Federated Average Treatment Effect Estimation for Multi-Site RCTs with DataSHIELD

Francisco Andrade†[1,2*] **Rémi Khellaf**†[1] **Julie Josse**[1]

[1]PreMeDICaL, Inria, Inserm, Idesp, UMR 1318, Université de Montpellier [2]UMR1346, HeKA, Inserm, Inria, Université Paris Cité

Date published: 2026-11-15 Last modified: 2026-11-15

Abstract

We introduce `dsFedAte` and `dsFedAteClient`, two companion R packages for federated estimation of the average treatment effect (ATE) from multiple randomized controlled trials (RCTs), without sharing individual-level data. Built on the open-source DataSHIELD privacy-preserving infrastructure, they implement a covariate-adjusted G-formula estimator accommodating heterogeneous covariate distributions and study effects. Unlike iterative federated optimization methods, our approach fits arm-specific linear outcome models in a single communication round by aggregating local sufficient statistics, yielding estimates identical to those obtained from centrally pooled data. The entire analysis is accessible through a single client-side function, `ds.EstimateAte()`. Beyond implementation, we also present methodological contributions. First, we derive the asymptotic variance of the ATE estimator, accounting for heterogeneous covariate distributions and study effects, and express its estimator in terms of locally aggregable quantities, enabling federated confidence intervals equivalent to those obtained from pooled data. Second, we establish consistency and asymptotic normality under misspecified linear outcome models, showing that covariate adjustment is asymptotically at least as efficient as the unadjusted stratified difference-in-means estimator. We illustrate the packages through reproducible examples using the DSLite simulation environment, covering homogeneous and heterogeneous populations, DataSHIELD disclosure controls, and adjustment for study effects. A simulation study validates the theoretical results and demonstrates nominal confidence interval coverage.

Keywords: federated learning, causal inference, average treatment effect, DataSHIELD, R package, privacy-preserving statistics, multi-site RCT

submitted

Contents

1 Introduction **2**
1.1 Contributions. . . . 3

2 Background and Problem Statement **4**
2.1 Multicenter Causal Inference 4
2.1.1 Linear Outcome Model 5
2.2 Federated Learning and the DataSHIELD Architecture 6
2.2.1 DataSHIELD 6
2.2.2 Disclosure controls. . . . 7
2.2.3 Local simulation with DSLite. . . . 7
2.3 Estimation goals and method 8

*Corresponding author: francisco.de-lima-andrade@inria.fr

3 Package Design **9**
3.1 Client package: `dsFedAteClient` . . . 9
3.1.1 Pipeline Overview . . . 9
3.1.2 Arguments . . . 10
3.1.3 Return Values . . . 11
3.2 Server package: `dsFedAte` . . . 11
3.2.1 `localAteDS()`. . . . 11
3.2.2 `addStudyIdDS()`. . . . 12
3.3 Data Flow Summary . . . 12

4 Tutorial Examples **13**
4.1 Shared parameters . . . 14
4.2 Helpers . . . 14
4.3 Example 1 — Homogeneous populations . . . 14
4.4 Example 2 — Disclosure thresholds and privacy levels . . . 15
4.5 Example 3 — Heterogeneous covariate distributions . . . 16
4.6 Example 4 — Study effects . . . 17
4.7 Example 5 — Consistency under model misspecification . . . 18
4.8 Coverage summary . . . 20

5 Discussion **20**

References **21**

Appendix A: Variance **22**
Proof of Proposition 1 . . . 22

Appendix B: Computing the Variance from Decentralised Data **23**
The Last Term in the Variance Formula . . . 23
The Remaining Terms . . . 24

Appendix C: Misspecified linear G-formula asymptotic normality **25**

Appendix D: IRLS vs FedAvg **27**

Session information **28**

[†]These authors contributed equally to this work.

1 Introduction

Causal inference aims at evaluating the effect of a treatment on an outcome in a specified population. In evidence-based medicine, randomized controlled trials (RCTs) are considered the gold standard for this task, providing unconfounded estimates of average treatment effects (ATEs) that drive high-stakes clinical and regulatory decisions (French Health Authority 2024). However, individual trials often suffer from limited sample sizes and restrictive eligibility criteria, limiting their precision and external validity (Rothwell 2005). Combining information across multiple studies mitigates these limitations, making evidence synthesis through meta-analysis the pinnacle of the clinical evidence hierarchy (Barratt et al. 1999; Evans 2003). Meta-analysis can be conducted using either aggregate data (AD) or individual participant data (IPD). While AD meta-analysis is convenient, IPD meta-analysis offers greater statistical flexibility, enabling covariate adjustment, harmonization of outcomes, and rigorous handling of site effects (Riley and Steyerberg 2010; Burke et al. 2017). Unfortunately, pooling IPD in

submitted

a centralized repository is frequently prohibited by privacy regulations, institutional governance, and logistical barriers, forcing researchers to rely on less informative study-level summaries.

Federated learning (McMahan et al. 2017) offers a decentralized paradigm to bypass these data-sharing bottlenecks. In a federated network, algorithms are trained collaboratively through the exchange of intermediate aggregate quantities, while the underlying individual-level observations remain local. Recently, this paradigm has been extended to *federated causal inference* (Xiong et al. 2023; Khellaf et al. 2024), enabling the statistical benefits of an IPD meta-analysis without pooling raw data. Under suitable conditions, federated procedures can output causal effect estimates and variances that match or approximate the results obtained from centrally pooled analyses.

Despite the theoretical maturity of federated causal inference, a major barrier to its widespread adoption is the lack of practical software implementations. While there exist deployment platforms for standard federated machine learning tasks (He et al. 2020; Ziller et al. 2021), there is not for its causal inference methods, hindering their diffusion among researchers.

To bridge this gap, we propose a practical deployment of federated causal inference using `DataSHIELD` (Wolfson et al. 2010; Gaye et al. 2014). `DataSHIELD` is an established R-based framework engineered to perform privacy-preserving distributed analyses. It allows participating studies to execute local computations and return only disclosure-controlled summary quantities to a central analysis server. In this article, we extend the `DataSHIELD` ecosystem by introducing methods for the federated estimation of ATEs across multiple RCTs, providing researchers with a secure, deployable tool to conduct covariate-adjusted IPD meta-analyses without ever centralizing participant-level records.

1.1 Contributions.

We introduce two companion R packages, `dsFedAte` (server-side) and `dsFedAteClient` (client-side), implementing within DataSHIELD the federated ATE estimator of Khellaf et al. (2024) using decentralized IPD from multiple RCTs. The implementation builds on native DataSHIELD functions and allows individual-level patient data to remain at their respective sites. Unlike the iterative optimization scheme proposed by Khellaf et al. (2024), we introduce a one-shot federated procedure, better suited to DataSHIELD's `dsBase` package and more communication-efficient for moderate covariate dimensions. By aggregating local covariance matrices and covariate–outcome cross-products, it estimates the outcome-model parameters in a single communication round and yields the same ATE as the centralized pooled estimator.

We provide two methodological contributions, beyond the implementation. First, we derive the asymptotic variance in the general setting, accounting for heterogeneous covariate distributions and site effects. We express this variance as a sum of locally computable contributions that can be aggregated centrally, enabling federated variance estimation and the construction of confidence intervals. Second, we extend the known results of linear adjustment in a single RCT to the multiple RCTs setting. We also prove the consistency and asymptotic normality of the estimator even when the outcome models are estimated with misspecified linear regressions (Proposition 2).

Finally, our work contributes to a growing movement of theory-backed implementations for federated causal inference. A complementary approach is available in the `dsMatching` and `dsMatchingClient` DataSHIELD packages by Gusinow et al. (2026), who operate in a multi-site observational data setting. Under the assumption of homogeneous nuisance functions across sites, they focus on the federated estimation of parametric propensity scores to derive matching-based estimators. In contrast, we operate within a multiple RCT setting with heterogeneous outcome distributions. We focus on federated g-formula estimators, accommodating site effects and guaranteeing robustness to model misspecification.

Submitted

2 Background and Problem Statement

2.1 Multicenter Causal Inference

We represent each individual by a realization of the random tuple (H, X, A, Y), where $H \in [K]$ denotes the study/site/center indicator, $A \in \{0, 1\}$ the binary treatment indicator, X a d-dimensional vector of pre-treatment covariates, and Y the observed outcome. Following the potential outcomes framework of (Rubin 1974), we posit the existence of two counterfactual random variables Y^0 and Y^1, representing the outcomes an individual would have experienced under treatment 0 or 1, respectively. We link these potential outcomes to the observed outcome Y through the Stable Unit Treatment Value Assumption (SUTVA), which requires that there is no interference between individuals and no hidden versions of treatment.

Assumption 1 (SUTVA). $Y = AY^1 + (1 - A)Y^0$.

To compare two treatments, one typically specifies a contrast function Φ and targets $\Phi(\mathbb{E}[Y^1], \mathbb{E}[Y^0])$. In this work, we focus on the risk difference $\Phi(x, y) = x - y$, yielding the **estimand of interest**:

$$\tau := \mathbb{E}[Y^1 - Y^0]. \tag{ATE}$$

In practice, for any given individual, only one of the two potential outcomes Y^1 or Y^0 is ever observed, a fact known as the *fundamental problem of causal inference*. The goal is nonetheless to estimate τ from the observable quantities (H, X, A, Y), which in general requires imposing assumptions on the joint distribution of (H, X, A, Y^1, Y^0). Beyond SUTVA, we adopt the following conditions on the treatment assignment mechanism.

Assumption 2 (RCT within sites). *Treatment is randomised within each site:* $A \perp\!\!\!\perp (Y^0, Y^1) \mid H$.

Assumption 3 (Positivity). *For every* $k \in [K]$, $0 < p_k := \Pr(A = 1 \mid H = k) < 1$.

Denote by P_k the covariate distribution in study k, and set $\rho_k = P(H = k)$. Assumptions 1 to 3 together yield the identification formula

$$\tau = \sum_k \rho_k \, \mathbb{E}_{P_k}[\mu_1(X, k) - \mu_0(X, k)]\,. \tag{G-formula}$$

where the *response function* μ_a, for $a \in \{0, 1\}$, is defined as

$$\mu_a(x, k) := \mathbb{E}[Y \mid H = k,\ X = x,\ A = a]. \tag{1}$$

To estimate τ via (G-formula) from the observed data $\mathscr{D} = (H_i, X_i, A_i, Y_i)_{i=1}^n$, we proceed in two steps: we first fit estimators $\hat{\mu}_1$ and $\hat{\mu}_0$ on the data, and then plug them into (G-formula). For each site k, define the local treatment effect estimate

$$\hat{\tau}_k := \frac{1}{n_k} \sum_{i:\, H_i = k} \big[\hat{\mu}_1(X_i, k) - \hat{\mu}_0(X_i, k)\big], \tag{2}$$

where $n_k := \#\{i : H_i = k\}$. The overall estimator is then the weighted average

$$\hat{\tau} = \sum_k \frac{n_k}{n} \cdot \hat{\tau}_k. \tag{3}$$

Throughout this work, $\hat{\mu}_1$ and $\hat{\mu}_0$ are specified as **linear models**, that is, for $a \in \{0, 1\}$,

$$\hat{\mu}_a(x, k) = \hat{d}_a^\top k + \hat{\beta}_a^\top x,$$

submitted

where

$$(\hat{d}_a,\, \hat{\beta}_a) = \underset{d,\,\beta}{\operatorname{argmin}} \sum_{i=1}^{n} \left(Y_i - d^\top H_i - \beta^\top X_i\right)^2 \mathbb{1}_{[A_i=a]}, \tag{4}$$

with H encoded as a one-hot vector.

2.1.1 Linear Outcome Model

The tutorial examples in Section 4 all rely on a linear outcome model, implying that $\hat{\mu}_1$ and $\hat{\mu}_0$ are correctly specified. The function `dsFedAteClient::ds.EstimateAte` provides two optional logical parameters, `study_effects` and `heterogeneous`, which encode assumptions about the outcome model. Setting `study_effects` to `FALSE` imposes the constraint $\mu_a(\cdot, k) = \mu_a(\cdot, l)$ for all $k, l \in [K]$, while setting `heterogeneous` to `FALSE` assumes that the covariate distributions satisfy $P_k = P_l$ for all $k, l \in [K]$. Under linearity, the outcome model takes the form

$$Y^a = d_a^\top H + \beta_a^\top X + \varepsilon_a, \tag{5}$$

where $H \in \{0,1\}^K$ is treated as a one-hot encoded vector (by a slight abuse of notation), $\mathbb{E}[\varepsilon_a | X, H] = 0$. The *no study effects* assumption corresponds to

$$Y^a = c_a + \beta_a^\top X + \varepsilon_a \tag{6}$$

so that to obtain $\hat{\mu}_1$ and $\hat{\mu}_0$, Y is regressed on X alone.

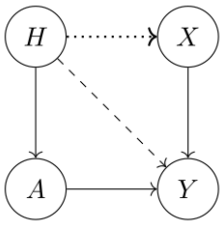

Figure 1: The directed acyclic graph representing the causal structure of the fully heterogeneous setting, with heterogeneous populations and a center effect. Setting `study_effects` to `FALSE` amounts to removing the dashed arrow from H to Y, while setting `heterogeneous` to `FALSE` corresponds to removing the dotted arrow from H to X.

Proposition 1 (Variance, proved in Appendix A). Let $Z = (H^\top, X^\top)^\top$ and $\theta_a = (d_a^\top, \beta_a^\top)^\top$. Under Model Equation 5, and homoscedastic errors $\mathbb{V}(\varepsilon_a \mid Z) = \sigma_a^2$, $\hat{\tau}$ satisfies $\sqrt{n}(\hat{\tau} - \tau) \xrightarrow{d} \mathcal{N}(0, V)$, with

$$V = \sigma_1^2 \mathbb{E}[Z]^\top M_1^{-1} \mathbb{E}[Z] + \sigma_0^2 \mathbb{E}[Z]^\top M_0^{-1} \mathbb{E}[Z] + \mathbb{V}(Z^\top \Delta), \tag{7}$$

where $\Delta = \theta_1 - \theta_0$, $M_a = \mathbb{E}[\mathbb{1}(A = a) Z Z^\top]$. If we further assume no study effects and fit outcome models on $X' = (1, X^\top)^\top$, we get

$$V = \sigma_1^2 \mathbb{E}[X']^\top \tilde{M}_1^{-1} \mathbb{E}[X'] + \sigma_0^2 \mathbb{E}[X']^\top \tilde{M}_0^{-1} \mathbb{E}[X'] + \mathbb{V}((\beta_1 - \beta_0)^\top X), \tag{8}$$

where $\tilde{M}_a = \mathbb{E}[\mathbb{1}(A = a) X' X'^\top]$. Finally, if the distribution of covariates is homogeneous across studies,

$$V = \frac{\sigma_1^2}{p} + \frac{\sigma_0^2}{1-p} + \mathbb{V}((\beta_1 - \beta_0)^\top X). \tag{9}$$

submitted

A notable robustness result (Lin 2013) shows that in an RCT, even when the true relationship between covariates and potential outcomes is nonlinear, fitting separate linear models for $\hat{\mu}_a$ and using them to impute potential outcomes still yields a consistent—though only asymptotically unbiased—estimate of the average treatment effect τ. However, this result applies to a single RCT, whereas in our setting the full pooled data is not an RCT—only the data within each study is. This distinction necessitates a generalization of the above result, the proof of which is deferred to the appendix.

Proposition 2 (Asymptotic normality with misspecified linear outcome models, proved in Appendix C). *Let the empirical G-formula estimator $\hat{\tau}$ Equation 3 be constructed using the arm-specific OLS predictions $\hat{\mu}_a(X, H) = \hat{d}_a^\top H + \hat{\beta}_a^\top X$, where H is a vector of one-hot encoded site indicators and $(\hat{d}_a, \hat{\beta}_a)$ obtained as in Equation 4. Under Assumption 2 and Assumption 3, and $X \perp\!\!\!\perp A \mid H$, even if the true conditional expectations are non-linear in X, $\hat{\tau}$ satisfies*

$$\sqrt{n}(\hat{\tau} - \tau) \xrightarrow{d} \mathcal{N}(0, V^*),$$

where V^* is the asymptotic variance. Furthermore, V^* is less than or equal to the asymptotic variance of the unadjusted stratified difference-in-means estimator.

Remark. In particular, this result establishes that when the outcome is binary, one may still use a linear regression to estimate $\hat{\mu}_a$ as a means to estimate τ. Moreover, the second part of the proposition justifies the sole focus on the G-formula throughout the rest of the article.

Remark. This extends the asymptotic guarantees of Wager (2024) to multi-site settings. While Liu and Yang (2020) derive a similar variance bound in a stratified RCT, they assume a finite population and use a single pooled and fully interacted regression. In contrast, we use a super-population framework and prove that arm-specific G-formula regressions achieve this bound including site indicators, avoiding centering and interactions.

2.2 Federated Learning and the DataSHIELD Architecture

In multicenter settings, collaboration can increase sample size and population diversity, but legal, privacy, competitive, or logistical constraints may prevent centralizing individual-level data.

Federated computing addresses this by keeping raw observations local while exchanging only aggregated information. It includes federated analytics, which enables collaborative data analysis; and federated learning, which trains a global model without centralizing sites' raw data. Thus, federated learning is a set of statistical and optimization methods for learning shared models under data-sharing constraints.

Federated architectures can be organized around a star-shaped graph, known as *client–server* architecture, where sites communicate through a central coordinator, or a fully decentralized architecture, such as peer-to-peer learning, where sites communicate directly without a central server. Finally, two main settings related to the number of participating sites K are distinguished (Kairouz and McMahan 2021). In cross-device settings, a large number of intermittently available devices, such as smartphones, collaborate. In a *cross-silo* setting, collaboration is performed by a smaller number of stable organizations, such as hospitals, companies, or research centers.

2.2.1 DataSHIELD

DataSHIELD is an open-source platform for privacy-aware federated analysis of sensitive individual-level data, built around the principle of "taking the analysis to the data, not the data to the analyst" (Gaye et al. 2014). It follows a central client–server architecture (see Figure 2): an analyst interacts through a central client, which sends approved commands to servers located at participating institutions, where the data remain stored locally (Wilson et al. 2017). Note that this terminology is

submitted

reversed in the federated learning literature, where the central coordinating entity is typically called the *server* and participating institutions are referred to as *clients*. DataSHIELD was initially developed mainly for federated analytics, with sites returning disclosure-controlled summary statistics, but its ecosystem now also includes statistical tasks such as fitting distributed linear regressions.

Its architecture combines server-side R packages, such as `dsBase`, installed within each institution's infrastructure, with client-side packages, such as `dsBaseClient`, through which the analyst submits approved analysis commands. Each site's data are typically managed through an **Opal** repository, which restricts execution to whitelisted functions and returns only disclosure-controlled outputs. Communication between the client and institutional servers is authenticated and encrypted, allowing distributed analyses to be centrally orchestrated while individual-level data remain behind each site's firewall.

Server-side functions process the data locally and return only non-disclosive summaries — for example, score vectors and information matrices for a generalised linear model — which the client combines to produce the final result. The procedure is mathematically equivalent to analysing all data in a single database, while respecting the federated constraint. The present package accesses DataSHIELD through the package `DSI` [Marcon (2025a)][,] an abstraction layer that defines a common interface for connecting to DataSHIELD-compliant data repositories regardless of their back-end implementation (e.g. Opal, DSLite).

2.2.2 Disclosure controls.

In DataSHIELD, privacy is understood primarily as a disclosure control. It prevents the execution of functions whose outputs may reveal individual-level information or enable re-identification or inference attacks. It is enforced operationally at the server side through two mechanisms. First, each data server operates at a configurable *privacy control level* that determines which server-side functions are permitted (DataSHIELD, n.d.). Five levels are defined, ranging from `"permissive"` (no restrictions) through `"avocado"` (blocks known subsetting attacks), `"banana"` (blocks known inference attacks), and `"carrot"` (blocks both some subsetting and some inference attacks) to `"non-permissive"` (blocks all known attacks). The default is `"banana"`, which permits the subsetting and GLM operations used by this package. Second, a set of numeric disclosure thresholds — retrieved via `dsBase::listDisclosureSettingsDS()` — impose additional constraints on every result returned. The two most relevant to this package are:

- `nfilter.subset`: the minimum number of rows a dataset must contain before any result is returned (default: 3).
- `nfilter.glm`: the maximum ratio of model parameters to observations; GLMs that exceed this ratio relative to the local arm size are blocked (default: 0.33, requiring at least ~ 3 observations per parameter).

Further safeguards, such as suppression of contingency-table cells with low counts and omission of extrema from quantile functions, are built directly into individual server-side functions (Gaye et al. 2014; Wilson et al. 2017).

2.2.3 Local simulation with DSLite.

The `DSLite` package (Marcon 2025b) emulates a multi-server DataSHIELD deployment on a single machine by running lightweight virtual servers inside the local R session. All examples in this article use DSLite: they are fully self-contained and reproducible without institutional server access. In a real multi-site study the analyst would replace the DSLite login configuration with credentials for the participating institutional servers; the analysis code is otherwise identical.

submitted

2.3 Estimation goals and method

We now describe the outputs of the packages. As in Section 2.1, let $\mathscr{D} = (H_i, X_i, A_i, Y_i)_{i=1}^n$ denote a set of observations and $(Y_i^1, Y_i^0)_{i=1}^n$ the corresponding potential outcomes, all satisfying Assumptions 1 to 3 for every i. For each site k, define the local data set $\mathscr{D}_k = \{(H, X, A, Y) \in \mathscr{D} : H = k\}$. As in Section 2.2, suppose that the data sets $\mathscr{D}_k$ are distributed across K sites and that all statistical analysis must be conducted without pooling individual-level records.

Goal. We aim to compute the following quantities using only functions from `dsBaseClient` without individual-level observations being sent out of their site:

1. The estimator $\hat{\tau}$ of the ATE defined in Equation 3, with $\hat{\mu}_1$ and $\hat{\mu}_0$ predicted as linear models;
2. A variance estimate $\hat{V}$ of V in Equation 7, yielding a 95 % confidence interval $\hat{\tau} \pm 1.96\hat{V}^{1/2}$.

Method. We obtain estimates of τ and V through the following two steps:

1. The coefficients $(\hat{d}_a, \hat{\beta}_a)$ defining the outcome models $\hat{\mu}_0$ and $\hat{\mu}_1$ — that is, the solutions to Equation 4 — are estimated in a federated manner using `ds.glm` from the `dsBaseClient` package. This function implements a federated variant of the Iteratively Reweighted Least Squares (IRLS) algorithm.
2. The client broadcasts the estimated coefficients $(\hat{d}_a, \hat{\beta}_a)$ to all servers. Each server k then computes its local treatment effect estimate $\hat{\tau}_k$ (see Equation 2), together with the aggregate summaries needed for the variance estimation (see Appendix B for further details), and returns these quantities to the client for final aggregation.

Remark. Estimating $\hat{\mu}_a$ for $a \in \{0, 1\}$ requires solving Equation 4 across sites without pooling individual-level data. Unlike the iterative federated gradient descent approach of Khellaf et al. (2024), we use DataSHIELD's `ds.glm`, which implements federated iteratively reweighted least squares (IRLS). For linear models, IRLS converges in a single iteration by aggregating local score vectors and information matrices. These quantities are already computed by `ds.glm`, providing the analyst with the global Hessian at no additional communication cost and thus facilitating variance estimation. See Appendix D for details and a comparison with federated gradient descent.

We now describe how to estimate V in Equation 7, Equation 8, Equation 9 using locally computable quantities that can be aggregated centrally. We first consider the general setting with heterogeneous covariate distributions and site effects (Equation 7).

- Variance of the predicted treatment effects, i.e., $\mathbb{V}(Z^\top \Delta)$. Let $\mathscr{I}_k = \{i : H_i = k\}$, $\bar{Z}_k = n_k^{-1} \sum_{i \in \mathscr{I}_k} Z_i$ and

$$S_k = \sum_{i \in \mathscr{I}_k} \left[\hat{\Delta}^\top (Z_i - \bar{Z}_k)\right]^2. \tag{10}$$

The variance $\mathbb{V}(Z^\top \Delta)$ is estimated by

$$\widehat{\mathbb{V}(Z^\top \Delta)} = \frac{1}{n-1}\left[\sum_{k=1}^K S_k + \sum_{k=1}^K n_k(\hat{\tau}_k - \hat{\tau})^2\right],$$

where $\hat{\tau}_k = \hat{\Delta}^\top \bar{Z}_k$ and $\hat{\tau} = n^{-1} \sum_k n_k \hat{\tau}_k$. Since these estimates are already available, only the S_k need to be transmitted.

- The vector $\mathbb{E}[Z]$ is estimated by $\bar{Z} = \frac{1}{n} \sum_{k=1}^K n_k \bar{Z}_k$ Thus, only site-specific covariate means and sample sizes are required.

submitted

- Residual variances. For each treatment arm $a \in \{0, 1\}$, we estimate

$$\hat{\sigma}_a^2 = \frac{1}{n^{(a)} - (K + d)} \sum_{k=1}^{K} \sum_{i \in \mathcal{I}_k} \mathbb{1}(A_i = a) \left(Y_i - \hat{\mu}_a(X_i, H_i)\right)^2 .$$

If the residual variances are assumed equal across treatment arms, we instead use

$$\hat{\sigma}^2 = \frac{1}{n - 2(K + d)} \sum_{a=0}^{1} \sum_{k=1}^{K} \sum_{i \in \mathcal{I}_k} \mathbb{1}(A_i = a) \left(Y_i - \hat{\mu}_a(X_i, H_i)\right)^2 .$$

The required residual sums of squares and degrees of freedom are already provided by `ds.glm()`.

- Inverse Gram matrices. We estimate M_a^{-1} by $\widehat{M}_a^{-1} = nG_a$, where

$$G_a = \left(\sum_{k=1}^{K} \sum_{i \in \mathcal{I}_k} Z_i Z_i^\top \mathbb{1}(A_i = a) \right)^{-1} .$$

The matrices G_a are directly available from `ds.glm()` without additional communication.

Without site effects (as in Equation 8), the same estimators apply by replacing Z_i with $X_i' = (1, X_i^\top)^\top$ and $K + d$ with $d + 1$. Under homogeneous covariate distributions and without site effects (Equation 9), the inverse Gram matrices and mean covariates are no longer needed for the residual-variance terms. Only the pooled treatment proportion p, estimated as $\hat{p} = n^{-1} \sum_{k=1}^{K} n_k^{(1)}$ is required.

This represents a novel contribution beyond (Khellaf et al. 2024), and its mathematical derivations and implementation in DataSHIELD are detailed in Appendix B.

3 Package Design

The federated computation of the estimator defined in Equation 3 and described in is realized through the DataSHIELD interface using two companion R packages: `dsFedAteClient`, which runs on the analyst's machine, and `dsFedAte`, which is installed on each participating data server. This follows the standard two-package architecture imposed by `DataSHIELD`: only the server-side package has direct access to individual-level data, while the analyst communicates with the participating servers exclusively through the aggregate interface provided by the `DSI` package (Marcon 2025a).

3.1 Client package: `dsFedAteClient`

To estimate treatment effects from decentralized RCT data, the analyst only needs to call a single function, `ds.EstimateAte()`, specifying the treatment, outcome, and covariate variables, as well as the assumptions regarding covariate heterogeneity and site effects. Missing data are not currently supported. The function is supported internally by several unexported helpers: `.add_study_id`, `.check_args()`, `.check_datasources()`, `.check_glm_validity()`, `.compute_variance()`, `.localAte()`).

3.1.1 Pipeline Overview

DataSHIELD distinguishes between two types of functions: *assign* functions, which perform modifications on the server side without returning data to the analyst, and *aggregate* functions, which return privacy-safe summaries to the analyst. Using `dsFedAteClient`, the analyst issues a single call to `ds.EstimateAte()`; internally, this triggers three kinds of DataSHIELD operations against each data server:

submitted

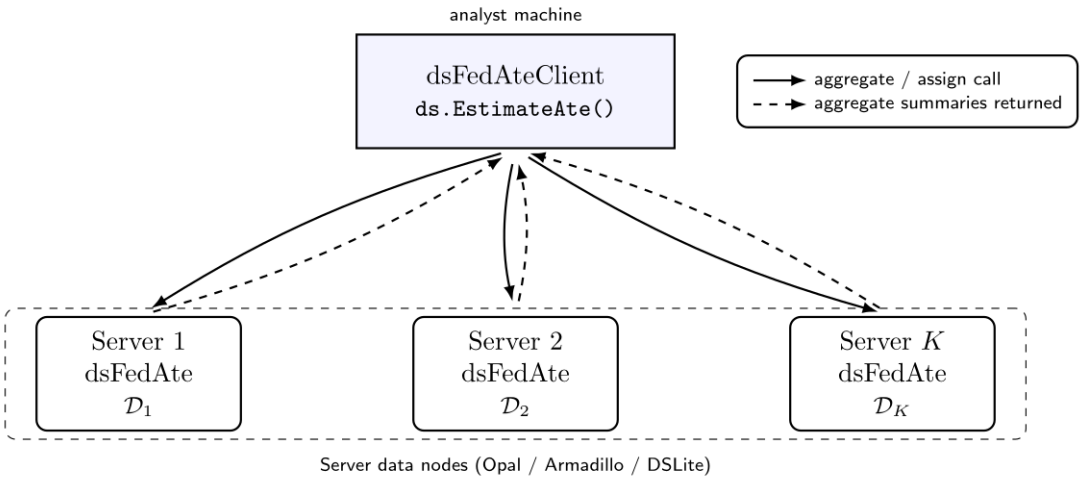

Figure 2: Two-package architecture. `dsFedAteClient` runs on the analyst's machine; `dsFedAte` is installed on each participating data server. Solid arrows are analyst-to-server queries (`DataSHIELD` aggregate or assign calls); dashed arrows are the aggregate summaries returned to the analyst.

1. one *assign* call to the server-side function `addStudyIdDS()` from the `dsFedAte` package, issued by the client-side function `.add_study_id` (only when `study_effects = TRUE`), corresponding to the outcome model defined in Equation 5. Since the regression requires a site indicator H, and each server only has access to its own site's data, this indicator is implemented locally as $K-1$ dummy columns, appended—with study 1 as the reference level—to a new server-side data frame.

2. two *assign* calls to `ds.dataFrameSubset()` from `dsBaseClient`, splitting the data into two arm-specific server-side data frames used in the subsequent federated linear regressions to compute $\hat{\mu}_0$ and $\hat{\mu}_1$.

3. two *aggregate* calls to `ds.glm()` from `dsBaseClient`, fitting the federated arm-specific linear models and returning the estimated coefficient vectors $(\hat{d}_0, \hat{\beta}_0)$ and $(\hat{d}_1, \hat{\beta}_1)$ that define $\hat{\mu}_0$ and $\hat{\mu}_1$. When `compute_variance && heterogeneous` is `TRUE`, `viewVarCov = TRUE` is additionally passed to `ds.glm()`, which also returns estimates of the matrices M_a^{-1} or $\tilde{M}^{-1}$ defined in Proposition 1, needed for the variance estimation – see the matrices G_a defined in Equation 13 in Appendix B for further details.

4. one *aggregate* call to `localAteDS()` from the `dsFedAte` package — issued by the client-side function `.localAte()` — which broadcasts $(\hat{d}_0, \hat{\beta}_0)$ and $(\hat{d}_1, \hat{\beta}_1)$ to each server and returns the site-specific estimates $\hat{\tau}_k$ along with aggregated local summaries, from which the analyst assembles the global estimate $\hat{\tau}$ and its variance $\hat{V}(\hat{\tau})$.

Both the first *assign* call and the last of these *aggregate* calls reach into the server package `dsFedAte`. The remaining steps are delegated to the reference `DataSHIELD` packages `dsBase` and `dsBaseClient`.

3.1.2 Arguments

The function `ds.EstimateAte` takes nine arguments. The first five specify the server connections (`datasources`) and the string identifiers for the data (`base_name`) and variables of interest: the outcome Y (`outcome_name`), the treatment A (`treatment_name`), and the covariates X (`covariate_names`). The four variable identifiers are required. The `datasources` argument is optional; if omitted, the function automatically searches for active connections. The remaining four arguments are Boolean

submitted

options that control variance estimation and outcome modeling assumptions. These are detailed in Table 1.

Table 1: Boolean options of `ds.EstimateAte()`.

Argument	Default	Effect
`compute_variance`	`TRUE`	If `TRUE`, returns variance and 95% CI.
`heterogeneous`	`TRUE`	If `TRUE`, computes the variance using Equation (7) or Equation (8) according to whether `study_effects` is `TRUE` or `FALSE`, passing `viewVarCov = TRUE` to `ds.glm()` and additionally requiring each server to return its column sums. If `FALSE`, computes the variance using Equation (9) — see Appendix B for further details.
`heteroscedastic_arms`	`FALSE`	If `TRUE`, σ_1^2 and σ_0^2 are assumed to differ and are estimated separately from the arm-specific residual sums of squares. If `FALSE`, the residual sums are pooled and combined after appropriate rescaling. — see Appendix B for further details.
`study_effects`	`FALSE`	If `TRUE`, adds $K-1$ integer study dummies to the design matrix on each server via the server-side function `addStudyIdDS()`, invoked by the client-side function `.add_study_id`.

3.1.3 Return Values

The result is a named list. Its shape depends on `compute_variance`, and is documented in Table 2. The vectors `n_k` and `ate_k` store the local sample sizes and local average treatment effects, respectively.

Table 2: Return value of `ds.EstimateAte()`.

Element	Type	Present when
`ate`	scalar	Always
`n_total`	scalar	Always
`n_k`	vector	Always
`ate_k`	vector	Always
`variance`	scalar	`compute_variance = TRUE`
`ci_lower`	scalar	`compute_variance = TRUE`
`ci_upper`	scalar	`compute_variance = TRUE`

3.2 Server package: `dsFedAte`

The server package exports exactly two functions, both invoked by the client (analyst) via internal calls within `ds.EstimateAte()`: an *aggregate* function `localAteDS()`, called via `.localAte`, and an *assign* function `addStudyIdDS()`, called via `.add_study_id`.

3.2.1 `localAteDS()`.

This is the core component of the server package. It is called once per server, after `ds.glm` has converged, with the pooled coefficients $(\hat{d}_a, \hat{\beta}_a)$ broadcast back to it. The function then:

1. Computes the local average treatment effect $\hat{\tau}_k$ as defined in Equation 2.

submitted

2. If `compute_variance = TRUE`, also computes the values S_k defined in Equation 10.

3. If both `compute_variance = TRUE` and `heterogeneous = TRUE`, it also returns the column sums $\sum_i (1, X_i^\top)^\top$ of the augmented design matrix, allowing the client to reconstruct the global covariate mean $\bar{X}$ required for the variance estimator in Equation 8 (or its study-augmented counterpart required for Equation 7) – for further details on the federated computation of the variance see Appendix B.

The returned object is a list. Table 3 lists its elements and the conditions under which each is present. The key privacy observation is that the function never returns any vector of length n_k: the server output is a scalar or a vector of length $d + 1$. The disclosure risk therefore scales with the covariate dimension rather than with the local sample size.

Table 3: Elements returned by `localAteDS()`. The number of returned scalars per site is bounded by $3 + d + 2$, independently of n_k.

Element	Description	Returned when
`n_k`	Local sample size n_k	Always
`n_k_1`	Local treated count $n_k^{(1)}$	Always
`tau_k`	Local ATE $\hat{\tau}_k$ as in Equation 2	Always
`s_k`	The values S_k as in Equation 10.	`compute_variance = TRUE`
`g_raw_k`	Column sums of $(1, X)$: length $d + 1$	`compute_variance` and `heterogeneous` both `TRUE`

3.2.2 `addStudyIdDS()`.

This assign function takes a data frame name, this server's 1-based study index k, and the total study count K, and appends $K - 1$ integer dummy columns `study_id2`, ..., `study_id`K that encode study membership (study 1 is the reference; on server $k > 1$, the k-th dummy equals 1 and the rest are 0). The augmented data frame is returned and stored by the client under a fresh symbol name via `DSI::datashield.assign()`.

Remark. The dummies are *integer* columns, not R factors. Locally, every server sees only one value of the study identifier — its own — so a factor column would have a single level, and `model.frame()` inside `ds.glm()` would fail with *"contrasts can be applied only to factors with 2 or more levels"*.

3.3 Data Flow Summary

Figure 3 shows a schema of a single call to `ds.EstimateAte()` in its default configuration (`compute_variance = TRUE`, `heterogeneous = TRUE`, `study_effects = FALSE`). For readability we show only the client and one server.

Two features of the flow are worth emphasising:

1. **Five server calls, independent of K and n.** `ds.EstimateAte()` issues two assigns to split each server's data frame by treatment arm (`ds.dataFrameSubset`), two aggregates to fit the per-arm federated OLS (`ds.glm`, which `dsBaseClient` implements as a federated IRLS – see Appendix D), and one aggregate to collect per-site plug-in pseudo-outcome scalars (`localAteDS`); each top-level call may itself expand into several network round-trips inside `dsBaseClient` or `DSI` for validation and disclosure-risk reporting. The two `ds.glm` fits and the final `localAteDS` are the three aggregates that carry estimator-relevant payloads. Enabling `study_effects = TRUE` adds one per-site assign (`addStudyIdDS`). The count depends only on this fixed structure, not on K, n, or d.

submitted

2. **Communication cost scales with $O(d^2)$, not $O(n)$.** The communication cost is dominated by the $(d+1)\times(d+1)$ the matrices G_a defined in Equation 13, one per arm, which are transmitted only when `heterogeneous = TRUE`, together with $O(d)$ column sums per site. All remaining quantities are scalars. Crucially, the local sample sizes n_k influence only numerical precision and have no other impact on bandwidth, so disclosure risk is determined by the covariate dimension d rather than by the number of observations. In the special cases where `compute_variance = FALSE`, or where `compute_variance = TRUE` and `heterogeneous = FALSE`, the protocol only requires scalar exchanges.

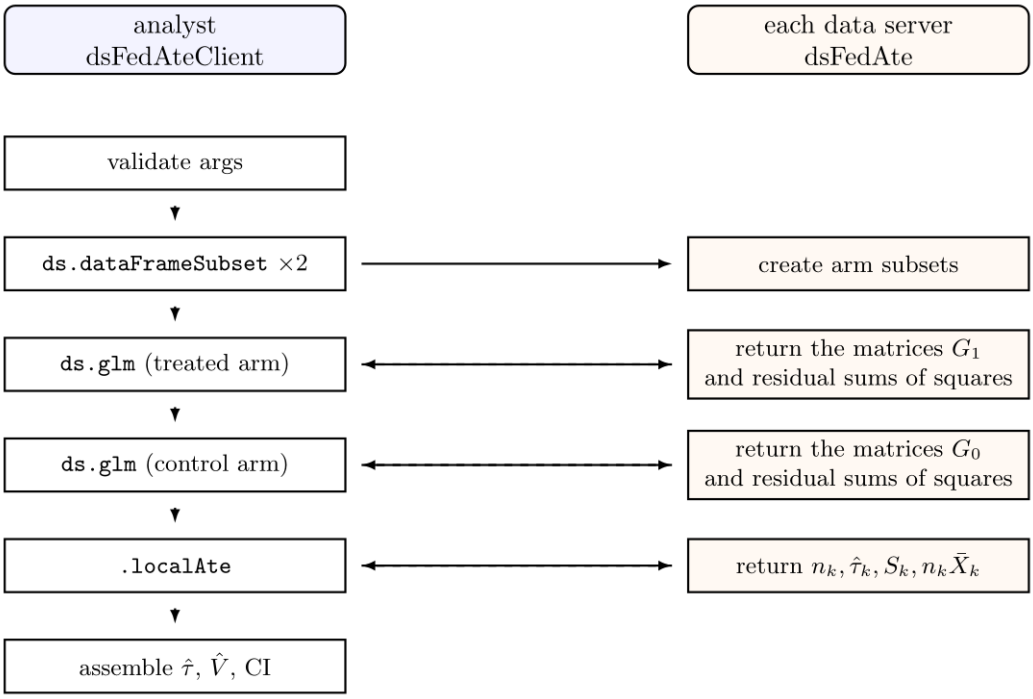

Figure 3: Data flow of a single call to `ds.EstimateAte()` with `compute_variance = TRUE` and `heterogeneous = TRUE`. There are five top-level DataSHIELD calls: two `ds.dataFrameSubset` assigns, two `ds.glm` aggregates (returning coefficients, residual sums as in Equation 14, and the matrices G_a as defined in Equation 13), and one `.localAte` aggregate (returning per-site scalars and a length-$(d+1)$ column-sum vector). Individual outcomes and covariate values never leave the servers.

4 Tutorial Examples

This section walks through five worked examples of federated Average Treatment Effect (ATE) estimation using `dsFedAteClient`. All examples use **DSLite** to emulate the federated infrastructure locally — no institutional server access is required. In a real deployment, each site must instead run a DataSHIELD-compatible server, typically either Opal (OBiBa 2024) (the reference implementation) or the lightweight Armadillo (Cadman et al. 2025); see DataSHIELD (2024) for installation and operational guidelines.

From the analyst's perspective, estimating the ATE from decentralised RCT data requires a single function call — `ds.EstimateAte()`. Its arguments are the connections (`datasources`), the server-side data-frame name (`base_name`), the outcome, treatment, and covariate column names (`outcome_name`, `treatment_name`, `covariate_names`), and the model switches `heterogeneous`, `study_effects`, and `heteroscedastic_arms`. The data at each site must share the same data-frame name and column names, and missing values are not supported.

SUBMITTED

Following the simulation settings of Appendices C.1–C.4 of Khellaf et al. (2025), Section 4.3 through Section 4.6 cover progressively richer settings: homogeneous covariate distributions; DataSHIELD's privacy controls in action, where treatment imbalance triggers the `nfilter.glm` disclosure threshold; heterogeneous covariate distributions; and study-level confounding. Each example closes with a 2000-replication coverage check; the three results are collected in Section 4.8, confirming that the variance formulas in Proposition 1 yield confidence intervals with nominal 95% coverage across these settings.

4.1 Shared parameters

All examples share the following structural parameters. The intercepts and slopes specified below generate the potential outcomes $Y^{(0)}$ and $Y^{(1)}$ for each individual.

4.2 Helpers

Six helpers — `generate_data()`, `centralised_ate()`, `centralised_variance()`, `make_connections()`, `run_coverage()`, and the pair `report_ate()` / `verify_against_centralised()` — are used throughout the examples. The first five are described briefly below; all code is folded by default, click **Show** to expand it.

- `generate_data()` generates K RCT datasets from the linear potential-outcomes model (Equation 5, or Equation 6 without study effects), covering three options: heterogeneous/homogeneous covariate distributions, presence/absence of study effects (d_k), and heteroscedastic/homoscedastic arms corresponding to $\sigma_0 \neq \sigma_1$ and $\sigma_0 = \sigma_1$.
- `centralised_ate()` computes the plug-in G-formula estimator Equation 3 in a centralised fashion, i.e., with access to the full dataset (not partitioned among servers), with the option to include study effects.
- `centralised_variance()` computes the variance according to Proposition 1 in a centralised fashion, i.e., with access to the full dataset (not partitioned among servers), covering three options: heterogeneous/homogeneous covariate distributions, presence/absence of study effects, and heteroscedastic/homoscedastic arms corresponding to $\sigma_0 \neq \sigma_1$ and $\sigma_0 = \sigma_1$.
- `make_connections()` creates a mock DSLite federation with one in-memory R object per site. The one thing worth flagging: `dsFedAte` methods must be registered manually on each mock server, since DSLite only auto-discovers `dsBase`.
- `run_coverage()` repeatedly simulates data (via `generate_data()`), fits the centralised estimator, and checks whether the true ATE lies inside the 95% delta-method confidence interval. Returns the empirical coverage over `n_rep` replications.

4.3 Example 1 — Homogeneous populations

Five sites with balanced sample sizes ($n_k = 200$), sharing the same covariate distribution — that is, $X \mid H = k \sim \mathcal{N}(\mu_X, \Sigma_X)$ for all $k \in [K]$ — and no site effects (`study_effects = FALSE`). Treatment probabilities differ across sites, $p_k = (0.65, 0.65, 0.65, 0.35, 0.35)$, but remain sufficiently far away from 0 and 1, so the GLM disclosure filter is satisfied. The outcome follows model Equation 6 with homoscedastic arms (`heteroscedastic_arms = FALSE`), i.e., $\sigma_0 = \sigma_1 = 1$. Each site is assumed to recruit from the same population (`heterogeneous = FALSE`); in the DAG of Figure 1, this corresponds to removing both the dashed arc ($H \to Y$) and the dotted arc ($H \to X$).

This section verifies that the federated estimator exactly matches its centralised counterpart, and similarly for the variances. It then demonstrates that the variance formula in Equation 9 achieves

submitted

nominal coverage across 2000 fresh replications of this setting.

Generate data.

Connect and estimate.

Results.

```
#> Federated   ATE: -1.1694  bias: -0.0694  SE: 0.0886  95% CI: [-1.3431, -0.9958]
```

We verify numerical equality with the centralised estimate:

```
#> Centralised ATE: -1.1694  SE: 0.0886
#> ATE equal:      TRUE
#> Var equal:      TRUE
```

Coverage. We check that the 95% CI achieves nominal coverage over 2000 fresh replications of this setting:

```
#> Empirical coverage: 94.9%
```

4.4 Example 2 — Disclosure thresholds and privacy levels

Same model parameters as Example 1, but with extreme treatment probabilities $p_k = (0.9, 0.9, 0.1, 0.1)$. With $n_k = 200$ and $p = 0.1$, the minority arm contains only 20 observations in expectation. This example illustrates two independent DataSHIELD disclosure controls — an insufficiently low disclosure threshold and an `"avocado"` privacy level — each triggered by a different operator setting, that can prevent estimation entirely.

Generate data.

Guard 1 — GLM disclosure threshold (`nfilter.glm` = 0.33) — estimation blocked.

`ds.EstimateAte` subsets data into arm-specific objects via `dataFrameSubsetDS2`, then fits an arm-specific OLS via `glmDS2`. The latter enforces $d + 1 \leq \texttt{nfilter.glm} \times n_k^{(a)}$. Each site has $n_k = 200$ observations and the minority arm is assigned with probability $p = 0.1$, giving an expected minority arm size of 20. With $d + 1 = 11$ parameters, the default threshold 0.33 blocks the call because $11 > 0.33 \times 20 = 6.6$.

```
#>
#>
#> MODEL FITTING TERMINATED AT FIRST ITERATION:
#> Any values of 1 in the following tables denote potential disclosure risks
#> please use the argument <datasources> to include only valid studies.
#> Errors by study are as follows:
#> 00000
#> 0000000000000000000000000000000000000000000000000000000000
#> 00000
#> 00000
#> 00011
#> No errorsNo errorsNo errorsSTUDY DATA OR APPLIED MODEL INVALID FOR THIS SOURCESTUDY DATA OR APPL
#>
#>
#> MODEL FITTING TERMINATED AT FIRST ITERATION:
#> Any values of 1 in the following tables denote potential disclosure risks
#> please use the argument <datasources> to include only valid studies.
```

submitted

```
#> Errors by study are as follows:
#> 00000
#> 0000000000000000000000000000000000000000000000000000000000
#> 00000
#> 00000
#> 11100
#> STUDY DATA OR APPLIED MODEL INVALID FOR THIS SOURCESTUDY DATA OR APPLIED MODEL INVALID FOR THIS
#> Error in `.check_glm_validity()` at FranciscoAndrade90-dsFedAteClient-5d3d720/R/ds.EstimateAte.
#> x DataSHIELD blocked the GLM in 5 study/studies: study4, study5, study1,
#>   study2, study3.
#> i At least one treatment subset in those studies has too few observations
#>   relative to the 11 model parameters.
#> i DataSHIELD's nfilter.glm threshold requires each arm to have at least
#>   n_params / nfilter.glm observations (default: 11 / 0.33 = 34).
#> > Consider reducing the number of covariates, or exclude the affected studies
#>   via the `datasources` argument.
```

Guard 1 resolved — raised threshold (`nfilter.glm` = 0.6) — estimation succeeds.

Raising `nfilter.glm` to $0.6 > 11/20 = 0.55$ satisfies the threshold. The estimate comes through with a larger standard error than Example 1 because extreme arm imbalance reduces effective sample size.

```
#> Federated   ATE: -1.1582  bias: -0.0582  SE: 0.0919  95% CI: [-1.3384, -0.9780]
```

Guard 2 — Privacy level `"avocado"` — subsetting blocked.

The `"avocado"` level blocks known subsetting attacks, so `dataFrameSubsetDS2` is not permitted under it. Because subsetting occurs *before* (see Figure 3) both the GLM and the `localAteDS` stages, the error originates from `dataFrameSubsetDS2`. Raising `nfilter.glm` does not help since the GLM stage is never reached.

```
#> Error:
#> ! There are some DataSHIELD errors, list them with datashield.errors()
```

Switching back to `"banana"` with `nfilter.glm` = 0.6 — as in the block above — restores estimation.

4.5 Example 3 — Heterogeneous covariate distributions

Five sites with unbalanced sample sizes ($n_1 = 600$, $n_2 = \ldots = n_5 = 120$), each recruiting from a distinct patient population: the covariate mean μ_k and covariance Σ_k vary across sites, so $X \mid H = k \sim \mathcal{N}(\mu_k, \Sigma_k)$. Treatment probabilities are the same as Example 1, $p_k = (0.65, 0.65, 0.65, 0.35, 0.35)$; even the smallest sites have an expected minority arm size of $0.35 \times 120 = 42 > 33$, so the default `"banana"` settings apply. The outcome follows model Equation 6 with no site effects (`study_effects = FALSE`) and homoscedastic arms (`heteroscedastic_arms = FALSE`). Each site recruits from a different population (`heterogeneous = TRUE`); in the DAG of Figure 1, this corresponds to removing the dashed arc ($H \to Y$) while keeping the dotted arc ($H \to X$).

Under distributional shift the homogeneous variance formula is no longer valid; the formula in Equation 8 must be used instead. Similar to Example 1, this section verifies that the federated estimator matches its centralised counterpart and that the heterogeneous variance formula achieves nominal 95% coverage across 2000 fresh replications.

Define site-specific distributions.

Generate data.

submitted

Connect and estimate.

Results.

```
#> Federated   ATE: -0.1953  bias: +0.1130  SE: 0.1307  95% CI: [-0.4516, 0.0609]
#> Centralised ATE: -0.1953  SE: 0.1307
#> ATE equal:      TRUE
#> Var equal:      TRUE
```

Coverage. We check that the 95% CI achieves nominal coverage over 2000 fresh replications of this setting:

```
#> Empirical coverage: 96.6%
```

4.6 Example 4 — Study effects

Five sites with unbalanced sample sizes ($n_1 = 600$, $n_2 = \ldots = n_5 = 300$), using the same site-specific covariate distributions as Example 3. Treatment probabilities are the same as Example 1, $p_k = (0.65, 0.65, 0.65, 0.35, 0.35)$; the smallest sites have an expected minority arm size of $0.35 \times 300 = 105 \gg 33$, so the default `"banana"` settings apply. The outcome follows model Equation 5 with site effects $d_k = (1, 2, -1, 30, 2)$ (`study_effects = TRUE`) and heteroscedastic arms $\sigma_0 = 0.5$, $\sigma_1 = 3.0$ (`heteroscedastic_arms = TRUE`). Each site recruits from a different population (`heterogeneous = TRUE`); in the DAG of Figure 1, both arcs are present: the dashed arc ($H \to Y$, via d_k) and the dotted arc ($H \to X$).

Because d_k enters both potential outcomes equally it does not change the true ATE — the shift cancels in $Y^{(1)} - Y^{(0)}$ — but when a site with a large shift ($d_4 = 30$) also has lower treatment probability ($p_4 = 0.35$), the unadjusted estimator conflates the site shift with the treatment effect. Augmenting the design matrix with $K - 1$ site-membership dummies absorbs the shifts (`study_effects = TRUE`). This section shows the bias of the unadjusted estimator, verifies that the adjusted estimator recovers the true ATE, and confirms nominal 95% coverage across 2000 fresh replications.

Generate data.

Connect.

Unadjusted — biased.

With `study_effects = FALSE`, the design matrix contains only the d covariates X; the site-intercept shifts d_k are omitted from the regression. This leads to a biased estimator.

```
#> Unadjusted ATE: -4.3609  bias: -4.4484  (true ATE: 0.0875)
```

Adjusted — unbiased.

```
#>   Adjusted ATE: 0.0556  bias: -0.0319  SE: 0.1472  95% CI: [-0.2329, 0.3441]
```

The adjusted estimate recovers the true ATE and the 95% CI covers it.

Verify equality with centralised adjusted analysis.

```
#> Centralised ATE: 0.0556  SE: 0.1472
#> ATE equal:      TRUE
#> Var equal:      TRUE
```

Coverage. We check that the 95% CI achieves nominal coverage over 2000 fresh replications of this setting:

submitted

```
#> Empirical coverage: 95.5%
```

4.7 Example 5 — Consistency under model misspecification

Examples 1–4 all assumed a linear outcome model, which matched the data-generating process by construction. This final example demonstrates a stronger result: the plug-in G-formula estimator obtained by regressing on X and H remains consistent for the ATE even when the true outcome model is nonlinear—the content of Proposition 2.

We retain the covariate and treatment-assignment structure of Example 3 — heterogeneous covariates and treatment probabilities $p_k = (0.65, 0.65, 0.65, 0.35, 0.35)$ — but now generate a binary outcome via a logistic model,

$$Y^a \mid X = x,\ H = k \ \sim\ \text{Bernoulli}\big(\sigma(c_a + x^\top \beta_a)\big), \tag{11}$$

so the linear G-formula is now misspecified. The outcome-model coefficients (c_a, β_a) used here are rescaled from the shared parameters so that the linear predictor stays in the sensitive range of the logistic link and the resulting ATE is of practical magnitude:

Although the data generating process has no site effects, `study_effects = TRUE` must still be passed to `centralised_ate()`, or, in a federated setting, to `ds.EstimateAte()`.

True ATE. Computed by Monte Carlo on a very large draw:

```
#> Monte Carlo true ATE: 0.13838
```

Simulation. For each total sample size n, we approximate the MSE by averaging the squared bias over 2000 Monte Carlo replications. Each replication generates a dataset, overwrites each site's `Y` with Bernoulli draws from the logistic model, and fits the linear G-formula with `study_effects = TRUE`:

MSE vs n. On a log-log scale the empirical MSE falls along a straight line of slope -1, i.e., MSE $\propto n^{-1}$, as expected for a $\sqrt{n}$-consistent estimator. Vertical bars are 95% normal confidence intervals around each MSE:

submitted

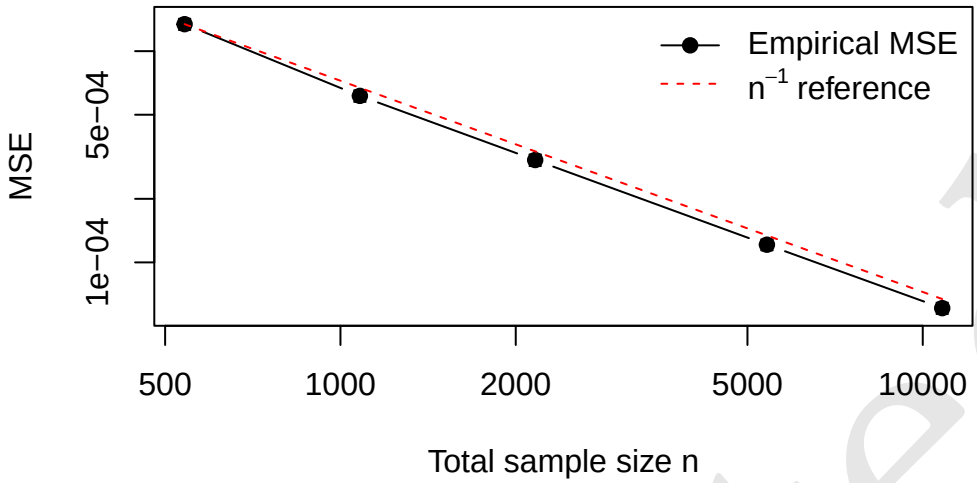

Figure 4: Empirical MSE of the federated ATE estimator as a function of total sample size (log-log scale), under a logistic data-generating process. Vertical bars are 95% normal confidence intervals around each MSE; the dashed line is the n^{-1} reference anchored at the smallest n.

Show bias–variance decomposition

The MSE above decomposes as MSE = Bias^2 + Var. The two panels below plot the components of that decomposition: on the left, the empirical absolute bias $|\bar{\hat{\tau}} - \tau|$, whose magnitude stays below the Monte Carlo resolution at every n — consistent with the estimator's finite-sample bias being far smaller than its variance, as expected for a consistent estimator under misspecification — and on the right, the empirical variance, which tracks the n^{-1} reference and dominates the MSE at every sample size.

submitted

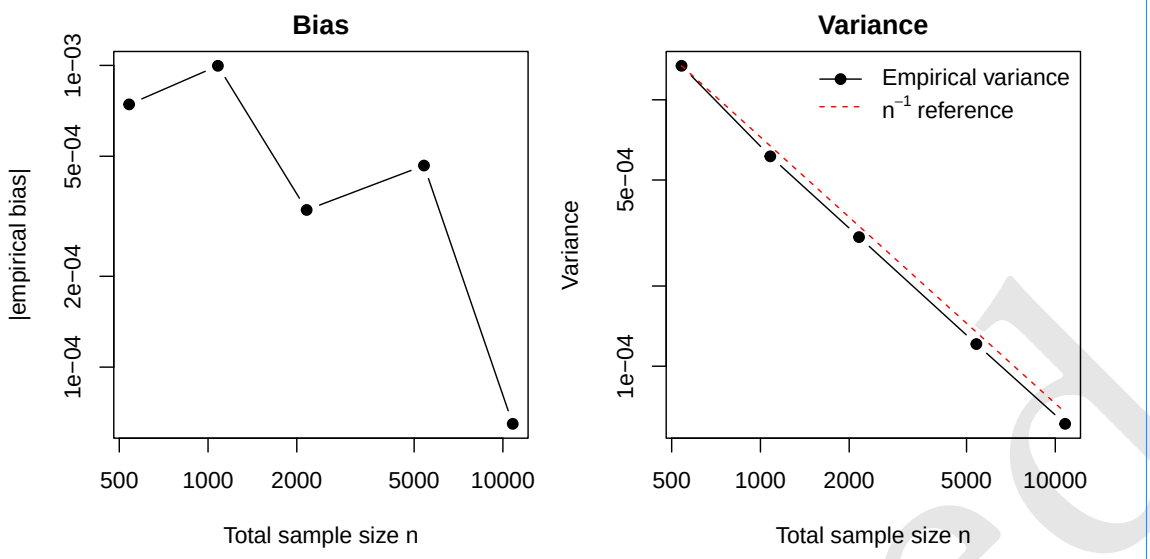

Figure 5: Monte Carlo decomposition of the MSE above. Left: absolute empirical bias $|\bar{\hat{\tau}} - \tau|$ versus total sample size (log-log). Right: empirical variance versus total sample size, with the n^{-1} reference anchored at the smallest n.

4.8 Coverage summary

The three per-example coverage checks reported in Section 4.3, Section 4.5, and Section 4.6 are collected below. All sit near the nominal 95%, confirming that the variance in Proposition 1 yields well-calibrated intervals across the simulation settings considered here.

Table 4: Empirical 95% CI coverage over 2000 replications per setting.

Setting	Coverage
Homogeneous (Example 1)	94.9%
Heterogeneous covariates (Example 3)	96.6%
Study effects + heteroscedastic arms (Example 4)	95.5%

5 Discussion

In this article, we present a federated implementation of ATE estimation across multiple RCTs to facilitate its practical adoption.

An important extension is the application of the proposed framework to multi-site observational studies. In this setting, identification requires conditional exchangeability, $(Y^1, Y^0) \perp\!\!\!\perp A \mid X, H$, rather than randomization. The current G-formula estimator remains consistent under correctly specified linear outcome models, but the robustness to model misspecification established in Proposition 2 no longer holds. Extensions of the framework thus include flexible federated estimation of outcome models and propensity scores to enable doubly robust estimators such as augmented inverse probability weighting (AIPW). Currently, DataSHIELD's `dsBase` primarily supports parametric propensity-score estimation through logistic regression via `ds.glm`.

Further extensions include incorporating external control arms (sites with $p_k = 0$), which violate site-level positivity and require additional identification assumptions, and generalizing treatment

submitted

effects to external target populations through federated density-ratio estimation under appropriate transportability and overlap assumptions.

References

Barratt, Alexandra, Les Irwig, Paul Glasziou, et al. 1999. "Users' Guides to the Medical Literature: XVII. How to Use Guidelines and Recommendations about Screening." *Jama* 281 (21): 2029–34.

Burke, Danielle L, Joie Ensor, and Richard D Riley. 2017. "Meta-Analysis Using Individual Participant Data: One-Stage and Two-Stage Approaches, and Why They May Differ." *Statistics in Medicine* 36 (5): 855–75.

Cadman, Tim, Mariska K Slofstra, Marije A van der Geest, et al. 2025. "MOLGENIS Armadillo: A Lightweight Server for Federated Analysis Using DataSHIELD." *Bioinformatics* 41 (1): btae726. https://doi.org/10.1093/bioinformatics/btae726.

DataSHIELD. 2024. *For Data Custodians*. Https://datashield.org/for-data-custodians/. https://datashield.org/for-data-custodians/.

DataSHIELD. n.d. *Disclosure Control*. Https://wiki.datashield.org/en/opmanag/disclosure-control.

Evans, David. 2003. "Hierarchy of Evidence: A Framework for Ranking Evidence Evaluating Healthcare Interventions." *Journal of Clinical Nursing* 12 (1): 77–84.

French Health Authority. 2024. *Pricing & Reimbursement of Drugs and HTA Policies in France*. https://www.has-sante.fr/upload/docs/application/pdf/2014-03/pricing_reimbursement_of_drugs_and_hta_policies_in_france.pdf.

Gaye, Amadou, Yannick Marcon, Julia Isaeva, et al. 2014. "DataSHIELD: Taking the Analysis to the Data, Not the Data to the Analysis." *International Journal of Epidemiology* 43 (6): 1929–44.

Gusinow, Roy, Andrei Scott Morgan, Lorenzo Maria Canziani, et al. 2026. "Privacy-Preserving Matching for Federated Causal Inference in Multicentre Patient Cohorts." *medRxiv*, 2026–07.

He, Chaoyang, Songze Li, Jinhyun So, et al. 2020. "Fedml: A Research Library and Benchmark for Federated Machine Learning." *arXiv Preprint arXiv:2007.13518*.

Kairouz, Peter, and H Brendan McMahan. 2021. "Advances and Open Problems in Federated Learning." *Foundations and Trends in Machine Learning* 14 (1-2): 1–210.

Khellaf, Rémi, Aurélien Bellet, and Julie Josse. 2024. "Federated Causal Inference: Multi-Study Ate Estimation Beyond Meta-Analysis." *arXiv Preprint arXiv:2410.16870*.

Khellaf, Rémi, Aurélien Bellet, and Julie Josse. 2025. "Federated Causal Inference from Multi-Site Observational Data via Propensity Score Aggregation." *arXiv Preprint arXiv:2505.17961*.

Lin, Winston. 2013. "Agnostic Notes on Regression Adjustments to Experimental Data: Reexamining Freedman's Critique." *The Annals of Applied Statistics*, 295–318.

Liu, Hanzhong, and Yuehan Yang. 2020. "Regression-Adjusted Average Treatment Effect Estimates

submitted

in Stratified Randomized Experiments." *Biometrika* 107 (4): 935–48.

Marcon, Yannick. 2025a. *DSI: 'DataSHIELD' Interface*. https://doi.org/10.32614/CRAN.package.DSI.

Marcon, Yannick. 2025b. *DSLite: 'DataSHIELD' Implementation on Local Datasets*. https://doi.org/10.32614/CRAN.package.DSLite.

McMahan, Brendan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. "Communication-Efficient Learning of Deep Networks from Decentralized Data." *Artificial Intelligence and Statistics*, 1273–82.

OBiBa. 2024. *OBiBa Opal Documentation*. Https://opaldoc.obiba.org/en/latest/. https://opaldoc.obiba.org/en/latest/.

Riley, Richard D, and Ewout W Steyerberg. 2010. "Meta-Analysis of a Binary Outcome Using Individual Participant Data and Aggregate Data." *Research Synthesis Methods* 1 (1): 2–19.

Rothwell, Peter M. 2005. "External Validity of Randomised Controlled Trials:'to Whom Do the Results of This Trial Apply?"' *The Lancet* 365 (9453): 82–93.

Rubin, Donald B. 1974. "Estimating Causal Effects of Treatments in Randomized and Nonrandomized Studies." *Journal of Educational Psychology* 66 (5): 688–701.

Wager, Stefan. 2024. "Causal Inference: A Statistical Learning Approach." *Online Book*.

Wilson, Rebecca C, Oliver W Butters, Demetris Avraam, et al. 2017. "DataSHIELD–New Directions and Dimensions." *Data Science Journal* 16: 21–21.

Wolfson, Michael, Susan E Wallace, Nicholas Masca, et al. 2010. "DataSHIELD: Resolving a Conflict in Contemporary Bioscience—Performing a Pooled Analysis of Individual-Level Data Without Sharing the Data." *International Journal of Epidemiology* 39 (5): 1372–82.

Xiong, Ruoxuan, Allison Koenecke, Michael Powell, Zhu Shen, Joshua T Vogelstein, and Susan Athey. 2023. "Federated Causal Inference in Heterogeneous Observational Data." *Statistics in Medicine* 42 (24): 4418–39.

Ziller, Alexander, Andrew Trask, Antonio Lopardo, et al. 2021. "Pysyft: A Library for Easy Federated Learning." In *Federated Learning Systems: Towards Next-Generation AI*. Springer.

Appendix A: Variance

Proof of Proposition 1

Proof. Let $Z = (H^\top, X^\top)^\top$ and $\theta_a = (d_a^\top, \beta_a^\top)^\top$. We prove that under Model Equation 5, $\hat{\tau}$ satisfies $\sqrt{n}(\hat{\tau} - \tau) \xrightarrow{d} \mathcal{N}(0, V)$, with the general heteroscedastic asymptotic variance:

$$V = \mathbb{E}[Z]^\top M_1^{-1} \Omega_1 M_1^{-1} \mathbb{E}[Z] + \mathbb{E}[Z]^\top M_0^{-1} \Omega_0 M_0^{-1} \mathbb{E}[Z] + \mathbb{V}(Z^\top \Delta),$$

where $\Delta = \theta_1 - \theta_0$, $M_a = \mathbb{E}[\mathbb{I}(A = a) Z Z^\top]$, and $\Omega_a = \mathbb{E}[\mathbb{I}(A = a) \varepsilon_a^2 Z Z^\top]$.

submitted

First, remark that $\hat{\tau} = \bar{Z}^\top \hat{\Delta}$ and $\tau = \mathbb{E}[Z]^\top \Delta$, where $\bar{Z} = \frac{1}{n}\sum_{i=1}^{n} Z_i$. Thus,

$$\begin{aligned}
\sqrt{n}(\hat{\tau} - \tau) &= \sqrt{n}\left(\bar{Z}^\top \hat{\Delta} - \mathbb{E}[Z]^\top \Delta\right) \\
&= \sqrt{n}\left(\bar{Z}^\top \hat{\Delta} - \bar{Z}^\top \Delta + \bar{Z}^\top \Delta - \mathbb{E}[Z]^\top \Delta\right) \\
&= \bar{Z}^\top \sqrt{n}(\hat{\Delta} - \Delta) + \sqrt{n}(\bar{Z} - \mathbb{E}[Z])^\top \Delta \\
&= \mathbb{E}[Z]^\top \sqrt{n}(\hat{\Delta} - \Delta) + \sqrt{n}(\bar{Z} - \mathbb{E}[Z])^\top \Delta + o_p(1),
\end{aligned}$$

where the last equality follows from Slutsky's theorem since $\bar{Z} \xrightarrow{p} \mathbb{E}[Z]$.

Next, we substitute the standard asymptotic linear expansion for the OLS estimators $\hat{\theta}_a$:

$$\sqrt{n}(\hat{\theta}_a - \theta_a) = M_a^{-1} \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \mathbb{I}(A_i = a) Z_i \varepsilon_{a,i} + o_p(1).$$

Since $\hat{\Delta} - \Delta = (\hat{\theta}_1 - \theta_1) - (\hat{\theta}_0 - \theta_0)$, $\sqrt{n}(\hat{\tau} - \tau)$ rewrites as a scaled sum of independent and identically distributed terms:

$$\begin{aligned}
\sqrt{n}(\hat{\tau} - \tau) = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \Big[&\mathbb{E}[Z]^\top M_1^{-1} \mathbb{I}(A_i = 1) Z_i \varepsilon_{1,i} \\
&- \mathbb{E}[Z]^\top M_0^{-1} \mathbb{I}(A_i = 0) Z_i \varepsilon_{0,i} + (Z_i - \mathbb{E}[Z])^\top \Delta \Big] + o_p(1).
\end{aligned}$$

By the Central Limit Theorem, the asymptotic variance V is the variance of a single summand. The covariances between these three terms are zero because the treatment groups are mutually exclusive ($\mathbb{I}(A = 1)\mathbb{I}(A = 0) = 0$), and Model Equation 5 is well-specified ($\mathbb{E}[\varepsilon_a \mid Z] = 0$), making the errors uncorrelated with $(Z - \mathbb{E}[Z])^\top \Delta$.

Therefore, the variance of the sum is the sum of the individual variances:

$$\begin{aligned}
V &= \mathbb{V}\left(\mathbb{E}[Z]^\top M_1^{-1} \mathbb{I}(A = 1) Z \varepsilon_1\right) + \mathbb{V}\left(\mathbb{E}[Z]^\top M_0^{-1} \mathbb{I}(A = 0) Z \varepsilon_0\right) + \mathbb{V}\left((Z - \mathbb{E}[Z])^\top \Delta\right) \\
&= \mathbb{E}[Z]^\top M_1^{-1} \mathbb{E}[\mathbb{I}(A = 1)\varepsilon_1^2 Z Z^\top] M_1^{-1} \mathbb{E}[Z] + \mathbb{E}[Z]^\top M_0^{-1} \mathbb{E}[\mathbb{I}(A = 0)\varepsilon_0^2 Z Z^\top] M_0^{-1} \mathbb{E}[Z] + \mathbb{V}(Z^\top \Delta) \\
&= \mathbb{E}[Z]^\top M_1^{-1} \Omega_1 M_1^{-1} \mathbb{E}[Z] + \mathbb{E}[Z]^\top M_0^{-1} \Omega_0 M_0^{-1} \mathbb{E}[Z] + \mathbb{V}(Z^\top \Delta).
\end{aligned}$$

Under homoscedasticity ($\mathbb{V}(\varepsilon_a \mid Z) = \sigma_a^2$), we have $\Omega_a = \sigma_a^2 M_a$, which simplifies the first two terms to $\sigma_1^2 \mathbb{E}[Z]^\top M_1^{-1} \mathbb{E}[Z]$ and $\sigma_0^2 \mathbb{E}[Z]^\top M_0^{-1} \mathbb{E}[Z]$, recovering the expression in Proposition 1. ☒ □

Appendix B: Computing the Variance from Decentralised Data

This section explains how the variance can be estimated in a decentralised data setting. We begin with the last term, $\mathbb{V}\big((\beta_1 - \beta_0)^\top X\big)$, before addressing the two remaining terms.

The Last Term in the Variance Formula

We now focus on estimating $\mathbb{V}\big((\beta_1 - \beta_0)^\top X\big)$; the case of $\mathbb{V}(Z^\top \Delta)$ follows by an identical argument. A natural estimator is the sample variance of $z_i = (\hat{\beta}_1 - \hat{\beta}_0)^\top X_i$, given by

$$\frac{1}{n-1} \sum_{k=1}^{K} \sum_{i \in \mathcal{I}_k} \big(z_i - \bar{z}\big)^2,$$

where $\mathcal{I}_k := \{i : H_i = k\}$ denotes the set of indices belonging to site k. However, this expression is not directly amenable to decentralised computation, since the global mean $\bar{z}$ appears inside each local

submitted

sum and prevents a clean decomposition into site-level quantities. To overcome this, we note that $\bar{z} = \bar{X}^\top(\hat{\beta}_1 - \hat{\beta}_0)$, and that by definition $\bar{X}_k^\top(\hat{\beta}_1 - \hat{\beta}_0) - \bar{z} = \hat{\tau}_k - \hat{\tau}$. Adding and subtracting $\bar{X}_k^\top(\hat{\beta}_1 - \hat{\beta}_0)$ inside each squared term yields

$$\frac{1}{n-1}\sum_{k=1}^{K}\sum_{i\in\mathcal{I}_k}\left(z_i - \bar{z}\right)^2 = \frac{1}{n-1}\sum_{k=1}^{K}\sum_{i\in\mathcal{I}_k}\left(z_i - \bar{X}_k^\top(\hat{\beta}_1 - \hat{\beta}_0) + \hat{\tau}_k - \hat{\tau}\right)^2.$$

Expanding the square and grouping terms gives

$$= \frac{1}{n-1}\left[\sum_k S_k + \sum_k n_k(\hat{\tau}_k - \hat{\tau})^2 + 2\sum_{k=1}^{K}\sum_{i\in\mathcal{I}_k}\left(z_i - \bar{X}_k^\top(\hat{\beta}_1 - \hat{\beta}_0)\right)(\hat{\tau}_k - \hat{\tau})\right],$$

where we define the site-level sum of squared residuals S_k as in Equation 10. The cross-product term vanishes because the site-level residuals sum to zero:

$$\sum_{i\in\mathcal{I}_k}\left(z_i - \bar{X}_k^\top(\hat{\beta}_1 - \hat{\beta}_0)\right) = 0.$$

We therefore obtain the decomposition

$$\frac{1}{n-1}\sum_{k=1}^{K}\sum_{i\in\mathcal{I}_k}\left(z_i - \bar{z}\right)^2 = \frac{1}{n-1}\left[\sum_k S_k + \sum_k n_k(\hat{\tau}_k - \hat{\tau})^2\right]. \tag{12}$$

The key insight is that all quantities involved can be computed locally at each site and subsequently aggregated. Although $\hat{\beta}_1$ and $\hat{\beta}_0$ are global parameters, the variance is estimated after fitting the linear models independently using `ds.glm`, and is therefore only computed once the client (analyst) has broadcast the values of $\hat{\beta}_1$ and $\hat{\beta}_0$. The aggregation leading to Equation 12 can thus be carried out in the same round as the aggregation of τ_k, at the cost of each server further sending S_k.

`DataSHIELD` implementation

When `compute_variance = TRUE`, the servers compute S_k within the `localAteDS()` function – invoked on the server side by the client-side function `.localAte()` – and broadcast it alongside n_k and τ_k, which are transmitted regardless of the value of `compute_variance`. Upon receiving these values the client/analyst can thus perform the aggregation in Equation 12. Moreover, the computation of S_k is performed regardless of the values of `heterogeneous` and `heteroscedastic_arms`.

The Remaining Terms

This section describes how the quantities σ_1^2, σ_0^2, $\mathbb{E}[X']$ and M_1^{-1} appearing in the variance formula Equation 8 are estimated in a federated manner using the two packages, under `heterogeneous = TRUE`. We conclude by outlining the adjustments required to compute the variance formula Equation 7.

The Gram Matrices.

A natural estimator of M_a^{-1} is

$$\left(\frac{1}{n}\sum_{i=1}^{n} X_i'(X_i')^\top \mathbb{I}(A_i = a)\right)^{-1} = n\underbrace{\left(\sum_{i=1}^{n} X_i'(X_i')^\top \mathbb{I}(A_i = a)\right)^{-1}}_{:=G_a}, \tag{13}$$

so that only G_a and n are needed. The matrix G_a is returned directly by `ds.glm()` when the argument `viewVarCov = TRUE` is set.

Submitted

The Residual Variances.

When `heteroscedastic_arms = TRUE`, each arm has its own variance, estimated by

$$\hat{\sigma}_a^2 = \frac{1}{n^{(a)} - (d+1)} \sum_{i=1}^{n} \big(Y_i - \hat{\mu}_a(X_i)\big)^2 \mathbb{I}(A_i = a), \tag{14}$$

where $n^{(a)}$ denotes the number of observations assigned to treatment a. No extra communication is required: the list returned by `ds.glm()` contains an element `dev` that is exactly the sum appearing in Equation 14. When `heteroscedastic_arms = FALSE`, a single variance is shared by both arms and is estimated by

$$\hat{\sigma}^2 = \frac{1}{n - 2(d+1)} \sum_{a=0}^{1} \sum_{i=1}^{n} \big(Y_i - \hat{\mu}_a(X_i)\big)^2 \mathbb{I}(A_i = a). \tag{15}$$

The Covariate Mean.

Each server k transmits is local the covariate sum $n_k \bar{X}_k$, and the client aggregates these according into

$$\frac{1}{n} \sum_k n_k \bar{X}_k. \tag{16}$$

In the `dsFedAte` package, $\bar{X}_k$ is computed by `localAteDS()` whenever `heterogeneous = TRUE` and sent back to the client through the client-side function `.localAte()`, which is what invokes `localAteDS()` on the servers.

Study Effects

The variance formula in Equation 7 follows the same reasoning. It amounts to substituting X_i' in Equation 13 with $Z_i = (H_i^T, X_i^T)$, where H_i is one-hot encoded, replacing $\hat{\mu}_a(X_i)$ in Equation 14 and Equation 15 with $\hat{\mu}_a(X_i, H_i)$, and replacing $\bar{X}_k$ in Equation 16 with $\bar{Z}_k$.

Homogeneous Covariates

The estimator in Equation 9 admits a straightforward federated implementation. Estimation of σ_a^2 follows the procedure described above, leaving p as the only remaining quantity to be estimated. A natural estimator is

$$\hat{p} := \frac{1}{n} \sum_{k=1}^{K} n_k^{(1)},$$

where $n_k^{(1)}$ denotes the local treated count, reported to the client upon invocation of `localAteDS()`.

Appendix C: Misspecified linear G-formula asymptotic normality

Proof. For each center k and treatment arm $w \in \{0, 1\}$, define the index set $\mathscr{C}_k^{(w)} = \{i : H_i = k, W_i = w\}$ with cardinality $n_k^{(w)} = |\mathscr{C}_k^{(w)}|$. Let $n_k = n_k^{(0)} + n_k^{(1)}$ denote the total number of individuals in center k, and let $\hat{p}_k = n_k^{(1)}/n_k$ be the observed treatment proportion within that center. For any variable $Z \in \{X, Y\}$, write $\bar{Z}_k^{(w)}$ for the sample mean of Z over $\mathscr{C}_k^{(w)}$, and let $\bar{X}_k$ denote the marginal sample mean of X over all individuals in center k.

submitted

First, as stated in Lemma 1, $\hat{\tau}$, defined in Equation 3, can be expressed as

$$\hat{\tau} = \sum_{k=1}^{K} \frac{n_k}{n} \Big\{ \bar{Y}_k^{(1)} - \bar{Y}_k^{(0)} - \big(\bar{X}_k^{(1)} - \bar{X}_k^{(0)}\big)^\top \hat{\bar{\beta}}_k \Big\}, \tag{17}$$

where $\hat{\bar{\beta}}_k := (1 - \hat{p}_k)\hat{\beta}^{(1)} + \hat{p}_k \hat{\beta}^{(0)}$.

Let $(d_w^\star, \beta_w^\star)$, the population counterpart of Equation 4, such that

$$(d_w^\star, \beta_w^\star) = \operatorname*{argmin}_{d,\,\beta} \mathbb{E}\Big[\big(Y - d^\top H - \beta^\top X\big)^2 \,\Big|\, W = w \Big], \tag{18}$$

and $\bar{\beta}_k = (1 - p_k)\beta_1^\star + p_k \beta_0^\star$. Then, we have

$$\hat{\tau} - \tau = \underbrace{\sum_{k=1}^{K} \frac{n_k}{n} \Big\{ \big(\bar{X}_k^{(1)} - \bar{X}_k^{(0)}\big)^\top \big(\bar{\beta}_k - \hat{\bar{\beta}}_k\big) \Big\}}_{=: A_n} + \underbrace{\sum_{k=1}^{K} \frac{n_k}{n} \Big\{ \bar{Y}_k^{(1)} - \bar{Y}_k^{(0)} - \big(\bar{X}_k^{(1)} - \bar{X}_k^{(0)}\big)^\top \bar{\beta}_k \Big\}}_{=: B_n} - \tau. \tag{19}$$

Similar to Theorem 1.3 of Wager (2024), the Huber–White analysis of misspecified linear regression implies that $\hat{\beta}^{(w)} \xrightarrow{p} \beta_w^\star$, provided the design matrix for $(H^\top, X^\top)^\top$ is non-singular. Furthermore, by the Weak Law of Large Numbers, $\hat{p}_k \xrightarrow{p} p_k$. By the Continuous Mapping Theorem, their convex combination converges, meaning $\hat{\bar{\beta}}_k \xrightarrow{p} \bar{\beta}_k$, and thus $(\bar{\beta}_k - \hat{\bar{\beta}}_k) = o_P(1)$.

Furthermore, under within-center randomization ($X \perp\!\!\!\perp A \mid H$), covariates are exchangeable by treatment groups, so $\bar{X}_k^{(1)}$ and $\bar{X}_k^{(0)}$ both converge to $\mathbb{E}[X \mid H = k]$. By the Central Limit Theorem, their difference $\sqrt{n_k}\,(\bar{X}_k^{(1)} - \bar{X}_k^{(0)}) \to \mathcal{N}(0, \Sigma_k)$. Because $n_k/n \xrightarrow{p} \mathbb{P}(H = k) > 0$, the global scaled difference $\sqrt{n}\,(\bar{X}_k^{(1)} - \bar{X}_k^{(0)}) = O_P(1)$. Therefore, $\sqrt{n} A_n = O_P(1) \cdot o_P(1) = o_P(1)$.

Finally, remark that B_n in Equation 19 has the structure of a **stratified estimator** of the form studied in Theorem 2.1 of Wager (2024), applied to a modified outcome. Specifically, the transformed potential outcomes and their observed counterpart are defined as

$$\tilde{Y}(w) = Y(w) - X^\top \bar{\beta}_H, \qquad \tilde{Y}_i(W_i) = Y_i - X_i^\top \bar{\beta}_{H_i}, \tag{20}$$

and let $\bar{\tilde{Y}}_k^{(w)} = \frac{1}{n_k} \sum_{i \in \mathcal{C}_k^{(w)}} \tilde{Y}_i(w)$. With this notation, we have

$$B_n - \tau = \sum_{k=1}^{K} \frac{n_k}{n} \big(\bar{\tilde{Y}}_k^{(1)} - \bar{\tilde{Y}}_k^{(0)} \big) - \tau.$$

Theorem 2.1 of Wager (2024) applies directly because the transformed variables $\tilde{Y}_i(w)$ satisfy the requisite conditions of SUTVA, bounded second moments, and $W \perp\!\!\!\perp (\tilde{Y}(0), \tilde{Y}(1)) \mid H$, i.e.,

$$\mathbb{E}[\tilde{Y}(1) - \tilde{Y}(0)] = \mathbb{E}[Y(1) - Y(0)] = \tau,$$
$$\mathbb{E}[\tilde{Y}(1) - \tilde{Y}(0) \mid H = k] = \mathbb{E}[Y(1) - Y(0) \mid H = k] = \tau_k.$$

and the tuple $\{(X_i, \tilde{Y}_i(0), \tilde{Y}_i(1), W_i)\}$ satisfies all conditions required by that theorem. Therefore,

$$\sqrt{n}\,(B_n - \tau) \to \mathcal{N}(0, V^\star),$$

submitted

with $V^\star = \mathbb{V}_{P_H}(\tau_H) + \mathbb{E}_{P_H}[V_H]$, where $V_H = \frac{\mathbb{V}(\tilde{Y}(1)|H)}{p_H} + \frac{\mathbb{V}(\tilde{Y}(0)|H)}{1-p_H}$. With Slutsky's theorem, this concludes that

$$\sqrt{n}(\hat{\tau} - \tau) \xrightarrow{d} \mathcal{N}(0, V^\star).$$

Finally, the unadjusted stratified difference-in-means estimator

$$\hat{\tau}_{\mathrm{DM}} = \sum_{k=1}^{K} \frac{n_k}{n} \left(\bar{Y}_k^{(1)} - \bar{Y}_k^{(0)} \right)$$

has an asymptotic variance of $V_{\mathrm{DM}} = \mathbb{V}_{P_H}(\tau_H) + \mathbb{E}_{P_H}[V_H^{\mathrm{DM}}]$, where $V_H^{\mathrm{DM}} = \frac{\mathbb{V}(Y(1)|H)}{p_H} + \frac{\mathbb{V}(Y(0)|H)}{1-p_H}$. By definition of $(d_w^\star, \beta_w^\star)$ in Equation 18, the parameters minimize the mean squared error such that:

$$\mathbb{E}\left[(Y(w) - d_w^{\star\top} H - X^\top \beta_w^\star)^2 \mid H\right] \le \mathbb{E}\left[(Y(w) - \mathbb{E}[Y(w) \mid H])^2 \mid H\right] = \mathbb{V}(Y(w) \mid H).$$

This implies $\mathbb{V}(\tilde{Y}(w) \mid H) \le \mathbb{V}(Y(w) \mid H)$, which yields $V^\star \le V_{\mathrm{DM}}$, concluding the proof. ☒ □

Lemma 1. *The estimator $\hat{\tau}$ can be written in the closed form given in Equation 17.*

Proof. The objective in Equation 4 splits over centers, $\sum_k \sum_{i \in \mathcal{C}_k^{(w)}} (Y_i - d_k - X_i^\top \beta)^2$, so the optimality condition for $\hat{d}_k$ is $-2n_k^{(w)}\{\bar{Y}_k^{(w)} - \hat{d}_k - (\bar{X}_k^{(w)})^\top \beta\} = 0$, whence

$$\hat{d}_k^{(w)} = \bar{Y}_k^{(w)} - \left(\bar{X}_k^{(w)}\right)^\top \hat{\beta}^{(w)}. \tag{21}$$

Substituting Equation 21 into $\hat{\tau}$ yields,

$$\begin{aligned}
\hat{\tau} &= \sum_k \frac{n_k}{n} \left(\hat{d}_k^{(1)} - \hat{d}_k^{(0)} + \bar{X}_k^\top \left(\hat{\beta}^{(1)} - \hat{\beta}^{(0)} \right) \right) \\
&= \sum_k \frac{n_k}{n} \left(\bar{Y}_k^{(1)} - \bar{Y}_k^{(0)} + \left(\bar{X}_k - \bar{X}_k^{(1)} \right)^\top \hat{\beta}^{(1)} - \left(\bar{X}_k - \bar{X}_k^{(0)} \right)^\top \hat{\beta}^{(0)} \right).
\end{aligned}$$

Since $\bar{X}_k = \hat{p}_k \bar{X}_k^{(1)} + (1 - \hat{p}_k)\bar{X}_k^{(0)}$ we have $\bar{X}_k - \bar{X}_k^{(1)} = -(1 - \hat{p}_k)(\bar{X}_k^{(1)} - \bar{X}_k^{(0)})$ and $\bar{X}_k - \bar{X}_k^{(0)} = \hat{p}_k(\bar{X}_k^{(1)} - \bar{X}_k^{(0)})$; collecting the slope terms gives Equation 17. ☒ □

Appendix D: IRLS vs FedAvg

Recent approaches to federated causal inference employ varying optimization strategies for nuisance parameter estimation. For instance, Khellaf et al. (2024) estimate regression functions using a multi-shot Federated Averaging (FedAvg) procedure restricted to a single local update per communication round. This configuration corresponds to the standard recursion of centralized gradient descent:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla \mathcal{L}_k(\theta^{(t)}),$$

where η is the learning rate, $\theta^{(t)}$ is the global parameter at round t, and $\nabla \mathcal{L}_k(\theta^{(t)})$ is the local gradient of the empirical loss computed at site k.

In contrast, DataSHIELD implements generalized linear models (via the `ds.glm` function) using a distributed Iteratively Reweighted Least Squares (IRLS) algorithm. In this approach, the central server broadcasts the current parameter estimates at each iteration. Each site then computes local

Submitted

working responses and information matrices based on its local data. The server aggregates these local matrices to compute a Newton-Raphson update:

$$\theta^{(t+1)} = \theta^{(t)} - \left(\sum_{k=1}^{K} \nabla^2 \mathscr{L}_k(\theta^{(t)}) \right)^{-1} \sum_{k=1}^{K} \nabla \mathscr{L}_k(\theta^{(t)}),$$

where $\nabla^2 \mathscr{L}_k(\theta^{(t)})$ denotes the local Hessian (or empirical Fisher information matrix) evaluated at site k. This second-order method typically converges in fewer communication rounds than first-order gradient approaches like FedAvg, at the cost of larger amounts of float exchanged per round ($O(d^2)$).

Furthermore, in the specific context of linear regression with a quadratic loss function, the estimation admits a closed-form solution, enabling a one-shot federated procedure. Each site k computes its local uncentered covariance matrix $M_{k,a} = \sum_{i \in \mathscr{C}_k^{(a)}} Z_i Z_i^\top$ and cross-product vector $\nabla_{k,a} = \sum_{i \in \mathscr{C}_k^{(a)}} Z_i Y_i$ for each treatment arm a. The central server then aggregates these sufficient statistics to compute the global OLS estimator:

$$\hat{\theta}_a = \left(\sum_{k=1}^{K} M_{k,a} \right)^{-1} \left(\sum_{k=1}^{K} \nabla_{k,a} \right).$$

This one-shot aggregation retrieves the centralized pooled solution without requiring iterative communication. It is equivalent to the FedAvg algorithm limit as the number of communication rounds T goes to infinity.

Session information

```
#> R version 4.6.0 (2026-04-24)
#> Platform: x86_64-pc-linux-gnu
#> Running under: Ubuntu 24.04.5 LTS
#>
#> Matrix products: default
#> BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.12.0
#> LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.12.0  LAPACK version 3.12.0
#>
#> locale:
#>  [1] LC_CTYPE=C.UTF-8       LC_NUMERIC=C           LC_TIME=C.UTF-8
#>  [4] LC_COLLATE=C.UTF-8     LC_MONETARY=C.UTF-8    LC_MESSAGES=C.UTF-8
#>  [7] LC_PAPER=C.UTF-8       LC_NAME=C              LC_ADDRESS=C
#> [10] LC_TELEPHONE=C         LC_MEASUREMENT=C.UTF-8 LC_IDENTIFICATION=C
#>
#> time zone: Etc/UTC
#> tzcode source: system (glibc)
#>
#> attached base packages:
#> [1] stats     graphics  grDevices datasets  utils     methods   base
#>
#> other attached packages:
#> [1] MASS_7.3-66          DSLite_1.4.1         rly_1.7.8
#> [4] DSI_1.8.0            R6_2.6.1             progress_1.2.3
#> [7] dsFedAte_0.1.0       dsFedAteClient_0.1.0
#>
#> loaded via a namespace (and not attached):
```

submitted

```
#>  [1] shape_1.4.6.1           xfun_0.61               lattice_0.22-9
#>  [4] gamlss_5.5-5            vctrs_0.7.3             tools_4.6.0
#>  [7] childsds_0.9.11         Rdpack_2.6.6            generics_0.1.4
#> [10] parallel_4.6.0          tibble_3.3.1            pan_2.0
#> [13] pkgconfig_2.0.3         jomo_2.7-6              Matrix_1.7-5
#> [16] data.table_1.18.6.1     assertthat_0.2.1        lifecycle_1.0.5
#> [19] compiler_4.6.0          deldir_2.0-4            codetools_0.2-20
#> [22] htmltools_0.5.9         yaml_2.3.12             glmnet_5.1
#> [25] mice_3.19.0             pillar_1.11.1           nloptr_2.2.1
#> [28] crayon_1.5.3            tidyr_1.3.2             reformulas_0.4.4
#> [31] iterators_1.0.14        rpart_4.1.27            boot_1.3-32
#> [34] foreach_1.5.2           mitml_0.4-5             nlme_3.1-169
#> [37] tidyselect_1.2.1        digest_0.6.39           dplyr_1.2.1
#> [40] purrr_1.2.2             splines_4.6.0           gamlss.dist_6.1-11
#> [43] fastmap_1.2.0           grid_4.6.0              gamlss.data_6.0-7
#> [46] cli_3.6.6               magrittr_2.0.5          survival_3.8-6
#> [49] broom_1.0.13            prettyunits_1.2.0       backports_1.5.1
#> [52] dsBaseClient_6.3.6.9000 rmarkdown_2.32          interp_1.1-6
#> [55] nnet_7.3-20             lme4_2.0-6              hms_1.1.4
#> [58] evaluate_1.0.5          knitr_1.52              rbibutils_2.4.1
#> [61] rlang_1.3.0             Rcpp_1.1.2              glue_1.8.1
#> [64] dsBase_6.3.6.9000       renv_1.2.3              minqa_1.2.8
#> [67] jsonlite_2.0.0
```

submitted