26 September 2024

Rheinische Friedrich-Wilhelms-
Universität Bonn, Helmholtz Munich

Contact

roy.gusinow@helmholtz-muenchen.de

Speaker

Roy Gusinow

dsMatchIt Update

Federated Matching Methods for Covariate Balancing

Horizon2020
European Union Funding
for Research & Innovation

HELMHOLTZ MUNICH

MUDS Munich School for Data Science
HELMHOLTZ · TUM · LMU

UNIVERSITÄT BONN

Table of Contents

Introduction

Challenges in Estimating Causal Effects

- Confounding variables can bias results
- Treated and control groups may differ in important ways

Need for Adjusting Methods

- Essential to control for confounders
- Accurate estimation of treatment effects requires balancing covariates

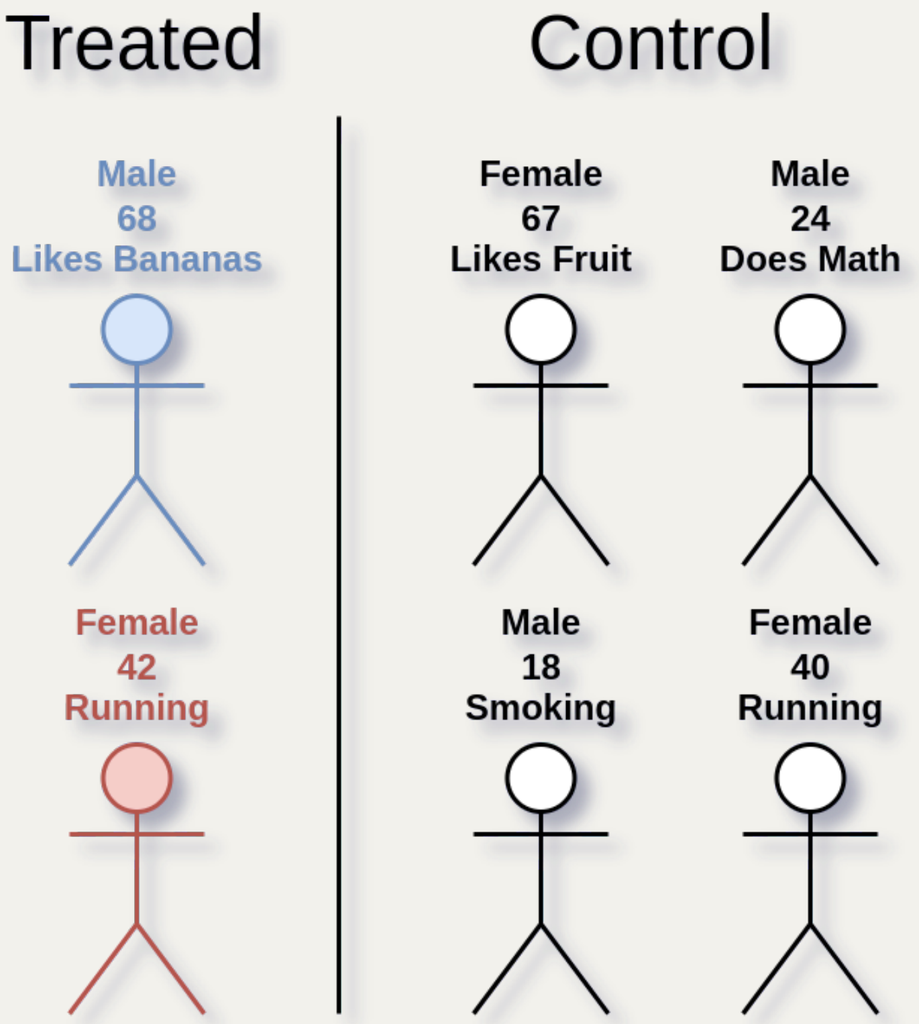
What is Matching?

Matching

- ***Pairs treated individuals*** with similar control individuals
- Based on key covariates to ensure comparability

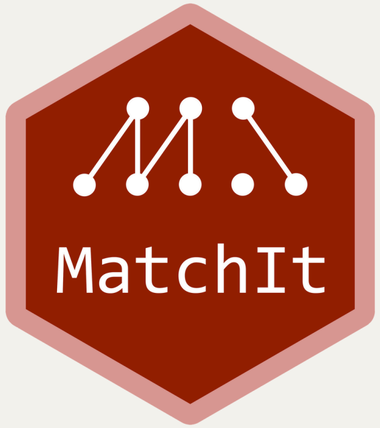
Aims

- ***Reduces bias*** from confounding variables
- Mimics a randomised controlled trial by balancing the control and treated groups

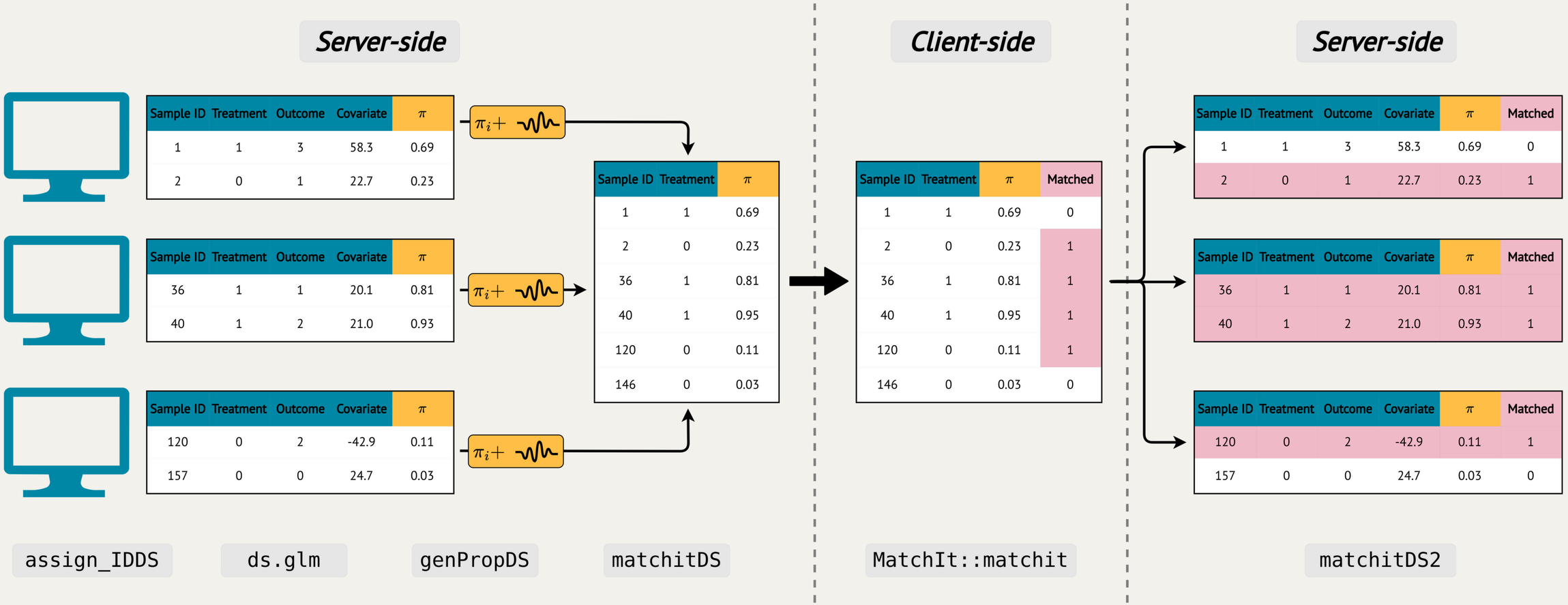


What is dsMatchIt?

- **Overview**
  - An extension of the MatchIt package for federated environments
  - Integrates with DataSHIELD for secure federated analysis
- **Features**
  - ***Performs matching across distributed datasets***
  - Supports various matching algorithms
  - Provides federated ATC, ATT and ATE functionality
  - Visualisations after matching and analysis



Steps in Federated Matching



Steps in Federated Matching

1. Define Distance Measure

- Calculate propensity scores within each data holder

2. Run Matching Algorithm

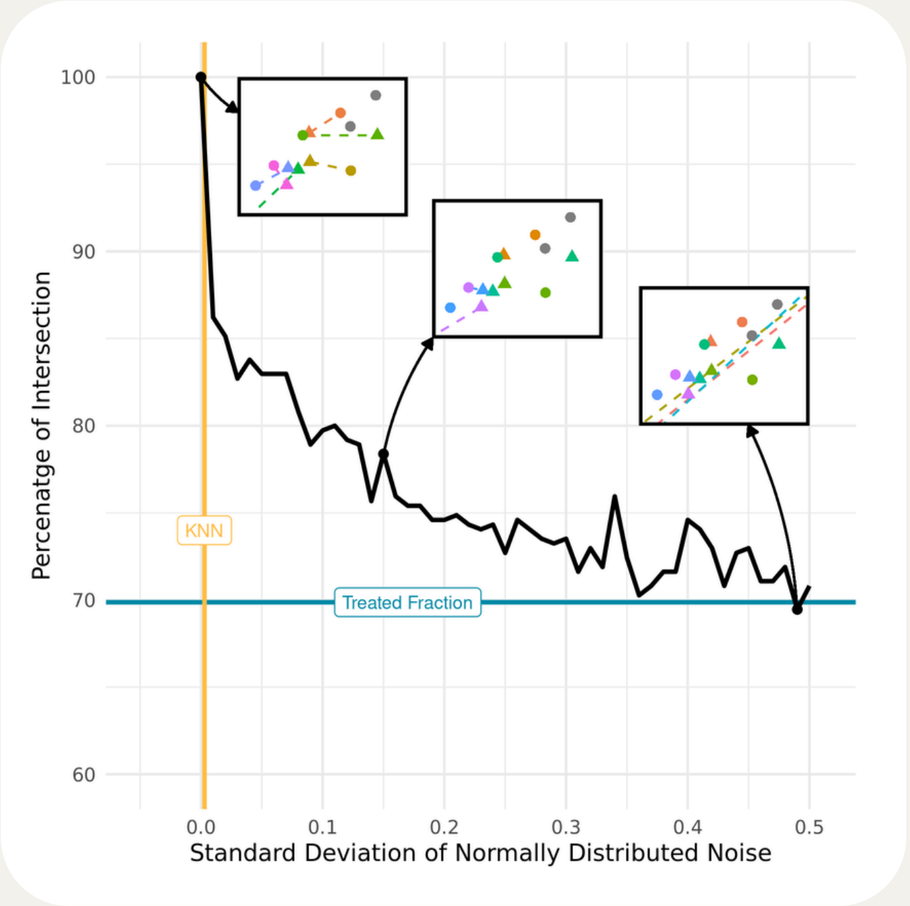
- Match individuals across datasets using aggregated data

3. Assess Covariate Balance

- Evaluate balance using federated summary statistics

4. Perform Analyses

- Estimate treatment effects on matched data



Steps in Federated Matching

1. Define Distance Measure

- Calculate propensity scores within each data holder.

2. Run Matching Algorithm

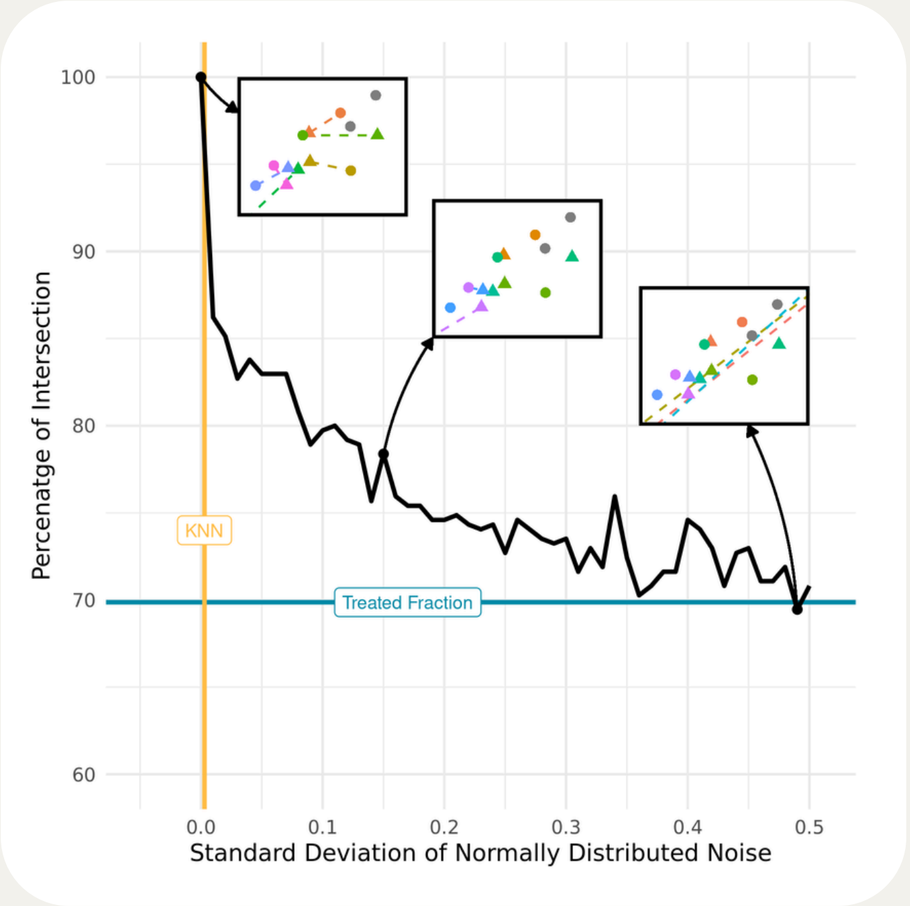
- Match individuals across datasets using aggregated data.

3. Assess Covariate Balance

- Evaluate balance using federated summary statistics.

4. Perform Analyses

- Estimate treatment effects on matched data.



Performing Analyses in dsMatchit

Uncertainity Quantification of the ATE and Federated Clustered Standard Errors

Performing Analyses

Calculation after Matching

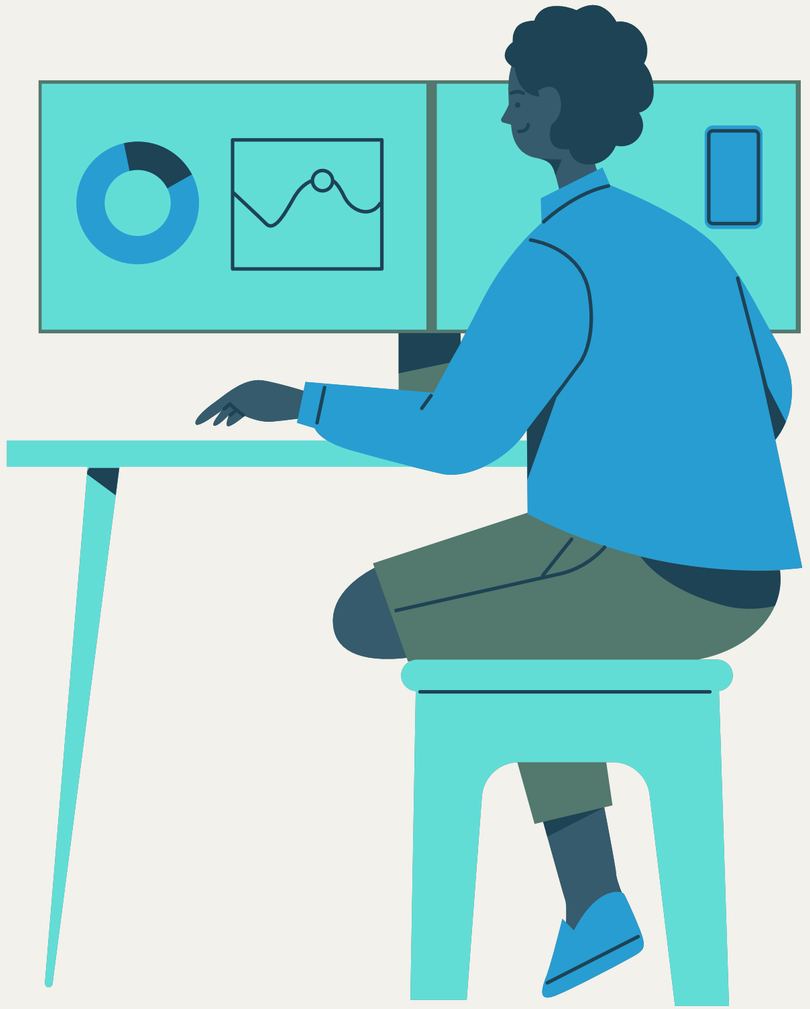
- Use matched data to compute mean differences
- This adjusts for any remaining imbalance

Average Treatment Effect (ATE)

- Calculating the difference in expected outcomes between groups

Robust Standard Errors of ATE

- Account for variability of data and clustering after matching
- ***Give uncertainty in the estimated effects***



Performing Analysis

We compare the average estimated difference of the outcome of the patients who would have received treatment against the outcome if the same patients did not receive the treatment. Practically, we can write the estimate as calculating averages:

Average Treatment Effect

$$ATE\left[X;\hat{\beta}\right] = \frac{1}{N}\sum_{s=1}^{S}\underbrace{\sum_{i=1}^{n_S(D=1)} Y_i(D=1;\hat{\beta})}_{\text{Calculated at each server, } s} - \frac{1}{N}\sum_{s=1}^{S}\underbrace{\sum_{i=1}^{n_S(D=0)} Y_i(D=0;\hat{\beta})}_{\text{Calculated at each server, } s},$$

where each summation is an aggregation, so it may be ***calculated at the server/data holder*** and returned to the central client using *ds.glm*.

Calculating Standard Errors using Delta Method

We now find the uncertainty around the Average Treatment Effect estimate. In the context of matching, there are 2 points to consider:

- Variability of the residuals may not consistent across observations. ***Heteroskedasticity-consistent standard errors*** rely on less assumptions
- There is now a ***dependence between observations***, as they have been matched

Thus, we need to calculate ***Cluster-robust Standard Errors***.

Delta Method via First-Order Taylor Approximation

$$\text{Var}\left[ATE\left(X;\hat{\beta}\right)\right] \approx \nabla ATE\left(X;\hat{\beta}\right)^{T} \cdot Cov(X)_{CL-R} \cdot \nabla ATE\left(X;\hat{\beta}\right)$$

Cluster-robust Covariance Matrix

The variance-covariance matrix can be derived,

$$Cov(X)_{CL-R} = \hat{B} \cdot \hat{M}_{CL} \cdot \hat{B},$$

Cluster-robust Covariance Matrix

The variance-covariance matrix can be derived,

Sandwich Estimator

$$Cov(X)_{CL-R} = \hat{B} \cdot \hat{M}_{CL} \cdot \hat{B},$$

with the ***sandwich estimator***.

Cluster-robust Covariance Matrix

The variance-covariance matrix can be derived,

Sandwich Estimator

$$Cov(X)_{CL-R} = \hat{B} \cdot \hat{M}_{CL} \cdot \hat{B},$$

with the ***sandwich estimator***.

To make our sandwich, we'll need to find our

- bread, $\hat{B}$ 
- meat, $\hat{M}_{CL}$ 
- in a secure and federated manner using DataShield

Cluster-robust Covariance Matrix - The Bread

The “bread” estimate is based on the empirical version of the inverse Hessian of the objective function, $\phi(Y, X, \hat{\beta})$. That is,

The Bread

$$\hat{B}(\beta) = \left(\mathbb{E}[-\frac{\partial \psi(Y, X, \hat{\beta})}{\partial \hat{\beta}}] \right)^{-1},$$

This is equivalent to the ***global covariance matrix of the estimated parameters***, $\hat{\beta}$ which is available to all data holders as it is aggregated.

Cluster-robust Covariance Matrix - The Meat

For the meat estimator, $\hat{M}_{CL}$, we can approximate this by the homoskedastic covariance which can be derived in a federated manor. The meat estimator $\hat{M}_{CL}$,

The Meat

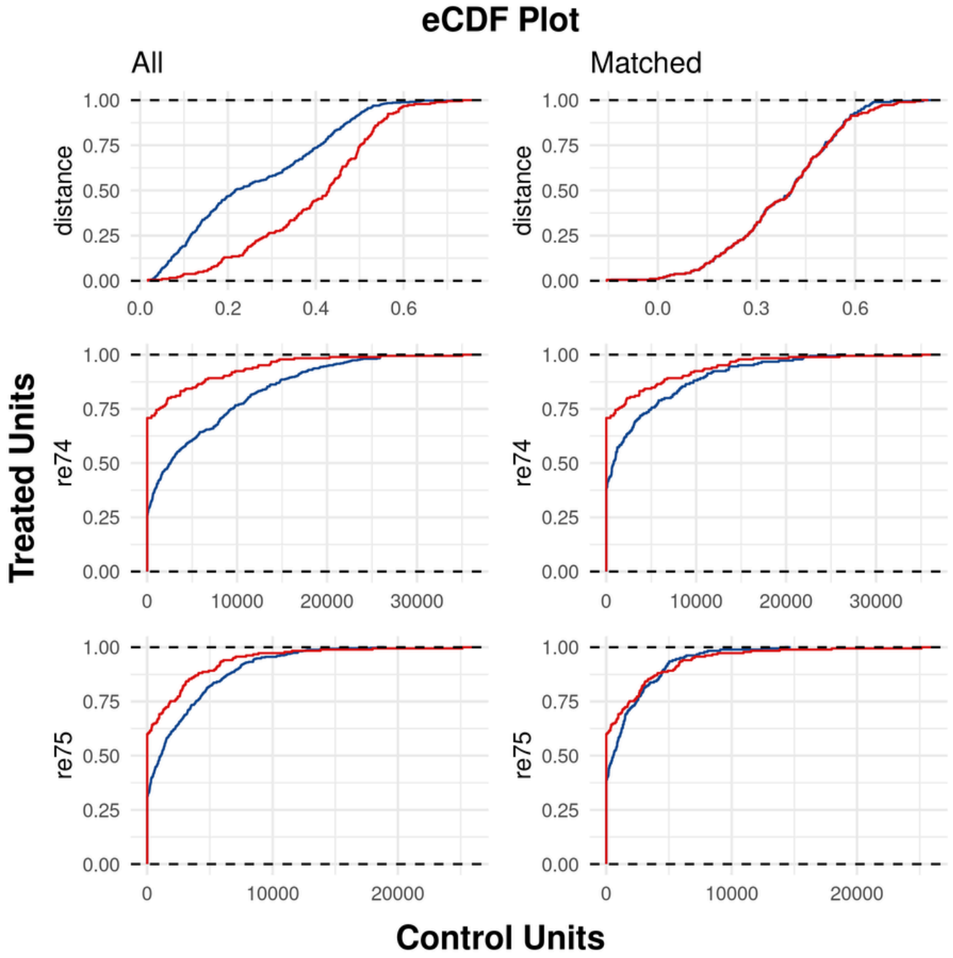
$$\hat{M}_{CL} = \frac{1}{N}\sum_{g=1}^{G} \underbrace{\left(\sum_{i=1}^{n_g} \psi\left(Y_{i,g}, X_{i,g}, \hat{\beta}\right)\right)}_{\text{Subclass summation at each server}} \underbrace{\left(\sum_{i=1}^{n_g} \psi\left(Y_{i,g}, X_{i,g}, \hat{\beta}\right)\right)^{T}}_{\text{Subclass summation at each server}}.$$

In ***low-permissive mode***, we utilise ***subclassification matching*** in which each data provider should have *g* stratified subsamples of the design matrix, *X*. Unlike 1-to-1 nearest neighbour matching, where we have *g = N/2* , in subclassifcation we typically set *g ≤ 10* for each server.

An Econometrics Example

Estimating the Effect of Training Programs on Average Salaries in 1978

An Econometrics Example

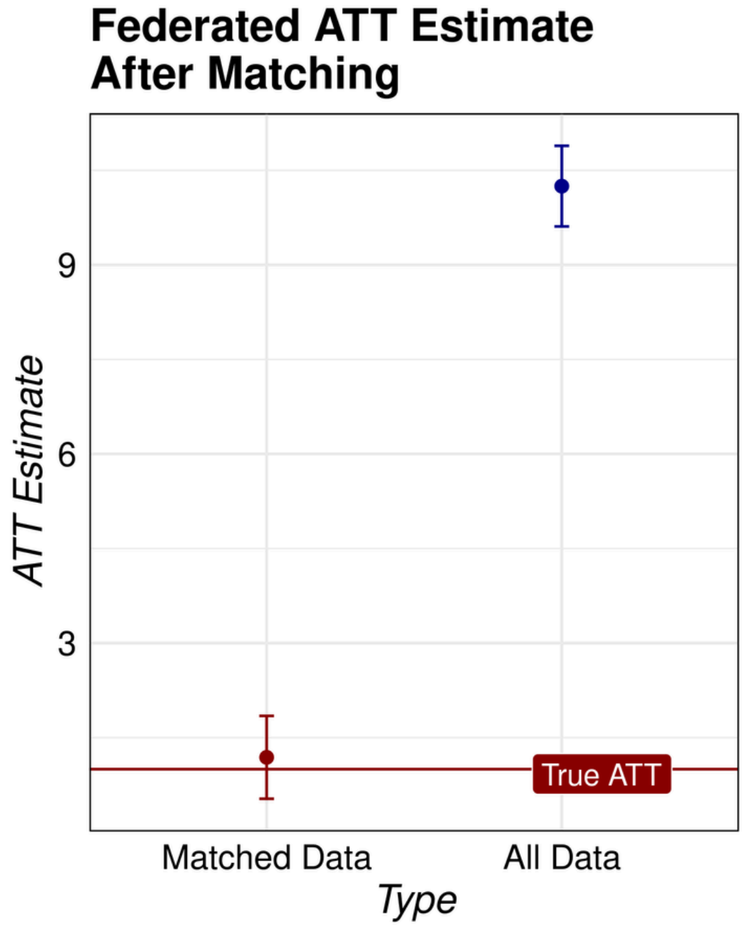


1. Define Distance Measure
2. Run Matching Algorithm
3. Assess Covariate Balance
4. Perform Analysis

```
ds.eCDF_plot(unmatched = "DST",
             matched = "matched_pooled",
             formula = treat ~ distance + re74 + re75)
```

Empirical Cumulative Distribution (eCDF) Plot before and after federated matching

An Econometrics Example



1. Define Distance Measure
2. Run Matching Algorithm
3. Assess Covariate Balance
4. Perform Analysis

```
avg_FL <- ds.avg_compute(fit = ds_fit,
                         data = "md",
                         treatment = "treat",

                         avg_type = "ATT",
                         error_type = HC_type,
                         clusters = clusters,
                         eps = 1e-7,
                         datasources = NULL)
```

dsMatchit on Github

Summary

Get summary statistics after the matching is done to see how good the matching procedure went with the additive noise to the propensity scores. Note that the corresponding domain of the eCDF cannot be provided as this is disclosive. As a result, the eCDF mean and maximum results are an approximation of the true resultsby contstruction and equidistance domain.

Centralised

```
# summary of unmatched/matched results
norm_summary <- summary(norm_match, un = T, improvement = T)
```

Federated

```
fed_summary.combined <- ds.match_summary(unmatched_obj = "DST",
                                         matched_obj = "matched_pooled",
                                         type = "combined",
                                         bin_num = 429,
                                         treatment = "treat")
```

Comparing

All Data

Centralised:

	Means Treated	Means Control	Std. Mean Diff.	Var. Ratio	eCDF Mean	eCDF Max	Std. Pair Dist.
distance	0.3950710	0.2608668	0.9490401	0.7549800	0.2314272	0.3700498	NA
age	25.8162162	28.0303030	-0.3094453	0.4399955	0.0813391	0.1577270	NA
educ	10.3459459	10.2354312	0.0549647	0.4958934	0.0347223	0.1113715	NA
married	0.1891892	0.5128205	-0.8263093	NA	0.3236313	0.3236313	NA
nodegree	0.7081081	0.5967366	0.2449702	NA	0.1113715	0.1113715	NA
re74	2095.5736886	5619.2365064	-0.7210838	0.5181285	0.2247949	0.4470358	NA
re75	1532.0553138	2466.4844431	-0.2902629	0.9562930	0.1341714	0.2876457	NA

The Overall Picture

- The federated method of the error estimates is ***equivalent*** to the non-federated calculation
- Bias corrections, *(HC0 – HC3)* are also handled
- Classification allows for ***high data security***, but users may can also choose a high permissive to conduct more accurate analyses too

Where can I find it?

- Complete manuscript is being written
- You can find the ***repository and corresponding tutorial*** on Github!

GitHub

26 September 2024

Rheinische Friedrich-Wilhelms-
Universität Bonn, Helmholtz Munich

Contact

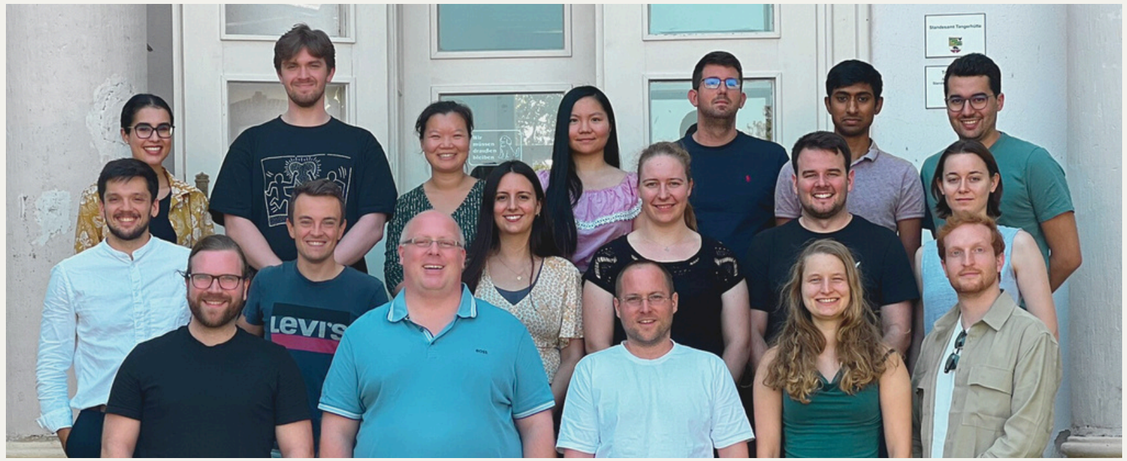
roy.gusinow@helmholtz-muenchen.de

Speaker

Roy Gusinow

Thank you for listening

and thanks to my lab group



In particular:
Jan Hasenauer
Manuel Huth
Carolina Alvarez
Jonas Arruda

Horizon2020
European Union Funding
for Research & Innovation

HELMHOLTZ MUNICH

MUDS Munich School for Data Science
HELMHOLTZ · TUM · LMU

UNIVERSITÄT BONN