ECMA 31360, PSet 3: Solutions

YOUR NAMES, University of Chicago

XX-XX-2026

([ ] out of 10p) PART I: Intent-to-Treat Versus Average Treatment Effect

([ ] out of 10p) Q1: An Exercise to Deep-dive the Difference b/w ITT and ATE

([ ] out of 2p) Q1.a: Describe ATE^o

([ ] out of 1p) Q1.b: Express $(Y_i^o(1), Y_i^o(0))$ as functions of $(Y_i(1), Y_i(0))$

([ ] out of 1p) Q1.c: Analytical Relationship between ATE^o and ATE

([ ] out of 2p) Q1.d: Selection into the NSW program based on flipping an unbalanced coin

---

([ ] out of 2p) Q1.e: Selection into the NSW program based on gains

---

([ ] out of 2p) Q1.f: When ITT and ATE Differ

([ ] out of 10p) PART II: Describe the Pseudo-Observational NSW Data

([ ] out of 3p) Q2: Compute Sample Averages of OPVs and outcome variable for PSID-1 and CPS-1 Samples

Script and Output

([ ] out of 4p) Q3: Compare the PSID-1 and CPS-1 Comparison Groups to the NSW-Treated Sample

---

([ ] out of 4p) Q4: Why do Dehajia and Wahba mimic observational data by combining the NSW-treated sample with survey data?

([ ] out of 10p bonus) PART III: Target Estimand: ATE versus ATT of the NSW Offer

([ ] out of 2p) Q5.a

([ ] out of 2p) Q5.b

([ ] out of 4p) Q5.c

([ ] out of 2p) Q5.d

([ ] out of 80p) PART IV: Regression-based Estimation of the Effect of the NSW Offer based on Pseudo-Observational Data

([ ] out of 9p) Q6: Implement the Treated-Control Comparison Estimator

([ ] out of 1p) Q6.a: Get the DM estimate and its SE under Homoskedasticity

Script and output

```
library(data.table)
library(dplyr)
library(lmtest)
library(sandwich)

# Robustly find the pseudo-observational file (course file name is nswpsid.csv)
fn <- c("nswpsid.csv", "nswpsid.csv")
fn <- fn[file.exists(fn)][1]
stopifnot(!is.na(fn))

dt <- data.table::fread(fn)

# Guidance: scale re74 and re75 by 1,000; also create agesq when needed later
dt[, re74 := re74/1000]
dt[, re75 := re75/1000]
dt[, agesq := age^2]
```

```r
# Keep treat numeric for OLS; create a factor copy for DoubleML's classif learner
dt[, treat_f := factor(treat, levels = c(0, 1))]

df <- as.data.frame(dt)

# DM regression (Specification 1)
m1 <- lm(re78 ~ treat, data = df)

# Homoskedastic SE (via coeftest + vcov(m1) per guidance)
tab_homosk <- lmtest::coeftest(m1, vcov. = vcov(m1))["treat", c("Estimate","Std. Error")]

tab_homosk
```

```
##   Estimate Std. Error
## -15204.776   1154.614
```

([ ] out of 3p) Q6.b: Get the Heteroskedasticity-robust SE of the DM estimator

Script and output

```r
tab_hc0 <- lmtest::coeftest(m1, vcov. = sandwich::vcovHC(m1, type = "HC0"))["treat", c("Estimate","Std. Error")]

m_re78_1 <- mean(df$re78[df$treat == 1])
m_re78_0 <- mean(df$re78[df$treat == 0])
dm_from_means <- m_re78_1 - m_re78_0

out_q6 <- rbind(
  homosk = c(rho_hat = unname(tab_homosk["Estimate"]), se = unname(tab_homosk["Std. Error"])),
  HC0    = c(rho_hat = unname(tab_hc0["Estimate"]),    se = unname(tab_hc0["Std. Error"]))
)

out_q6
```

```
##          rho_hat        se
## homosk -15204.78 1154.6143
## HC0    -15204.78  655.6691
```

```r
c(re78_1 = m_re78_1, re78_0 = m_re78_0, dm = dm_from_means)
```

```
##     re78_1     re78_0         dm
##   6349.145  21553.921 -15204.776
```

```r
stopifnot(isTRUE(all.equal(out_q6["homosk","rho_hat"], dm_from_means, tolerance = 1e-10)))
```

([ ] out of 5p) Q6.c: Discuss DM Estimator of ATT of NSW Offer

The difference-in-means (DM) estimator compares average post-treatment earnings in 1978 between the treated group (units with $D_i = 1$) and the control group (units with $D_i = 0$). In population terms, the DM estimand is

$$\mathbb{E}[Y_i \mid D_i = 1] - \mathbb{E}[Y_i \mid D_i = 0].$$

Using the potential outcomes framework and the measurement equation $Y_i = Y_i(1)D_i + Y_i(0)(1 - D_i)$, this estimand can be written as

$$\begin{aligned}\mathbb{E}[Y_i \mid D_i = 1] - \mathbb{E}[Y_i \mid D_i = 0] &= \mathbb{E}[Y_i(1) \mid D_i = 1] - \mathbb{E}[Y_i(0) \mid D_i = 0] \\ &= \underbrace{\mathbb{E}[Y_i(1) - Y_i(0) \mid D_i = 1]}_{\text{ATT}} + \underbrace{(\mathbb{E}[Y_i(0) \mid D_i = 1] - \mathbb{E}[Y_i(0) \mid D_i = 0])}_{\text{confounding term}}.\end{aligned}$$

Therefore, the DM estimator identifies the average treatment effect on the treated (ATT) if and only if

$$\mathbb{E}[Y_i(0) \mid D_i = 1] = \mathbb{E}[Y_i(0) \mid D_i = 0],$$

that is, if untreated potential outcomes are mean-independent of treatment assignment (unconditional mean independence).

In the pseudo-observational setting considered here, treatment is not randomly assigned: treated units (from the NSW experimental sample) and control units (from the PSID) differ systematically in baseline characteristics and pre-treatment earnings. As a result, the above condition is unlikely to hold, and the confounding term is generally nonzero. Consequently, the DM estimator does not have a credible causal interpretation as the ATT of the NSW offer in this context.

([ ] out of 8p) Q7: Prove Claim about identification and estimation of ATT

We prove Claim 1.

(i) Under (a)–(b), ATT is identified.

By definition,

$$ATT := \mathbb{E}[Y(1) - Y(0) \mid D = 1] = \mathbb{E}[\,\mathbb{E}(Y(1) - Y(0) \mid D = 1, X) \mid D = 1\,].$$

For treated units ($D = 1$), the observed outcome equals the treated potential outcome:

$$Y = Y(1) \quad \text{when } D = 1 \;\Rightarrow\; \mathbb{E}[Y(1) \mid D = 1, X] = \mathbb{E}[Y \mid D = 1, X].$$

Assumption (b) (CMIA0) states

$$\mathbb{E}[Y(0) \mid D = 1, X = x] = \mathbb{E}[Y(0) \mid D = 0, X = x] \;\; \forall x \in \mathcal{X}.$$

For control units ($D = 0$), the observed outcome equals the untreated potential outcome:

$$Y = Y(0) \quad \text{when } D = 0 \;\Rightarrow\; \mathbb{E}[Y(0) \mid D = 0, X] = \mathbb{E}[Y \mid D = 0, X].$$

Combining these,

$$\mathbb{E}[Y(1) - Y(0) \mid D = 1, X] = \mathbb{E}[Y \mid D = 1, X] - \mathbb{E}[Y \mid D = 0, X].$$

Hence,

$$\boxed{ATT = \mathbb{E}[\,\mathbb{E}[Y \mid D = 1, X] - \mathbb{E}[Y \mid D = 0, X] \mid D = 1\,].}$$

Assumption (a) (COC) guarantees that for each x in the relevant support there exist both treated and control units, so the conditional expectations above are well-defined.

(ii) Under (a)–(d), the observed conditional expectation function is linear:

$$\mathbb{E}[Y \mid D = d, X = x] = \alpha + \rho d + \beta' x.$$

When $D = 0$, we observe $Y = Y(0)$, and by (c),

$$\mathbb{E}[Y \mid D = 0, X = x] = \mathbb{E}[Y(0) \mid X = x] = \alpha_0 + \theta_0' x.$$

When $D = 1$, we observe $Y = Y(1)$, and by (d),

$$\mathbb{E}[Y \mid D = 1, X = x] = \mathbb{E}[Y(1) \mid D = 1, X = x] = \alpha_1 + \gamma + \theta_1' x.$$

With the homogeneity restriction $\theta_0 = \theta_1 = \theta$, define

$$\alpha := \alpha_0, \quad \beta := \theta, \quad \rho := (\alpha_1 + \gamma - \alpha_0).$$

Then for $d \in \{0,1\}$ we can write

$$\mathbb{E}[Y \mid D = d, X = x] = \alpha + \rho d + \beta' x,$$

which is linear in (d, x).

(iii) $\hat{\rho}$ is a consistent estimator of ρ.

Given the correct linear specification in (ii), i.i.d. sampling, and the usual OLS regularity conditions (in particular, no perfect multicollinearity), the OLS estimator $\hat{\rho}$ from the regression of Y on $(1, D, X)$ is consistent for the population coefficient ρ.

(iv) Under (a)–(d), $\rho = ATT$.

Using the conditional means above and $\theta_0 = \theta_1 = \theta$,

$$\mathbb{E}[Y(1) - Y(0) \mid D = 1, X = x] = (\alpha_1 + \gamma + \theta' x) - (\alpha_0 + \theta' x) = \alpha_1 + \gamma - \alpha_0 = \rho.$$

Since this expression does not depend on x,

$$ATT = \mathbb{E}[\mathbb{E}[Y(1) - Y(0) \mid D = 1, X] \mid D = 1] = \mathbb{E}[\rho \mid D = 1] = \rho.$$

Therefore, $\hat{\rho} \xrightarrow{p} \rho = ATT$, i.e., the OLS coefficient on D consistently estimates ATT under the stated assumptions.

([ ] out of 8p) Q8: Implement the Regression-Adjusted DM Estimator

([ ] out of 3p) Q8.a: Get the Adj.DM estimate and SE under Heteroskedasticity

Script and Output

```
# Spec 2: regression-adjusted DM per the exact x_i list in the problem
m2 <- lm(re78 ~ treat + age + agesq + edu + nodegree + black + hisp + re74 + re75, data = df)

tab2 <- lmtest::coeftest(m2, vcov. = sandwich::vcovHC(m2, type = "HC0"))
q8_report <- tab2["treat", c("Estimate","Std. Error","t value","Pr(>|t|)")]
q8_report
```

```
##    Estimate  Std. Error     t value    Pr(>|t|)
## 217.9438053 766.4444348   0.2843570   0.7761589
```

```
# (optional) 95% CI using robust SE
ci95 <- with(as.list(q8_report),
             c(lo = Estimate - 1.96*`Std. Error`, hi = Estimate + 1.96*`Std. Error`))
ci95
```

```
##        lo        hi
## -1284.287  1720.175
```

([ ] out of 5p) Q8.b: May the Adj.DM estimator improve over the DM estimator?

Yes, the Adj-DM estimator *may* improve over the DM estimator.

The DM estimator compares $\mathbb{E}[Y \mid D = 1] - \mathbb{E}[Y \mid D = 0]$ and identifies ATT only under a strong unconditional comparability condition such as

$$\mathbb{E}[Y(0) \mid D = 1] = \mathbb{E}[Y(0) \mid D = 0],$$

which is implausible in the pseudo-observational setting because treated (NSW) and controls (PSID) differ systematically in baseline characteristics and pre-treatment earnings.

In contrast, the adjusted difference-in-means (regression adjustment) estimates ρ from

$$Y_i = \alpha + \rho D_i + \beta' X_i + u_i,$$

where $X_i = (age_i, age_i^2, edu_i, nodegree_i, black_i, hisp_i, re74_i, re75_i)$. This approach attempts to account for *observable* differences between treated and controls. Under the conditions in Claim 1—in particular overlap (COC), conditional mean independence of untreated potential outcomes (CMIA0), and correct linear specification with homogeneous slopes—the population coefficient ρ equals ATT, and OLS provides a consistent estimator of ATT.

However, improvement is not guaranteed: if the linear functional form is misspecified or if there are important *unobserved* confounders not captured by X, the Adj-DM estimator may still be biased.

In summary, Adj-DM may improve over DM because it targets ATT under **conditional** comparability (after controlling for OPVs) rather than unconditional comparability. In Claim 1, ATT is identified if overlap holds and baseline conditional mean independence holds, i.e. $\mathbb{E}[Y(0) \mid D = 1, X = x] = \mathbb{E}[Y(0) \mid D = 0, X = x]$ for all x, and under the additional linear-in-parameters + homogeneous-slopes restrictions the regression coefficient ρ equals ATT. Thus, controlling for OPVs can reduce confounding from observed differences between NSW-treated and PSID individuals, potentially moving estimates toward the experimental benchmark. However, if the linear specification is misspecified or if selection operates through unobservables not captured by X, Adj-DM can remain biased.

([ ] out of 6p) Q9: Verify the Partialling-out Interpretation of OLS

([ ] out of 2p) Q9.a: Implement Procedure B

Script and Output

```
library(lmtest)
library(sandwich)

# We assume you already ran Q8 and created:
#   df (data frame) and m2 (Spec 2 regression)
# If not, run Q8 chunk first.

# Procedure B:
# 1) Regress D on X and keep residuals vhat
m_D_on_X <- lm(treat ~ age + agesq + edu + nodegree + black + hisp + re74 + re75, data = df)
vhat <- resid(m_D_on_X)

# 2) Regress Y on vhat; slope should equal rho from the full regression
m_B <- lm(re78 ~ vhat, data = df)

rho_A <- coef(m2)[["treat"]]      # from full regression (Spec 2)
rho_B <- coef(m_B)[["vhat"]]      # from Procedure B
c(rho_A = rho_A, rho_B = rho_B, diff = rho_A - rho_B)
```

```
##        rho_A        rho_B         diff
## 2.179438e+02 2.179438e+02 2.221157e-10
```

```
summary(m_B)
```

```
##
## Call:
## lm(formula = re78 ~ vhat, data = df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
```

```
## -20705 -11277  -1067   8326 100724
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  20502.4      302.3  67.820   <2e-16 ***
## vhat           217.9     1344.3   0.162    0.871
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15640 on 2673 degrees of freedom
## Multiple R-squared:  9.833e-06,  Adjusted R-squared:  -0.0003643
## F-statistic: 0.02629 on 1 and 2673 DF,  p-value: 0.8712
```

([ ] out of 2p) Q9.b: Implement Procedure C

Script and Output

```r
# Procedure C:
# 1) Regress D on X and keep residuals vhat (reuse vhat if already computed)

# 2) Regress Y on X and keep residuals ehat
m_Y_on_X <- lm(re78 ~ age + agesq + edu + nodegree + black + hisp + re74 + re75, data = df)
ehat <- resid(m_Y_on_X)

# 3) Regress ehat on vhat (often with no intercept)
m_C <- lm(ehat ~ 0 + vhat, data = df)

rho_C <- coef(m_C)[["vhat"]]
c(rho_A = rho_A, rho_C = rho_C, diff = rho_A - rho_C)
```

```
##        rho_A        rho_C         diff
## 2.179438e+02 2.179438e+02 2.591207e-10
```

```r
summary(m_C)
```

```
##
## Call:
## lm(formula = ehat ~ 0 + vhat, data = df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -64762  -4325   -523   3823 110481
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## vhat    217.9      864.7   0.252    0.801
##
## Residual standard error: 10060 on 2674 degrees of freedom
## Multiple R-squared:  2.375e-05,  Adjusted R-squared:  -0.0003502
## F-statistic: 0.06352 on 1 and 2674 DF,  p-value: 0.801
```

([ ] out of 2p) Q9.c: Interpretation

The Frisch–Waugh–Lovell (FWL) theorem implies that the coefficient on D_i (here $treat_i$) in the multiple regression

$$Y_i = \alpha + \rho D_i + \beta' X_i + u_i$$

can be obtained by *partialling out* the covariates X_i from both D_i and Y_i.

In particular, let $\hat{v}_i$ be the residual from regressing D_i on X_i:

$$D_i = \pi' X_i + v_i, \qquad \hat{v}_i = D_i - \widehat{\pi' X_i}.$$

Procedure B then regresses Y_i on $\hat{v}_i$ and the resulting slope equals $\hat{\rho}$. Equivalently, let $\hat{\varepsilon}_i$ be the residual from regressing Y_i on X_i:

$$Y_i = \delta' X_i + \varepsilon_i, \qquad \hat{\varepsilon}_i = Y_i - \widehat{\delta' X_i}.$$

Procedure C regresses $\hat{\varepsilon}_i$ on $\hat{v}_i$ (often without an intercept) and the slope again equals $\hat{\rho}$.

Thus, partialling-out means that OLS identifies the effect of D using only the variation in D that is orthogonal (linearly unrelated) to X, and the corresponding orthogonal component of Y. Numerically, Procedures B and C should reproduce exactly the same $\hat{\rho}$ as the full regression (up to numerical precision). —

([ ] out of 18p) Q10: Implement the DML Estimator to estimate the partially-linear specification

([ ] out of 2p) Q10.a: Why may we prefer Specification 3 over Specification 2?

Specification 2 imposes a *fully linear* conditional expectation function,

$$\mathbb{E}[Y \mid D = d, X = x] = \alpha + \rho d + \beta' x,$$

so identification/consistency relies not only on overlap and conditional mean independence, but also on **correct linear functional form** (and the associated "homogeneous slopes" restriction discussed in earlier claims).

Specification 3 instead adopts a *partially linear* structure,

$$\mathbb{E}[Y \mid D = d, X = x] = \rho d + g(x),$$

which keeps the causal parameter of interest ρ linear in D but allows the relationship between Y and X to be flexible through $g(\cdot)$. In the provided implementation, the nuisance functions are estimated flexibly via machine-learning (here, random forests), and the "partialling out" score plus **cross-fitting** is used to reduce overfitting bias. This makes Specification 3 attractive when we are concerned that a purely linear model (Specification 2) may be misspecified.

([ ] out of 2p) Q10.b: Relate Procedure D to Procedure C

Procedure C and Procedure D share the same logic: **residualize** Y and D with respect to X, then regress the residualized outcome on the residualized treatment. The difference is that Procedure C residualizes using **OLS linear projections**, while Procedure D residualizes using **flexible first-stage predictors** (machine learning) for $l(x) = \mathbb{E}[Y \mid X = x]$) and $(m(x) = \mathbb{E}[D \mid X = x]$, and then applies cross-fitting.

([ ] out of 1p) Q10.c: Install packages (here or at the top)

Script and Output

```r
pkgs <- c("DoubleML", "data.table", "mlr3", "mlr3learners")
to_install <- pkgs[!sapply(pkgs, requireNamespace, quietly = TRUE)]
if (length(to_install) > 0) install.packages(to_install)

library(DoubleML)
```

```
## Warning: package 'DoubleML' was built under R version 4.5.2
```

```r
library(data.table)
library(mlr3)
```

```
## Warning: package 'mlr3' was built under R version 4.5.2
```

```r
library(mlr3learners)
```

```
## Warning: package 'mlr3learners' was built under R version 4.5.2
```

([ ] out of 1p) Q10.d: Convert to `data.table` (here or above)

Script and Output

```r
# Load pseudo-observational data (PSID controls)
dt <- data.table::fread("nswpsid.csv")

# Guidance: scale re74 and re75 by 1,000
dt[, re74 := re74 / 1000]
dt[, re75 := re75 / 1000]

# (Optional, if you want parity with Spec 2)
dt[, agesq := age^2]

# Ensure data.table
data.table::setDT(dt)
head(dt)
```

```
##    treat   age   edu black  hisp married  re74  re75     re78   u74   u75
##    <int> <int> <int> <int> <int>   <int> <num> <num>    <num> <int> <int>
## 1:     1    37    11     1     0       1     0     0  9930.05     1     1
## 2:     1    30    12     1     0       0     0     0 24909.50     1     1
## 3:     1    27    11     1     0       0     0     0  7506.15     1     1
## 4:     1    33     8     1     0       0     0     0   289.79     1     1
## 5:     1    22     9     1     0       0     0     0  4056.49     1     1
## 6:     1    23    12     1     0       0     0     0     0.00     1     1
##    nodegree agesq
##       <int> <num>
## 1:        1  1369
## 2:        0   900
## 3:        1   729
## 4:        1  1089
## 5:        1   484
## 6:        0   529
```

([ ] out of 1p) Q10.e: Collect OPVs names (here or above)

Script and Output

```
# Original OPVs (pre-treatment covariates) in the NSW-PSID pseudo-observational file
pretreat_colnames <- c(
  "age", "edu", "black", "hisp", "married",
  "re74", "re75", "u74", "u75", "nodegree"
)

pretreat_colnames
```

```
##  [1] "age"      "edu"      "black"    "hisp"     "married"  "re74"
##  [7] "re75"     "u74"      "u75"      "nodegree"
```

([ ] out of 2p) Q10.f: Specify the "causal model"

Script and Output

```
dml_data_psid <- DoubleML::DoubleMLData$new(
  dt,
  y_col = "re78",
  d_cols = "treat",
  x_cols = pretreat_colnames
)

dml_data_psid
```

```
## ================= DoubleMLData Object ==================
##
##
## ------------------ Data summary      ------------------
## Outcome variable: re78
## Treatment variable(s): treat
## Covariates: age, edu, black, hisp, married, re74, re75, u74, u75, nodegree
## Instrument(s):
## Selection variable:
## No. Observations: 2675
```

```
class(dml_data_psid)
```

```
## [1] "DoubleMLData" "R6"
```

Commentary

This declares the outcome $Y =$ `re78`, treatment $D =$ `treat`, and covariates X as the OPVs. DoubleML will construct the partially linear regression target where $l(x) = \mathbb{E}[Y \mid X = x]$ and $m(x) = \mathbb{E}[D \mid X = x]$ are nuisance components estimated in the first stage, followed by the "partialling out" second stage.

([ ] out of 1p) Q10.g: Suppress messages

Script and Output

```
# Suppress verbose messages from mlr3 (per instructions)
if (requireNamespace("lgr", quietly = TRUE)) {
  lgr::get_logger("mlr3")$set_threshold("warn")
}
```

([ ] out of 2p) Q10.h: Specify learners from $m(\cdot)$ and $l(\cdot)$

Script and Output

```
# Specify an RF model as the learner model for l(x)=E[re78|X=x]
ml_l_rf <- mlr3::lrn("regr.ranger")

# Specify an RF model as the learner model for m(x)=E[treat|X=x]
# For classification, ensure probability predictions
ml_m_rf <- mlr3::lrn("classif.ranger", predict_type = "prob")

ml_l_rf
```

```
##
## -- <LearnerRegrRanger> (regr.ranger): Random Forest ------------------------------
## * Model: -
## * Parameters: num.threads=1, sigma2.threshold=0.01
## * Packages: mlr3, mlr3learners, and ranger
## * Predict Types: [response], se, and quantiles
## * Feature Types: logical, integer, numeric, character, factor, and ordered
## * Encapsulation: none (fallback: -)
## * Properties: hotstart_backward, importance, missings, oob_error,
## selected_features, and weights
## * Other settings: use_weights = 'use'
```

```
ml_m_rf
```

```
##
## -- <LearnerClassifRanger> (classif.ranger): Random Forest ----------------------
## * Model: -
## * Parameters: num.threads=1
## * Packages: mlr3, mlr3learners, and ranger
## * Predict Types: response and [prob]
## * Feature Types: logical, integer, numeric, character, factor, and ordered
## * Encapsulation: none (fallback: -)
## * Properties: hotstart_backward, importance, missings, multiclass, oob_error,
## selected_features, twoclass, and weights
## * Other settings: use_weights = 'use'
```

([ ] out of 2p) Q10.i: Initialize DML object

Script and Output

```
set.seed(3141)

obj_dml_plr <- DoubleML::DoubleMLPLR$new(
  dml_data_psid,
  ml_l = ml_l_rf, ml_m = ml_m_rf,
  n_folds = 2,
  score = "partialling out",
  apply_cross_fitting = TRUE
)

obj_dml_plr
```

```
## ================== DoubleMLPLR Object ==================
##
##
## ------------------ Data summary      ------------------
## Outcome variable: re78
## Treatment variable(s): treat
## Covariates: age, edu, black, hisp, married, re74, re75, u74, u75, nodegree
## Instrument(s):
## Selection variable:
## No. Observations: 2675
##
## ------------------ Score & algorithm ------------------
## Score function: partialling out
## DML algorithm: dml2
##
## ------------------ Machine learner   ------------------
## ml_l: regr.ranger
## ml_m: classif.ranger
##
## ------------------ Resampling        ------------------
## No. folds: 2
## No. repeated sample splits: 1
## Apply cross-fitting: TRUE
##
## ------------------ Fit summary       ------------------
##

## fit() not yet called.
```

([ ] out of 2p) Q10.j: Estimate the Partial-Linear Model

Script and Output

```
class(obj_dml_plr)
```

```
## [1] "DoubleMLPLR" "DoubleML"    "R6"
```

```r
methods("summary")
```

```
##  [1] summary.aov                          summary.aovlist*
##  [3] summary.aspell*                      summary.check_packages_in_dir*
##  [5] summary.connection                   summary.data.frame
##  [7] summary.Date                         summary.default
##  [9] summary.difftime                     summary.ecdf*
## [11] summary.factor                       summary.FutureJournal*
## [13] summary.glm                          summary.infl*
## [15] summary.lm                           summary.loess*
## [17] summary.loglm*                       summary.manova
## [19] summary.matrix                       summary.mlm*
## [21] summary.negbin*                      summary.nls*
## [23] summary.packageStatus*               summary.polr*
## [25] summary.POSIXct                      summary.POSIXlt
## [27] summary.ppr*                         summary.prcomp*
## [29] summary.princomp*                    summary.proc_time
## [31] summary.RichSOCKcluster*             summary.RichSOCKnode*
## [33] summary.rlang:::list_of_conditions* summary.rlang_error*
## [35] summary.rlang_message*               summary.rlang_trace*
## [37] summary.rlang_warning*               summary.rlm*
## [39] summary.shingle*                     summary.srcfile
## [41] summary.srcref                       summary.stepfun
## [43] summary.stl*                         summary.table
## [45] summary.Task*                        summary.trellis*
## [47] summary.tukeysmooth*                 summary.vctrs_sclr*
## [49] summary.vctrs_vctr*                  summary.warnings
## [51] summary.yearmon*                     summary.yearqtr*
## [53] summary.zoo*
## see '?methods' for accessing help and source code
```

```r
obj_dml_plr$fit()
```

```
## Warning: package 'future' was built under R version 4.5.2
```

```r
# Print (works for R6)
obj_dml_plr
```

```
## ================= DoubleMLPLR Object ==================
##
##
## ------------------ Data summary      ------------------
## Outcome variable: re78
## Treatment variable(s): treat
## Covariates: age, edu, black, hisp, married, re74, re75, u74, u75, nodegree
## Instrument(s):
## Selection variable:
## No. Observations: 2675
##
## ------------------ Score & algorithm ------------------
## Score function: partialling out
## DML algorithm: dml2
##
## ------------------ Machine learner   ------------------
## ml_l: regr.ranger
## ml_m: classif.ranger
##
## ------------------ Resampling        ------------------
```

```
## No. folds: 2
## No. repeated sample splits: 1
## Apply cross-fitting: TRUE
##
## ------------------ Fit summary       ------------------
##  Estimates and significance testing of the effect of target variables
##       Estimate. Std. Error t value Pr(>|t|)
## treat    -532.9     1077.5  -0.495    0.621
```

```r
# Use the R6-provided summary method if available
if ("summary" %in% names(obj_dml_plr)) {
  obj_dml_plr$summary()
}
```

```
## Estimates and significance testing of the effect of target variables
##       Estimate. Std. Error t value Pr(>|t|)
## treat    -532.9     1077.5  -0.495    0.621
```

```r
# Always safe: extract what you need directly
c(rho_hat = obj_dml_plr$coef, se = obj_dml_plr$se)
```

```
## rho_hat.treat      se.treat
##     -532.8684     1077.5354
```

([ ] out of 2p) Q10.k: DML estimate of ATT of NSW Offer

Script and Output

```r
# Fit DML (R6 object)
obj_dml_plr$fit()

# Print the fitted object (safe)
obj_dml_plr
```

```
## ================ DoubleMLPLR Object ==================
##
##
## ------------------ Data summary      ------------------
## Outcome variable: re78
## Treatment variable(s): treat
## Covariates: age, edu, black, hisp, married, re74, re75, u74, u75, nodegree
## Instrument(s):
## Selection variable:
## No. Observations: 2675
##
## ------------------ Score & algorithm ------------------
## Score function: partialling out
## DML algorithm: dml2
##
## ------------------ Machine learner   ------------------
## ml_l: regr.ranger
## ml_m: classif.ranger
##
## ------------------ Resampling        ------------------
```

```
## No. folds: 2
## No. repeated sample splits: 1
## Apply cross-fitting: TRUE
##
## ------------------ Fit summary       ------------------
##  Estimates and significance testing of the effect of target variables
##       Estimate. Std. Error t value Pr(>|t|)
## treat    -551.4     1071.9  -0.514    0.607
```

```
# Use the R6 summary method (NOT base::summary())
obj_dml_plr$summary()
```

```
## Estimates and significance testing of the effect of target variables
##       Estimate. Std. Error t value Pr(>|t|)
## treat    -551.4     1071.9  -0.514    0.607
```

```
# Extract estimate + SE directly (always safe)
rho_hat_dml <- as.numeric(obj_dml_plr$coef)
se_hat_dml  <- as.numeric(obj_dml_plr$se)
c(rho_hat_dml = rho_hat_dml, se_hat_dml = se_hat_dml)
```

```
## rho_hat_dml  se_hat_dml
##   -551.4114   1071.9258
```

The DML estimate of the ATT under Specification 3 is $\hat{\rho}_{\text{DML}} =$ `obj_dml_plr$coef` with standard error `obj_dml_plr$se`. With $n\_folds = 2$ and cross-fitting, the procedure produces two fold-specific estimates $\hat{\rho}_{1,2}$ and $\hat{\rho}_{2,1}$, and reports their average as $\hat{\rho}_{\text{DML}}$.

([ ] out of 10p) Q11: Compare ATT Estimates based on pseudo-observational data to Experimental estimates

```
library(lmtest)
library(sandwich)

# Helper: pull treat estimate + HC0 SE from an lm()
treat_est_hc0 <- function(mod, term = "treat") {
  tab <- lmtest::coeftest(mod, vcov. = sandwich::vcovHC(mod, type = "HC0"))
  c(att_hat = tab[term, "Estimate"], se = tab[term, "Std. Error"])
}

# Spec 1 (DM)
m1 <- lm(re78 ~ treat, data = df)
s1 <- treat_est_hc0(m1, term = "treat")

# Spec 2 (Adj-DM)
m2 <- lm(re78 ~ treat + age + agesq + edu + nodegree + black + hisp + re74 + re75, data = df)
s2 <- treat_est_hc0(m2, term = "treat")

# Spec 3 (DML) - assumes you ran Q10j already
s3 <- c(att_hat = as.numeric(obj_dml_plr$coef), se = as.numeric(obj_dml_plr$se))

# Experimental benchmark:
# Option A (preferred): compute from an experimental NSW file if present
bench <- c(att_hat = NA_real_, se = NA_real_)
```

```r
nsw_fn <- c("nsw.csv", "nsw_experimental.csv", "nsw_experiment.csv")
nsw_fn <- nsw_fn[file.exists(nsw_fn)][1]
if (!is.na(nsw_fn)) {
  nsw <- data.table::fread(nsw_fn)
  m_exp <- lm(re78 ~ treat, data = as.data.frame(nsw))
  bench <- treat_est_hc0(m_exp, term = "treat")
} else {
  # Option B: paste your PSet 2 DM numbers here (PSet3 background says ~1794)
  bench["att_hat"] <- 1794
  bench["se"]      <- NA_real_
}

est_table <- data.frame(
  spec   = c("Spec 1: DM", "Spec 2: Adj-DM (OLS)", "Spec 3: DML (PLR)", "Experimental benchmark (PSet 2)"),
  att_hat = c(s1["att_hat"], s2["att_hat"], s3["att_hat"], bench["att_hat"]),
  se      = c(s1["se"],      s2["se"],      s3["se"],      bench["se"])
)
est_table
```

```
##                              spec     att_hat        se
## 1                     Spec 1: DM -15204.7759  655.6691
## 2            Spec 2: Adj-DM (OLS)    217.9438  766.4444
## 3               Spec 3: DML (PLR)   -551.4114 1071.9258
## 4 Experimental benchmark (PSet 2)   1794.0000        NA
```

Commentary

Table X compares ATT estimates from the NSW–PSID pseudo-observational data under Specifications 1–3 to the experimental benchmark from PSet 2. In the pseudo-observational sample, the raw DM estimate (Spec 1) can differ sharply from the experimental estimate because the treated NSW sample and the PSID comparison sample differ systematically in pre-treatment variables; consequently, $\mathbb{E}[Y(0) \mid D = 1] \neq \mathbb{E}[Y(0) \mid D = 0]$ is plausible, so the confounding term is unlikely to be negligible.

Moving from Spec 1 to Spec 2 and Spec 3 progressively increases adjustment for OPVs: Spec 2 imposes a linear-in-parameters CEF in (D, X), while Spec 3 keeps ρ linear but allows $g(X)$ to be flexible and uses partialling-out with cross-fitting. In my results, the extent to which these adjustments move estimates toward the experimental benchmark indicates how much of the original confounding is explained by observed covariates (and, for DML, by non-linearities in their relationship to outcomes/treatment). In light of this comparison, LaLonde's concern is most compelling when the pseudo-observational design exhibits weak comparability; the more my estimates remain sensitive across Specs 1–3 (or remain far from the experimental benchmark), the more it supports his skepticism.

([ ] out of 21p) Q12: Imbens and Xu's Defense: The importance of overlap and the use of trimming

A central lesson from Imbens and Xu is that lack of overlap ("common support") can be a first-order obstacle to replicating experimental benchmarks with nonexperimental methods. When treated units have covariate profiles that are rare or absent among available controls, estimators must rely on extrapolation, and even sophisticated adjustments can become fragile. In the LaLonde-style designs (treated from the experiment, controls from large observational datasets), this can show up as treated propensity scores lying outside the support of control propensity scores, which undermines the credibility of any identification strategy that conditions on covariates.

Imbens and Xu emphasize trimming as a practical, data-driven way to improve overlap by restricting attention to units where treated and control observations are comparable. Their trimming procedure is designed to remove observations (including possibly treated units) whose propensity-score profiles do not overlap well with the other group. This improves the comparability of the remaining sample and reduces reliance on extrapolation—though it comes at the cost of smaller effective sample sizes and a change in the target population (the estimand becomes an ATT for the trimmed subpopulation).

Reading Imbens and Xu can soften LaLonde's broad skepticism: the inability to replicate benchmarks is not solely "because nonexperimental methods are bad," but often because the *design* generates poor overlap, making the task nearly impossible without restricting the sample. Once overlap is improved through trimming, a wide range of estimators can become substantially more stable and may cluster around the experimental benchmark. Thus, whether nonexperimental methods "work" depends critically on overlap diagnostics and careful design decisions (including trimming), not only on the choice among regression, matching, weighting, or DML.

```
# Propensity score model using OPVs (logit)
ps_mod <- glm(
  treat ~ age + edu + black + hisp + married + re74 + re75 + u74 + u75 + nodegree,
  family = binomial(),
  data = df
)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
df$ps_hat <- as.numeric(predict(ps_mod, type = "response"))

# Overlap diagnostics: treated vs control support
rng_t1 <- range(df$ps_hat[df$treat == 1])
rng_t0 <- range(df$ps_hat[df$treat == 0])
c(rng_treated_lo = rng_t1[1], rng_treated_hi = rng_t1[2],
  rng_ctrl_lo    = rng_t0[1], rng_ctrl_hi    = rng_t0[2])
```

```
## rng_treated_lo rng_treated_hi    rng_ctrl_lo    rng_ctrl_hi
##   2.812098e-04   9.902271e-01   2.220446e-16   9.865269e-01
```

```
# Support-based trimming: keep intersection of supports
lo <- max(rng_t1[1], rng_t0[1])
hi <- min(rng_t1[2], rng_t0[2])
df_trim <- subset(df, ps_hat >= lo & ps_hat <= hi)

c(N_full = nrow(df), N_trim = nrow(df_trim), lo = lo, hi = hi)
```

```
##       N_full       N_trim           lo           hi
## 2.675000e+03 1.390000e+03 2.812098e-04 9.865269e-01
```

```
# (Optional) re-estimate Spec 2 on trimmed sample to see stability
m2_trim <- lm(re78 ~ treat + age + agesq + edu + nodegree + black + hisp + re74 + re75, data = df_trim)
lmtest::coeftest(m2_trim, vcov. = sandwich::vcovHC(m2_trim, type = "HC0"))["treat", ]
```

```
##     Estimate   Std. Error      t value     Pr(>|t|)
## 1000.8481873  884.3468484    1.1317372    0.2579416
```