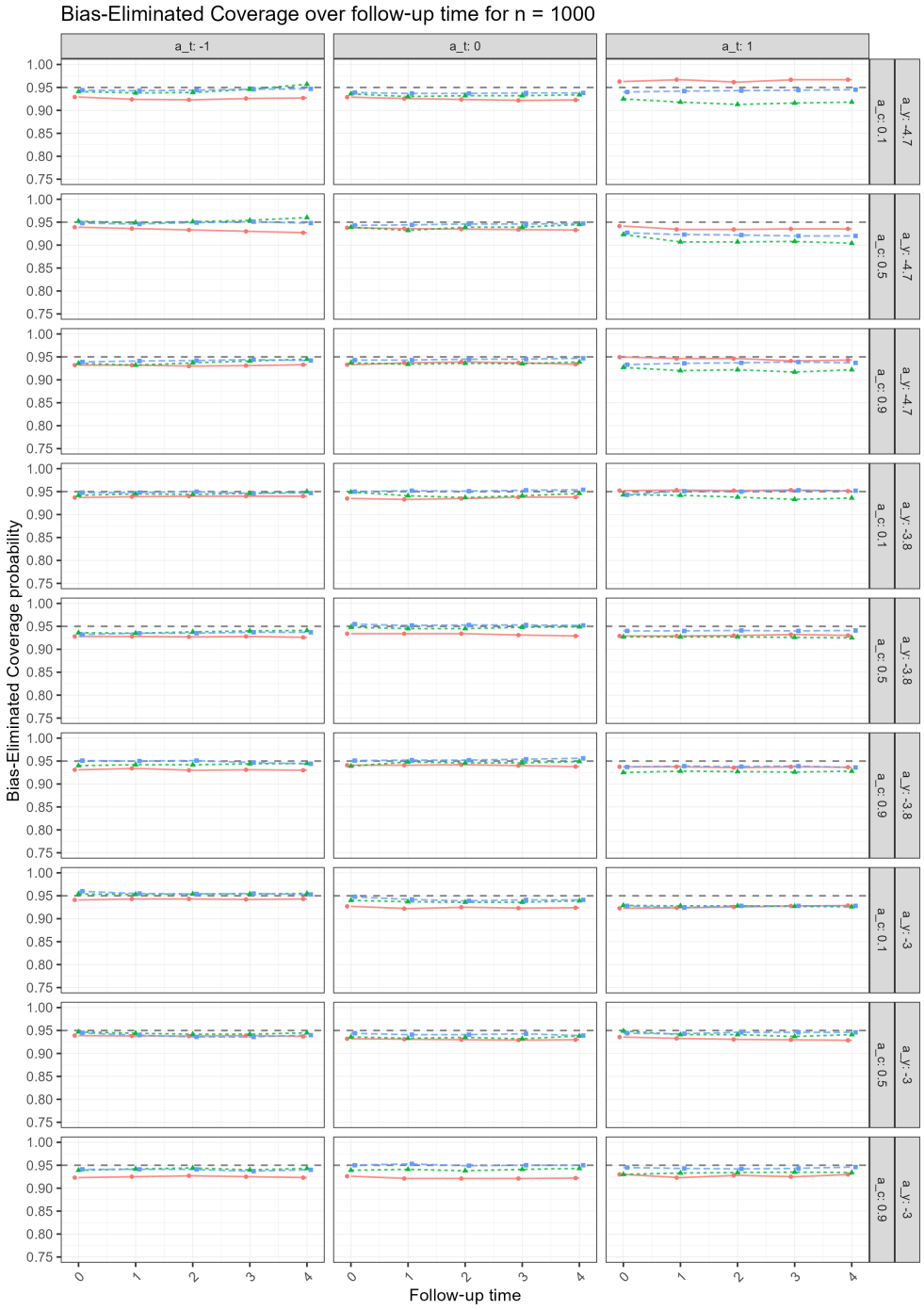

Fig. D8 Bias-eliminated coverage of the 95% confidence interval over 1000 simulations with sample size $n = 1000$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The dashed gray line marks 95%. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

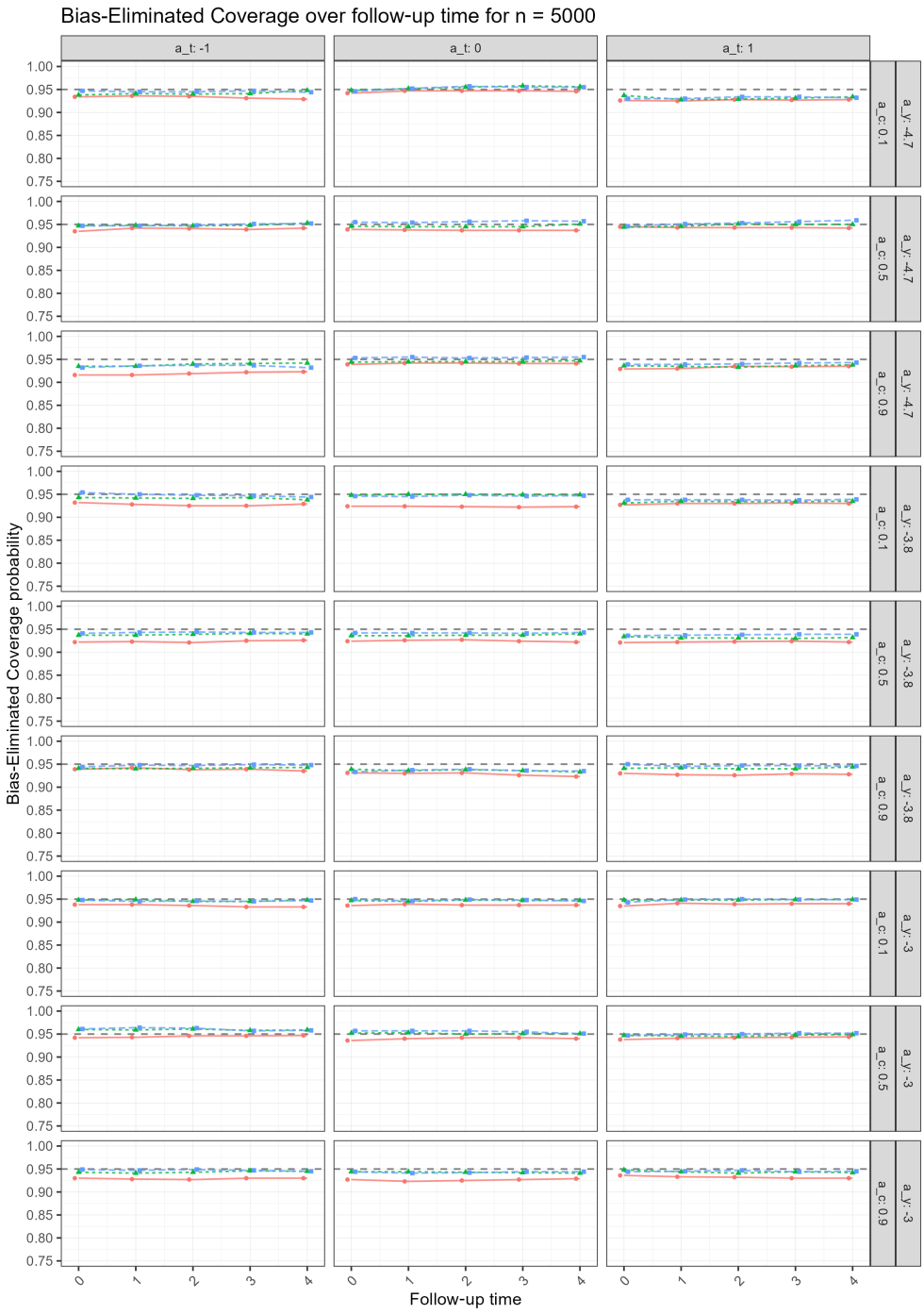

Fig. D9 Bias-eliminated coverage of the 95% confidence interval over 1000 simulations with sample size $n = 5000$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The dashed gray line marks 95%. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

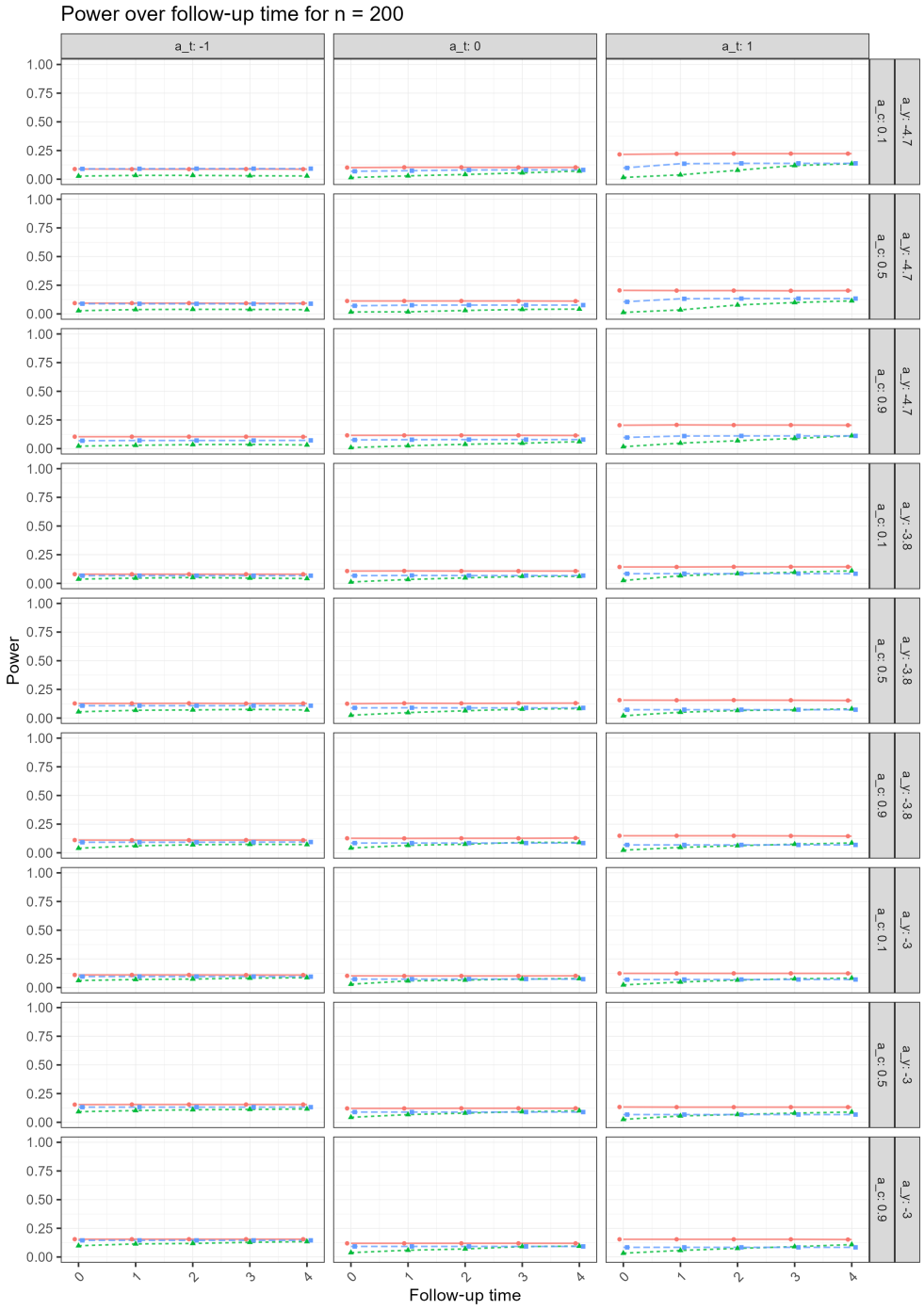

Fig. D10 Power of the 95% confidence interval over 1000 simulations with sample size $n = 200$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

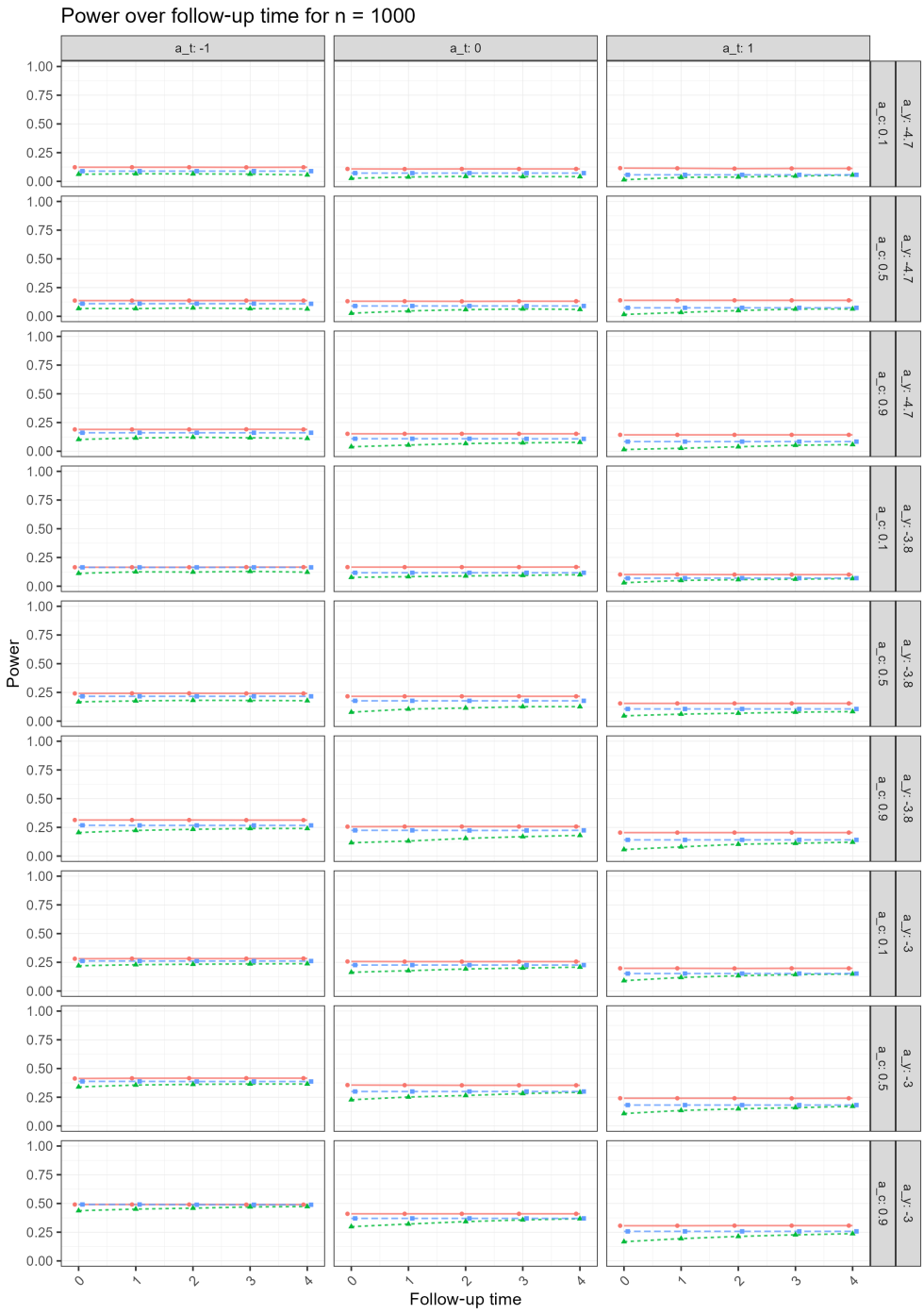

Fig. D11 Power of the 95% confidence interval over 1000 simulations with sample size $n = 1000$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

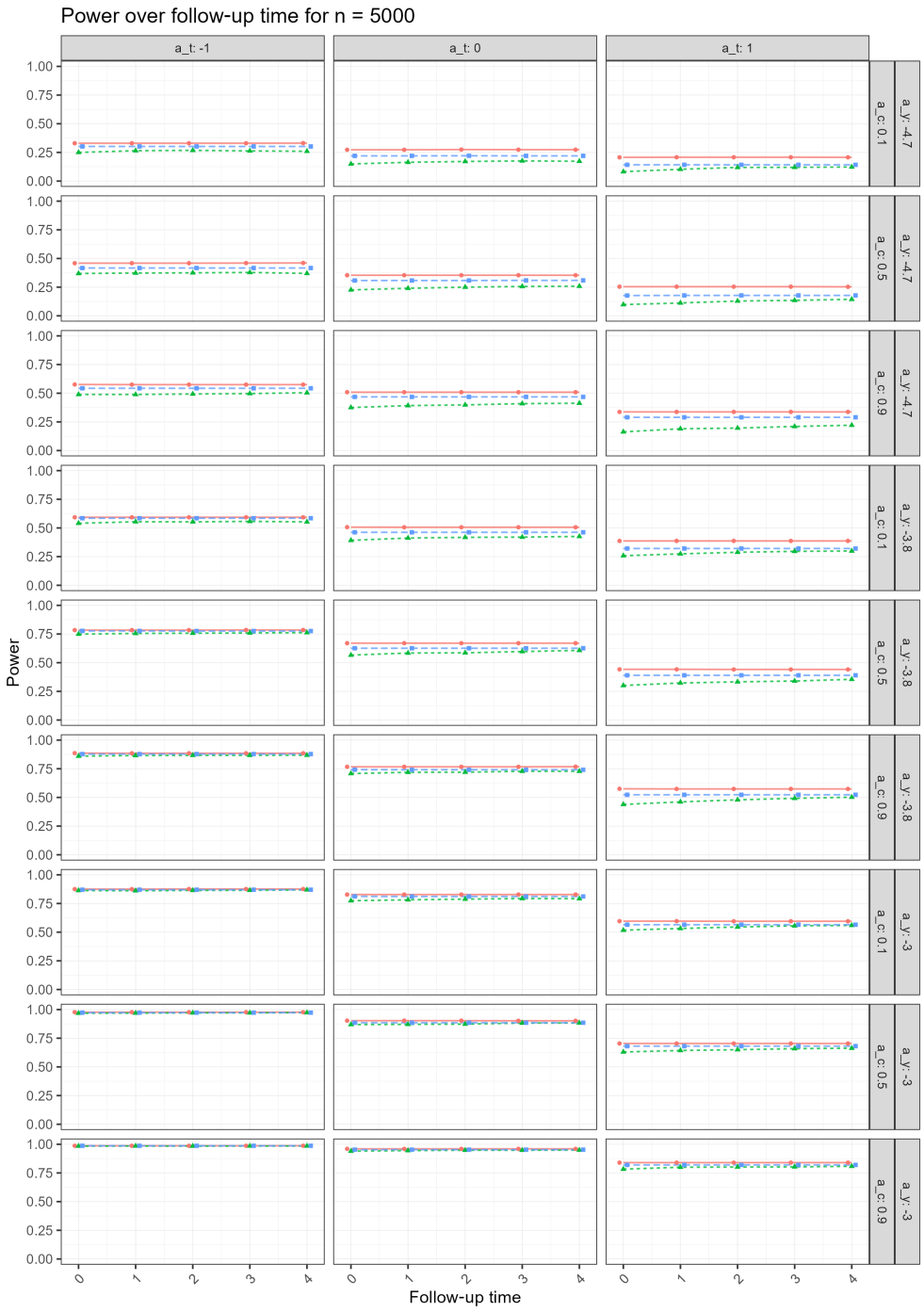

Fig. D12 Power of the 95% confidence interval over 1000 simulations with sample size $n = 5000$. The green line denotes the empirical bootstrap CI, the blue line denotes percentile bootstrap CI, and the red line denotes sandwich estimator CI. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

Appendix E Results Simulation: Point Estimates

In Appendix E we present all plots regarding the point estimates. The specific numbers to each plot can be found in the online supplementary materials on https://github.com/flo1met/thesis_TTE.

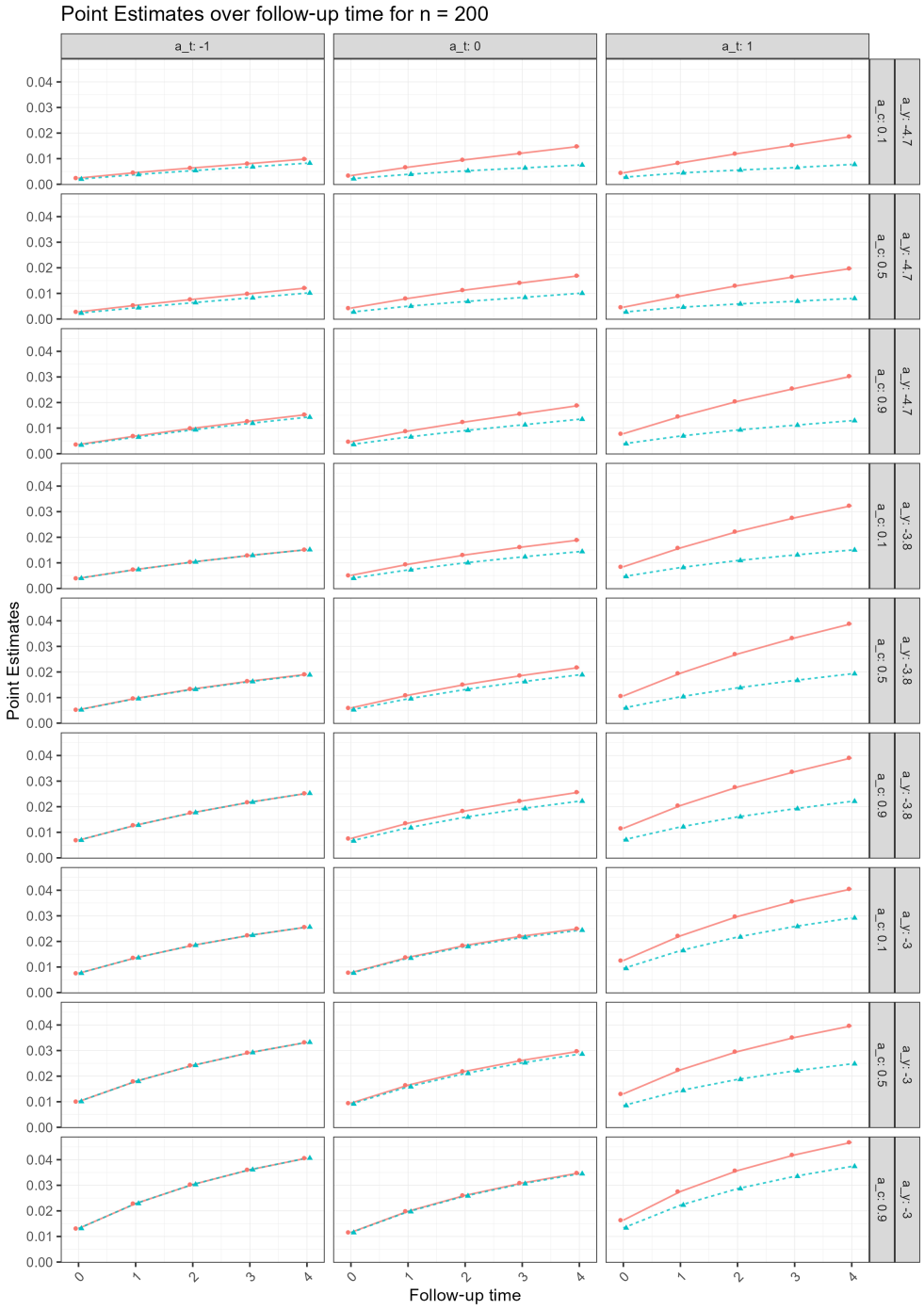

Fig. E13 Comparison of $\widehat{\text{MRD}}$ point estimates between `R` and `Julia` over 1000 simulations with sample size $n = 200$. The red line shows the `R` point estimates, and the blue line shows the `Julia` point estimates. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

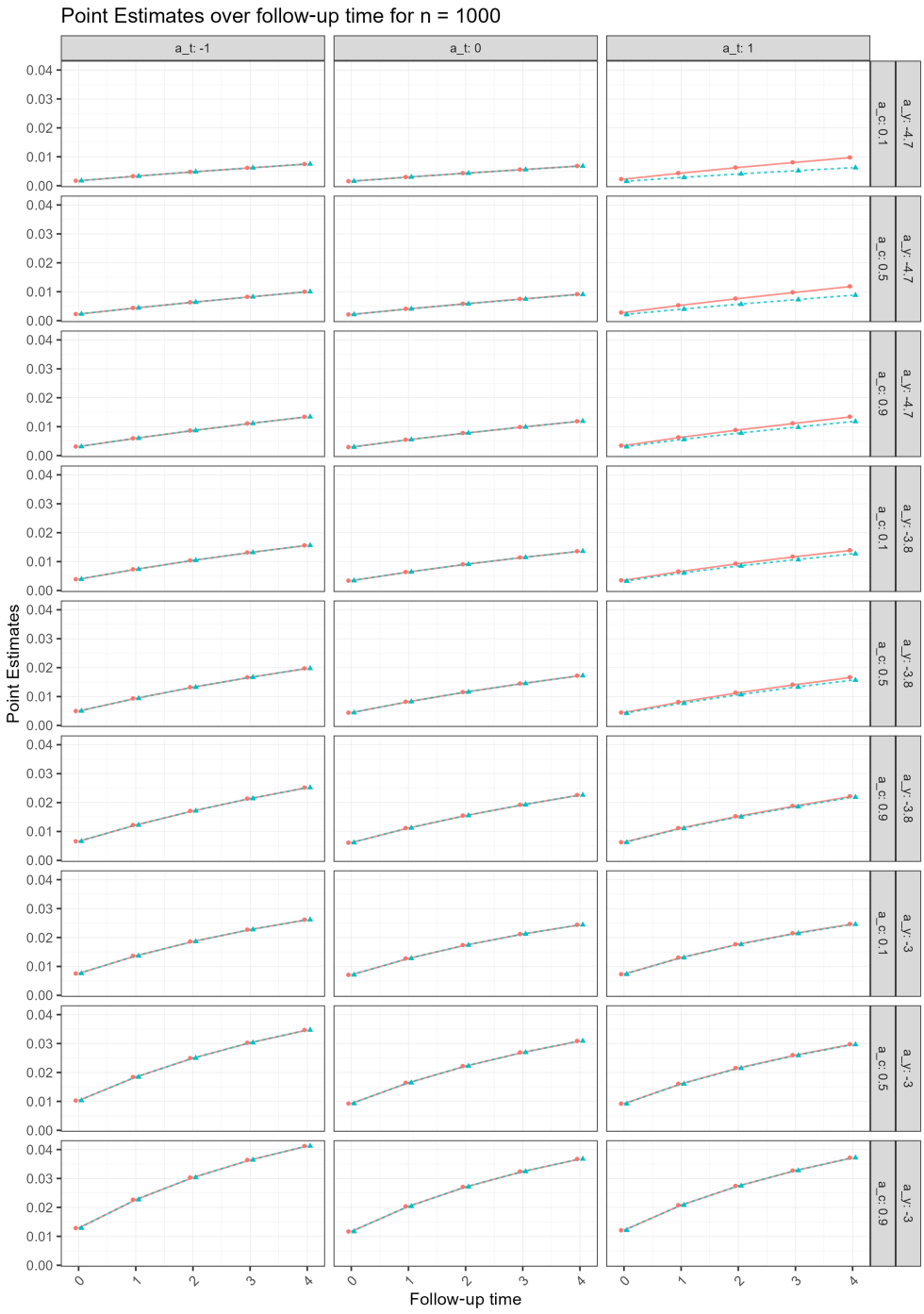

Fig. E14 Comparison of $\widehat{\text{MRD}}$ point estimates between R and Julia over 1000 simulations with sample size $n = 1000$. The red line shows the R point estimates, and the blue line shows the Julia point estimates. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

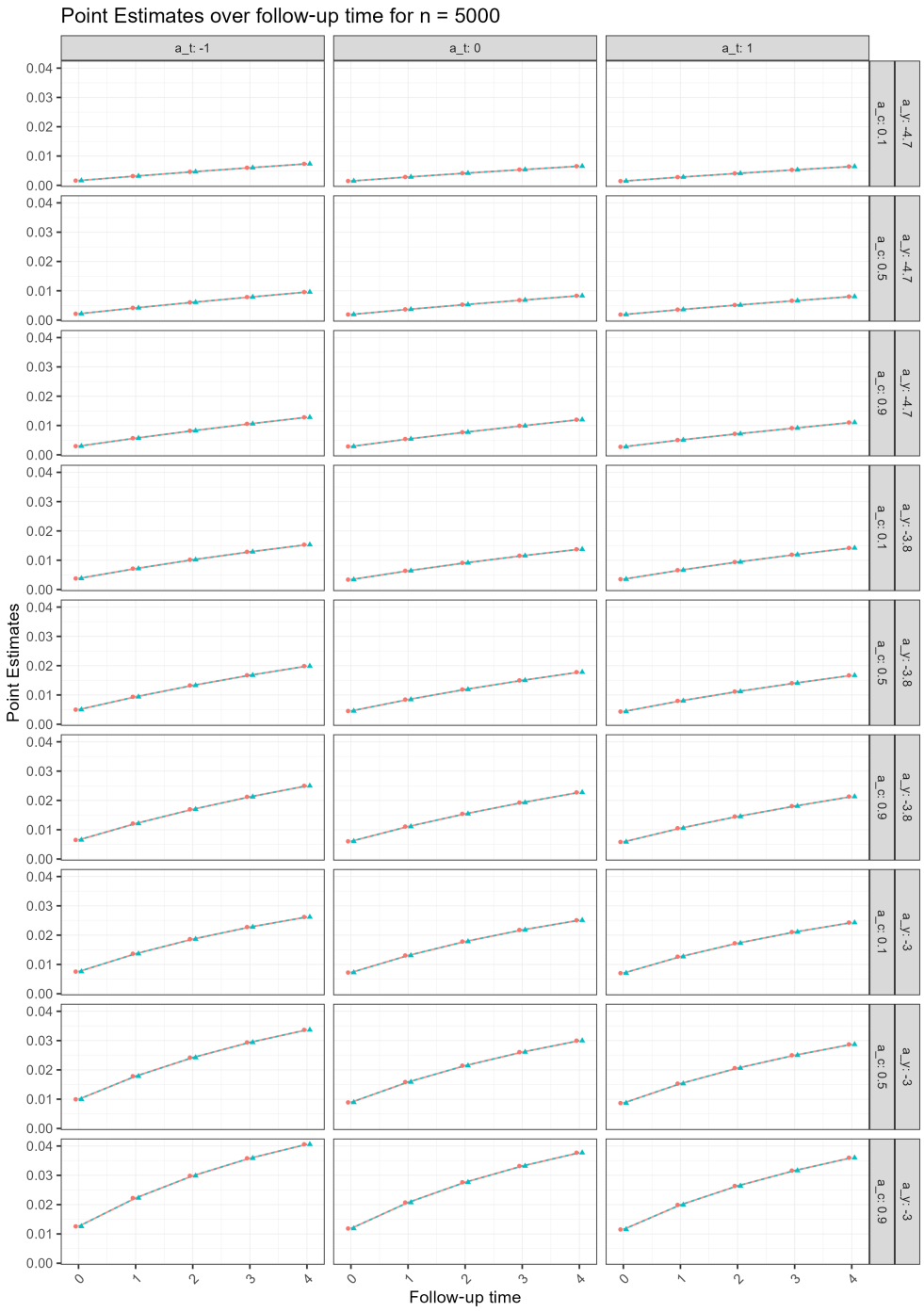

Fig. E15 Comparison of $\widehat{\text{MRD}}$ point estimates between R and Julia over 1000 simulations with sample size $n = 5000$. The red line shows the R point estimates, and the blue line shows the Julia point estimates. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

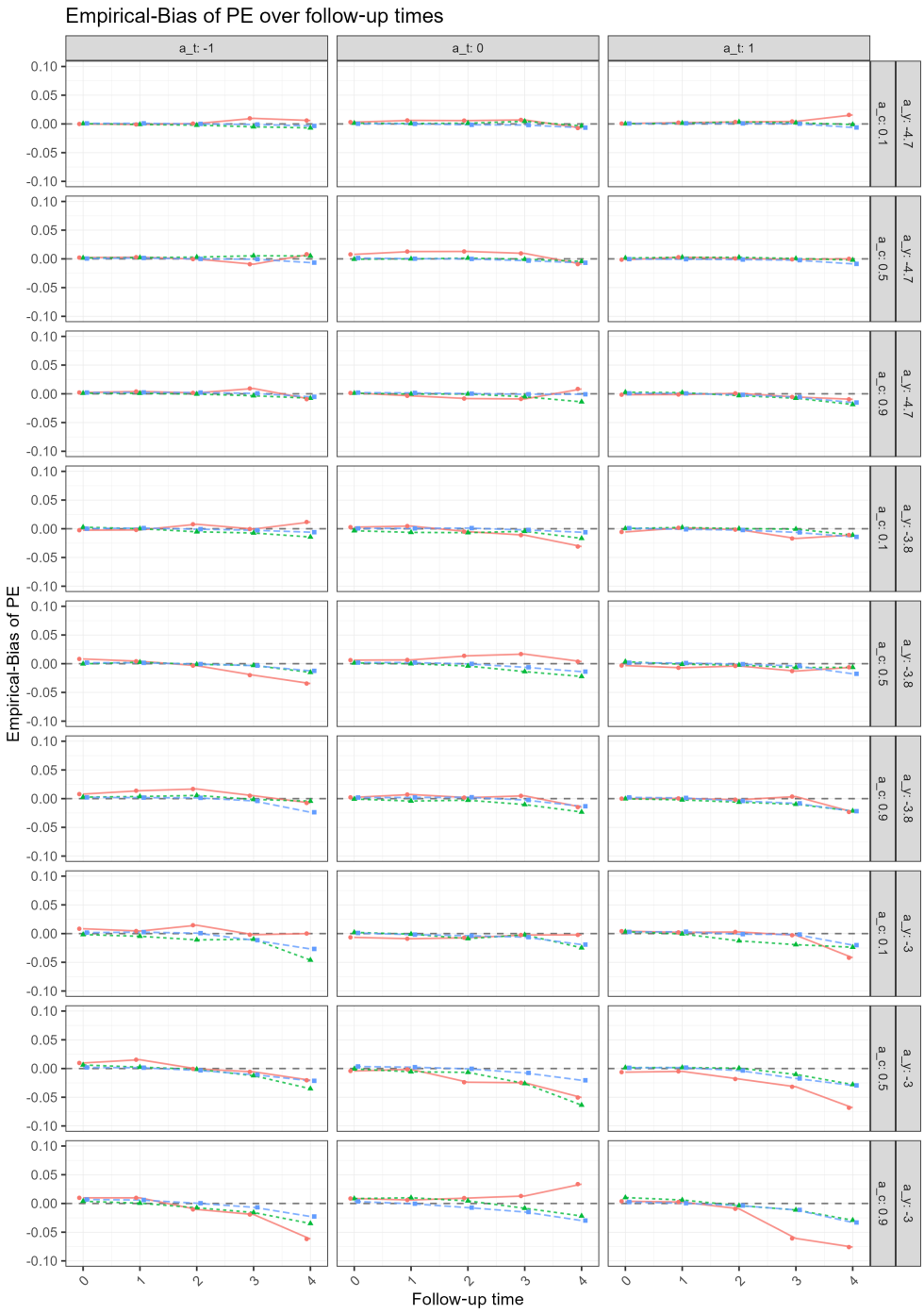

Fig. E16 Bias of the $\widehat{\text{MRD}}$ point estimates over 1000 simulations. The red line denotes the sample size $n = 200$, the green line denotes the sample size $n = 1000$, and the blue line denotes the sample size $n = 5000$. The dashed gray line marks the reference value 0. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

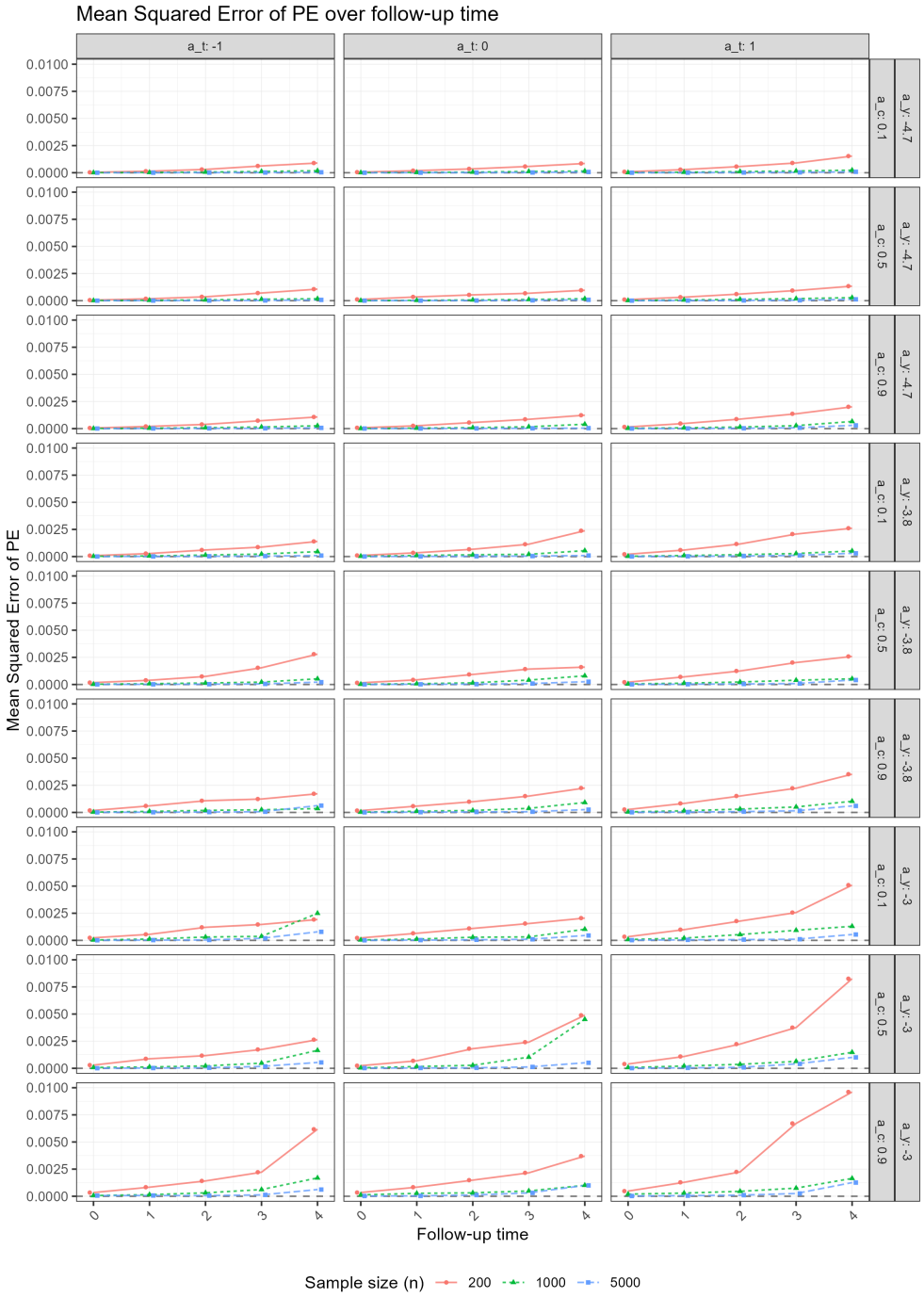

Fig. E17 Mean Squared Error (MSE) of the $\widehat{\text{MRD}}$ point estimates over 1000 simulations. The red line denotes the sample size $n = 200$, the green line denotes the sample size $n = 1000$, and the blue line denotes the sample size $n = 5000$. The dashed gray line marks the reference value 0. The outcome event rate indicator is denoted by a_y, the confounding strength is denoted by a_c, and the treatment prevalence indicator is denoted by a_t.

Appendix F Performance Comparison `Julia` and `R`

In Appendix F we present a performance comparison between the `TargetTrialEmulation.jl` and `TrialEmulation` packages with respect to runtime and memory usage.

F.1 Background

Writing the `TargetTrialEmulation.jl` package in `Julia` instead of extending existing software in `R` is motivated by the performance limitations of `R`, as discussed in Section 2.3. By leveraging `Julia`'s speed and memory efficiency, it is possible to scale analyses to larger datasets without compromising performance. To substantiate this choice, a second small-scale simulation study is conducted to compare the computational performance of `R` and `Julia` in terms of memory usage and computation time. To do this, data is generated for fixed parameters (outcome event rate indicator $a_y = -3.8$, confounding strength $a_c = 0.5$, treatment prevalence indicator $a_t = 0$) with sample sizes $n = \{2e^2, 1e^3, 5e^3, 2e^4, 5e^4, 1e^5\}$. Subsequently, sequential TTE is performed using `TrialEmulation` and `TargetTrialEmulation.jl`. Performance is evaluated under two conditions: first, excluding confidence interval estimation, and second, including the estimation of confidence intervals using sandwich-type estimators in `TrialEmulation`, and using bootstrap estimators in `TargetTrialEmulation.jl`.

Each configuration was replicated five times per sample size to assess variability in computational performance. While five replications may be insufficient to fully capture the extent of variability, prior experiments indicated small variation across runs. Nevertheless, future research should consider increasing the number of replications to quantify performance variability more robustly. To assess memory efficiency, we measure memory allocations, which capture the total amount of memory requested by the program during execution. This metric provides a comprehensive view of memory usage over the course of the computation. It is relevant for evaluating the scalability and efficiency of iterative and data-intensive procedures such as sequential TTE.

All simulations are run in `R` version `4.5.0` (R Core Team 2024) and `Julia` version `1.11.2` (Bezanson et al. 2017). The `TrialEmulation` package is used in version `0.0.4.2` (Su et al. 2024). Execution time and allocated memory are measured with the `@time` macro in `Julia`. In `R` time is measured using the base command `sys.time()` and allocated memory is measured using the `Rprofmem()`. The simulation is run on Windows 11 with an Intel Core i5-1335U, with 16 gigabytes of RAM. All scripts and results can be found on GitHub[3].

F.2 Results

The results of the performance comparison between `Julia` and `R` show a clear trend in both investigated scenarios. Figure F18 shows that in the direct comparison of the same task (only performing sequential TTE), `Julia` substantially outperforms `R` in terms of speed as well as memory allocated during the process. This confirms the expected behaviour. Figure F19 also shows a clear trend. `TrialEmulation`'s

[3] https://github.com/flo1met/thesis_TTE

implemented sandwich-type CIs are significantly faster and use fewer total memory allocations than `TargetTrialEmulation.jl`'s bootstrap CIs. Even though `Julia` is the by far more computationally efficient programming language, the high computational complexity of the resampling process that is necessary to estimate bootstrap confidence intervals, outweighs the computational advantages that `Julia` has over `R`.

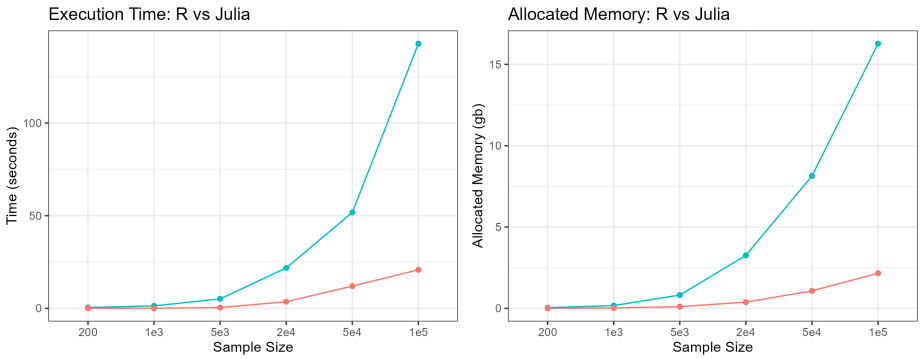

Fig. F18 Computation time and memory allocations of `TargetTrialEmulation.jl` and `TrialEmulation` across increasing sample sizes, excluding confidence interval estimation. Results are averaged over five replications. The red line marks `Julia`, the blue line marks `R`.

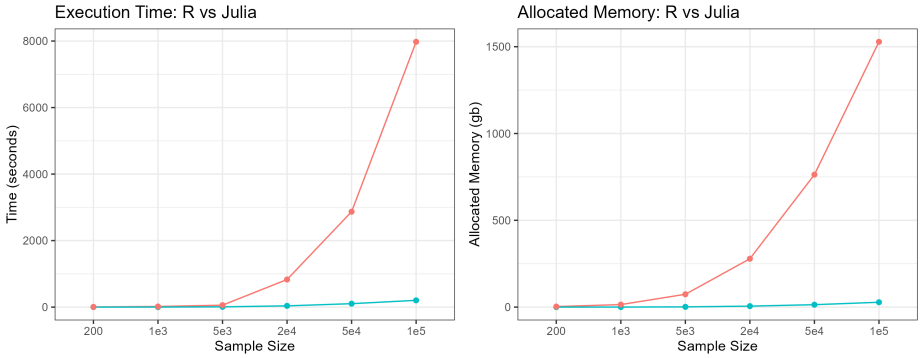

Fig. F19 Computation time and memory allocations of `TargetTrialEmulation.jl` and `TrialEmulation` across increasing sample sizes, including confidence interval estimation. Results are averaged over five replications. The red line marks `Julia`, the blue line marks `R`.

A limitation of the study is that `Julia` has longer pre-compilation times due to its just-in-time compiler. While this extra time is negligible in big samples, it will affect the estimation time of small samples. To adequately compare the performance of the `TargetTrialEmulation.jl` and the `TrialEmulation` package, the aim should be to closely replicate a realistic analysis scenario. This is left open for future research.

A further limitation is that only the total allocated memory was measured. While this gives a good measure of memory used and computational efficiency, to investigate possible hardware limitations, the peak memory usage (the highest amount of memory used in one process) should also be measured.

Overall, the results confirm that `Julia` provides substantial performance advantages over `R` for sequential TTE, particularly in terms of speed and memory efficiency. However, the computational cost of bootstrap resampling remains high even in `Julia`, highlighting a trade-off between estimator robustness and scalability. As concluded in Section 4, future research should focus on improving inference with bootstrap methods, while also enhancing computational scalability.

Appendix G Bootstrap Distributions

During the simulation outlined in Section 2.4, the bootstrap distributions were not saved. Future research on bootstrap methods for sequential TTE should save and systematically evaluate them to evaluate their performance and find indicators for possible improvements. Following, we reran one simulation scenario with all three sample sizes to show an example of the skewed distributions. While this is not a comprehensive simulation study, it indicates the bottlenecks of the currently implemented percentile and empirical bootstrap methods.

Figures G20-G22 show that all bootstrap distributions are skewed. This leads to problems in the estimation of the confidence intervals, because the empirical bootstrap method assumes approximate symmetry of the sampling distribution, which does not hold in this case, potentially resulting in biased or misleading interval bounds, as discussed in Section 4.

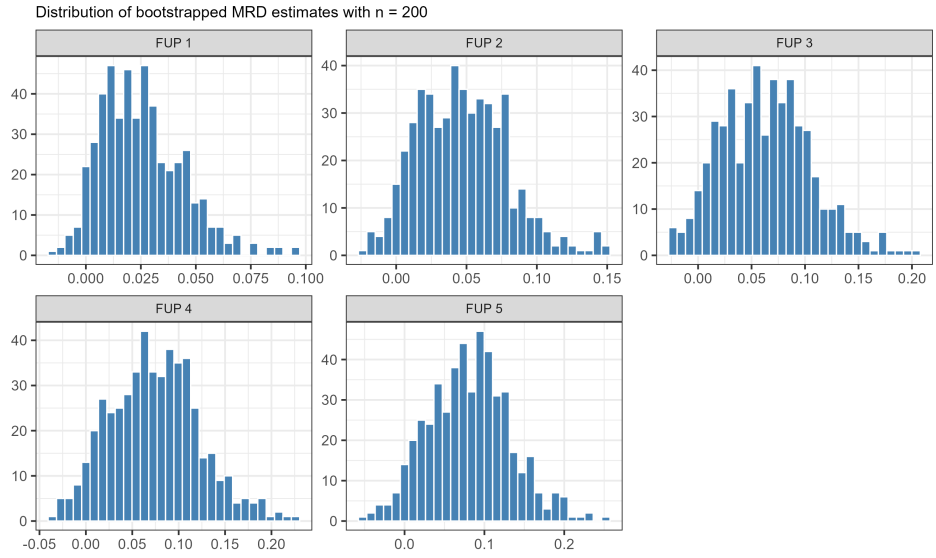

Fig. G20 Distribution of bootstrapped $\widehat{\text{MRD}}$ estimates across five follow-up periods (FUP 1–5) based on $n = 200$ resamples. Each panel displays the empirical distribution of the point estimate, capturing the sampling variability of the marginal risk difference over time.

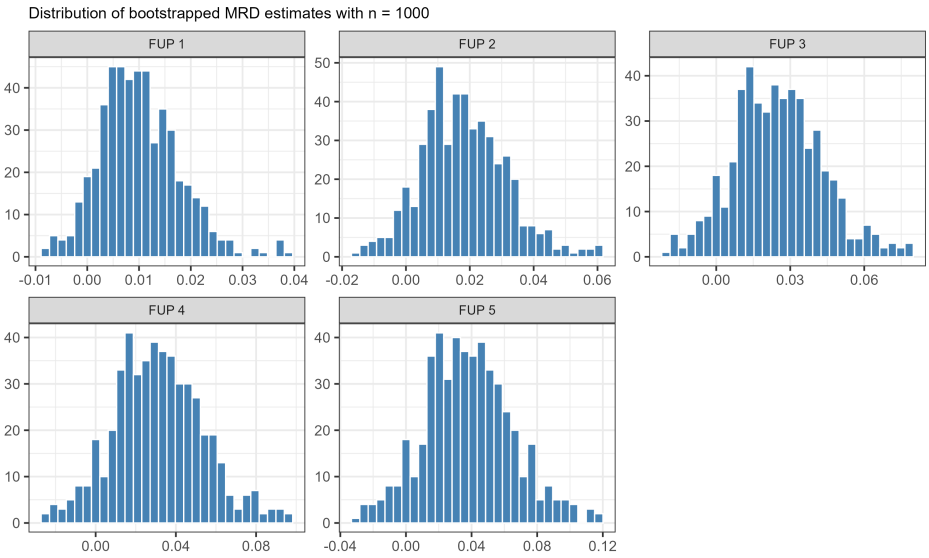

Fig. G21 Distribution of bootstrapped $\widehat{\text{MRD}}$ estimates across five follow-up periods (FUP 1–5) based on $n = 1000$ resamples. Each panel displays the empirical distribution of the point estimate, capturing the sampling variability of the marginal risk difference over time.

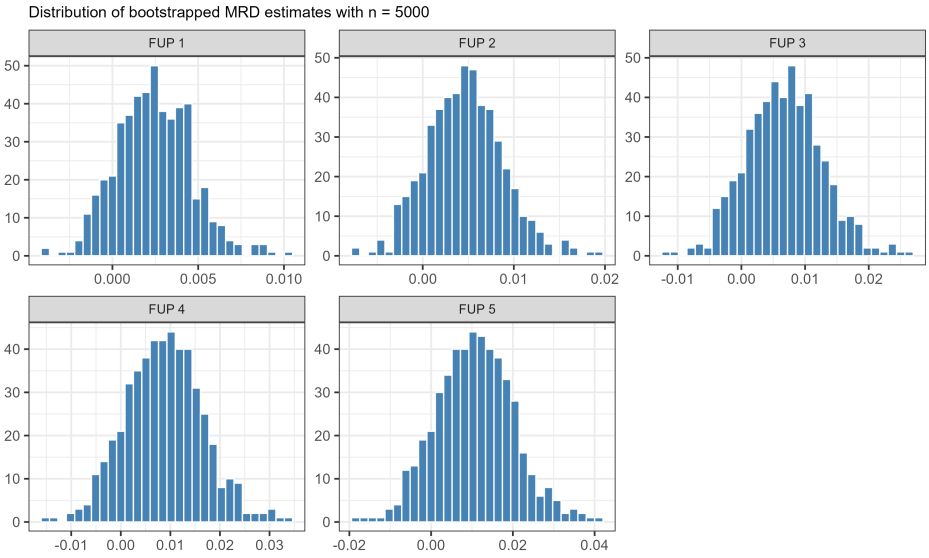

Fig. G22 Distribution of bootstrapped $\widehat{\text{MRD}}$ estimates across five follow-up periods (FUP 1–5) based on $n = 5000$ resamples. Each panel displays the empirical distribution of the point estimate, capturing the sampling variability of the marginal risk difference over time.