

# Day 1, Practical 1, Hely's solution

Helene Charlotte Wiese Rytgaard

May 30, 2023

## 1 Simulating data

Task 1.

```
library(data.table)

sim.fun <- function(n) {

  X1 <- runif(n, -2, 2)
  X2 <- rnorm(n)
  X3 <- rbinom(n, 1, prob=0.2)
  A <- rbinom(n, 1, prob=plogis(-0.25+0.8*X1+0.25*X3))
  Y <- rbinom(n, 1, prob=plogis(-0.9+1.9*X1^2+0.6*X2+0.5*A))

  return(data.table(X1=X1, X2=X2, X3=X3, A=A, Y=Y))

}
```

## 2 Computing the true value of the ATE

Task 2.

```
library(data.table)

sim.fun <- function(n, a=NULL) {
  X1 <- runif(n, -2, 2)
  X2 <- rnorm(n)
  X3 <- rbinom(n, 1, 0.2)
  if (length(a)>0) {
    A <- a
  } else {
    A <- rbinom(n, 1, prob=plogis(-0.25 + 0.8*X1 + 0.25*X3))
  }
  Y <- rbinom(n, 1, prob=plogis(-0.9 + 1.9*X1^2 + 0.6*X2 + 0.5*A))
  if (length(a)>0) {
    return(mean(Y))
  } else {
    return(data.table(id=1:n,X1=X1,X2=X2,X3=X3,A=A,Y=Y))
  }
}
```

```

set.seed(12)
message(paste0("EY1 = ", E.Y1 <- sim.fun(1e6, a=1)))
message(paste0("EY0 = ", E.Y0 <- sim.fun(1e6, a=0)))
message(paste0("ATE = ", ATE <- E.Y1 - E.Y0))

```

```

EY1 = 0.749921
EY0 = 0.68208
ATE = 0.0678409999999999

```

### 3 Estimation

#### Task 3.

```

set.seed(15)
head(sim.data <- sim.fun(1000))

```

```

id      X1      X2 X3 A Y
1:  1  0.4084562  0.38996075  0 0 0
2:  2 -1.2198243 -1.67449303  1 0 0
3:  3  1.8658349 -2.22881407  0 1 1
4:  4  0.6036221 -0.01388672  0 0 0
5:  5 -0.5317124  0.57686435  0 0 0
6:  6  1.9554368  0.15718650  0 0 1

```

```

message("fitted model for the outcome regression:")
summary(fit.f <- glm(Y~A+X1+X2+X3, family=binomial, data=sim.data))
message("-----")
message("fitted model for the propensity score:")
summary(fit.pi <- glm(A~X1+X2+X3, family=binomial, data=sim.data))

```

fitted model for the outcome regression:

Call:

```
glm(formula = Y ~ A + X1 + X2 + X3, family = binomial, data = sim.data)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.2530	-1.2483	0.6745	0.8255	1.3174

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	0.76917	0.10360	7.424	0.0000000000000113 ***
A	0.46609	0.16396	2.843	0.00447 **
X1	-0.01980	0.06944	-0.285	0.77554
X2	0.45090	0.07413	6.083	0.0000000001180179 ***
X3	0.23320	0.19117	1.220	0.22253

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1170.5 on 999 degrees of freedom  
Residual deviance: 1121.4 on 995 degrees of freedom  
AIC: 1131.4

Number of Fisher Scoring iterations: 4

-----  
fitted model for the propensity score:

Call:

```
glm(formula = A ~ X1 + X2 + X3, family = binomial, data = sim.data)
```

Deviance Residuals:

	Min	1Q	Median	3Q	Max
	-1.9285	-0.9355	-0.5552	0.9638	2.0656

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	-0.314837	0.079178	-3.976	0.00007 ***
X1	0.843342	0.066527	12.677	< 2e-16 ***
X2	-0.004127	0.068828	-0.060	0.9522
X3	0.379385	0.179716	2.111	0.0348 *

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1380.8 on 999 degrees of freedom  
Residual deviance: 1182.2 on 996 degrees of freedom  
AIC: 1190.2

Number of Fisher Scoring iterations: 4

```
##-- g-formula;
sim.data[, pred.EY1:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.EY0:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=0])]
message(paste0("g-formula estimate = ", round(sim.data[, mean(pred.EY1 - pred.EY0)],
5)))
##-- ipw;
sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
message(paste0("ipw estimate = ", round(sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred
.pi1))], 5)))
```

g-formula estimate = 0.08732

ipw estimate = 0.05979

Task 4.

```
fit.f2 <- glm(Y~A+X1.squared+X2+X3, family=binomial, data=sim.data[, X1.squared:=X1
~2])
```

```
fit.pi <- glm(A~X1+X2+X3, family=binomial, data=sim.data)
##-- g-formula;
sim.data[, pred.EY1:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.EY0:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=0])]
message(paste0("g-formula estimate = ", round(sim.data[, mean(pred.EY1 - pred.EY0)],
5)))
##-- ipw;
sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
message(paste0("ipw estimate = ", round(sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred
.pi1))], 5)))
```

g-formula estimate = 0.06703

ipw estimate = 0.05979

#### Task 5.

```
library(randomForestSRC)
fit.rf <- rfsrc(Y~A+X1+X2+X3, data=sim.data)
##-- g-formula;
sim.data[, pred.EY1:=predict(fit.rf, type="response", newdata=copy(sim.data)[, A:=1])$
predicted]
sim.data[, pred.EY0:=predict(fit.rf, type="response", newdata=copy(sim.data)[, A:=0])$
predicted]
message(paste0("g-formula estimate (RF) = ", round(sim.data[, mean(pred.EY1 - pred.EY0
)], 5)))
```

g-formula estimate (RF) = 0.06118

#### Task 6.

```
sim.data[, pred.EY1:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.EY0:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=0])]
sim.data[, mean(A/pred.pi1*(Y-pred.EY1) - (1-A)/(1-pred.pi1)*(Y-pred.EY0) + pred.EY1 -
pred.EY0)]
```

[1] 0.06638761

#### Task 7.

```
library(tmle)
tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
cbind(X1=sim.data$X1,X2=sim.data$X2,X3=sim.data$X3),
gform=A~X1+X2+X3, ## treatment model
Qform=Y~A+X1+X2+X3, ## outcome model
family="binomial",
cvQinit=FALSE)
##-- get the ATE estimate:
tmle.fit$estimates$ATE$psi
```

[1] 0.06626344

```
tmle.fit$Qinit$coef
fit.f$coef
```

(Intercept)	A	X1	X2	X3
0.76917331	0.46609109	-0.01979933	0.45089688	0.23320073

(Intercept)	A	X1	X2	X3
0.76917331	0.46609109	-0.01979933	0.45089688	0.23320073

#### Task 8.

```
tmle.fit2 <- tmle(Y=sim.data$Y, A=sim.data$A,
  cbind(X1=sim.data$X1, X1.squared=sim.data$X1^2,X2=sim.data$X2,X3=sim.data$X3
),
  gform=A~X1+X2+X3, ## treatment model
  Qform=Y~A+X1.squared+X2+X3, ## outcome model
  family="binomial",
  cvQinit=FALSE)
##-- get the ATE estimate:
tmle.fit2$estimates$ATE$psi
```

[1] 0.06791028

```
tmle.fit2$Qinit$coef
fit.f2$coef
```

(Intercept)	A	X1.squared	X2	X3
-0.8362875	0.5140438	1.8457299	0.5762989	0.1435289

(Intercept)	A	X1.squared	X2	X3
-0.8362875	0.5140438	1.8457299	0.5762989	0.1435289

```
sim.data[, pred.EY1:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.EY0:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=0])]
sim.data[, mean(A/pred.pi1*(Y-pred.EY1) - (1-A)/(1-pred.pi1)*(Y-pred.EY0) + pred.EY1 -
pred.EY0)]
```

[1] 0.06790105

## 4 Simulation study

#### Task 9.

```
library(here)

fit.g.glm1 <- list()
fit.g.glm2 <- list()
fit.g.rf <- list()
fit.ipw <- list()
fit.ipw2 <- list()
fit.ipw.rf <- list()
fit.tmle <- list()
fit.tmle.var <- list()
fit.ee <- list()
fit.ee.var <- list()
fit.tmle2 <- list()
fit.tmle2.var <- list()
```

```

fit.tee2 <- list()
fit.tee2.var <- list()

for (m in 1:500) {

  set.seed(m+110)
  sim.data <- sim.fun(1000)

  fit.f <- glm(Y~A+X1+X2+X3, family=binomial, data=sim.data)
  fit.f2 <- glm(Y~A+X1.squared+X2+X3, family=binomial, data=sim.data[, X1.squared:=X1
    ^2])
  fit.rf <- rfsrc(Y~A+X1+X2+X3, data=sim.data)
  fit.pi <- glm(A~X1+X2+X3, family=binomial, data=sim.data)
  fit.pi2 <- glm(A~X2+X3, family=binomial, data=sim.data)
  fit.rf.pi <- rfsrc(A~X1+X2+X3, data=sim.data)

  ##-- g-formula (misspecified);
  sim.data[, pred.f1:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=1])]
  sim.data[, pred.f0:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=0])]
  fit.g.glm1[[m]] <- sim.data[, mean(pred.f1 - pred.f0)]

  ##-- ipw (correctly specified);
  sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
  fit.ipw[[m]] <- sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred.pi1))]

  ##-- tmle (misspecified f);
  tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
    cbind(X1=sim.data$X1,X2=sim.data$X2,X3=sim.data$X3),
    gform=A~X1+X2+X3, ## treatment model
    Qform=Y~A+X1+X2+X3, ## outcome model
    family="binomial",
    cvQinit=FALSE)
  fit.tmle[[m]] <- tmle.fit$estimates$ATE$psi
  fit.tmle.var[[m]] <- tmle.fit$estimates$ATE$var

  ##-- estimating equation estimator (misspecified f);
  sim.data[, pred.f1:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=1])]
  sim.data[, pred.f0:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=0])]
  sim.data[, pred.f:=predict(fit.f, type="response", newdata=sim.data)]
  sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
  fit.tee[[m]] <- sim.data[, mean((A/pred.pi1 - (1-A)/(1-pred.pi1))*(Y - pred.f) +
    pred.f1 - pred.f0)]
  fit.tee.var[[m]] <- sim.data[, mean((A/pred.pi1*(Y-pred.f1) - (1-A)/(1-pred.pi1)*(Y-
    pred.f0) + pred.f1 - pred.f0)^2)/nrow(sim.data)]

  ##-- g-formula (correctly specified);
  sim.data[, pred.f1:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=1])
  ]
  sim.data[, pred.f0:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=0])
  ]
  fit.g.glm2[[m]] <- sim.data[, mean(pred.f1 - pred.f0)]

  ##-- tmle (correctly specified f);
  tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
    cbind(X1=sim.data$X1, X1.squared=sim.data$X1^2,X2=sim.data$X2,X3=sim.data$

```

```

X3),
  gform=A~X1+X2+X3, ## treatment model
  Qform=Y~A+X1.squared+X2+X3, ## outcome model
  family="binomial",
  cvQinit=FALSE)
fit.tmle2[[m]] <- tmle.fit$estimates$ATE$psi
fit.tmle2.var[[m]] <- tmle.fit$estimates$ATE$var

##-- estimating equation estimator (correctly specified f);
sim.data[, pred.f1:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=1])
]
sim.data[, pred.f0:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=0])
]
sim.data[, pred.f:=predict(fit.f2, type="response", newdata=sim.data)]
sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
fit.ee2[[m]] <- sim.data[, mean((A/pred.pi1 - (1-A)/(1-pred.pi1))*(Y - pred.f) +
  pred.f1 - pred.f0)]
fit.ee2.var[[m]] <- sim.data[, mean((A/pred.pi1*(Y-pred.f1) - (1-A)/(1-pred.pi1)*(Y-
  pred.f0) + pred.f1 - pred.f0)^2)/nrow(sim.data)]

##-- g-formula based on RF;
sim.data[, pred.f1:=predict(fit.rf, type="response", newdata=copy(sim.data)[, A:=1])
  $predicted]
sim.data[, pred.f0:=predict(fit.rf, type="response", newdata=copy(sim.data)[, A:=0])
  $predicted]
fit.g.rf[[m]] <- sim.data[, mean(pred.f1 - pred.f0)]

##-- ipw (misspecified);
sim.data[, pred.pi1:=predict(fit.pi2, type="response", newdata=sim.data)]
fit.ipw2[[m]] <- sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred.pi1))]

##-- rf (misspecified);
sim.data[, pred.pi1:=predict(fit.rf.pi, type="response", newdata=sim.data)$predicted
]
fit.ipw.rf[[m]] <- sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred.pi1))]

saveRDS(list(fit.g.glm1 = fit.g.glm1,
  fit.g.glm2 = fit.g.glm2,
  fit.g.rf = fit.g.rf,
  fit.ipw = fit.ipw,
  fit.ipw2 = fit.ipw2,
  fit.ipw.rf = fit.ipw.rf,
  fit.tmle = fit.tmle,
  fit.tmle.var = fit.tmle.var,
  fit.ee = fit.ee,
  fit.ee.var = fit.ee.var,
  fit.tmle2 = fit.tmle2,
  fit.tmle2.var = fit.tmle2.var,
  fit.ee2 = fit.ee2,
  fit.ee2.var = fit.ee2.var),
  file=paste0(here(), "/data/sim-data-output/",
    "save-est-sim-setting-1",
    ".rds"))
}

```

Task 10. See Figure 1.

```
library(ggplot2)

estimator.list <- readRDS(file=paste0(herc(), "/data/sim-data-output/",
                                     "save-est-sim-setting-1",
                                     ".rds"))

pdat <- data.table(estimator=c(rep("g-formula estimator (misspecified)",
                                  length(estimator.list$fit.g.glm1)),
                              rep("g-formula estimator (correctly specified)",
                                  length(estimator.list$fit.g.glm2)),
                              rep("g-formula estimator (random forest)",
                                  length(estimator.list$fit.g.rf)),
                              rep("IPW estimator (correctly specified)",
                                  length(estimator.list$fit.ipw)),
                              rep("IPW estimator (misspecified)",
                                  length(estimator.list$fit.ipw2)),
                              rep("IPW estimator (random forest)",
                                  length(estimator.list$fit.ipw.rf)),
                              rep("TMLE estimator (misspecified initial)",
                                  length(estimator.list$fit.tmle)),
                              rep("EE estimator (misspecified initial)",
                                  length(estimator.list$fit.tee))),
                  est=c(unlist(estimator.list$fit.g.glm1),
                        unlist(estimator.list$fit.g.glm2),
                        unlist(estimator.list$fit.g.rf),
                        unlist(estimator.list$fit.ipw),
                        unlist(estimator.list$fit.ipw2),
                        unlist(estimator.list$fit.ipw.rf),
                        unlist(estimator.list$fit.tmle),
                        unlist(estimator.list$fit.tee)))

pdat[, estimator.factor := factor(estimator, levels=pdat[, unique(estimator)], order=
  TRUE)]

ggplot(pdat) +
  theme_bw(base_size=25) +
  geom_boxplot(aes(x=est)) +
  facet_wrap(. ~ estimator.factor, ncol=2) +
  geom_vline(aes(xintercept=ATE), linetype="dashed", color="red") +
  xlab(expression(hat(psi)[n])) + ylab("")
```



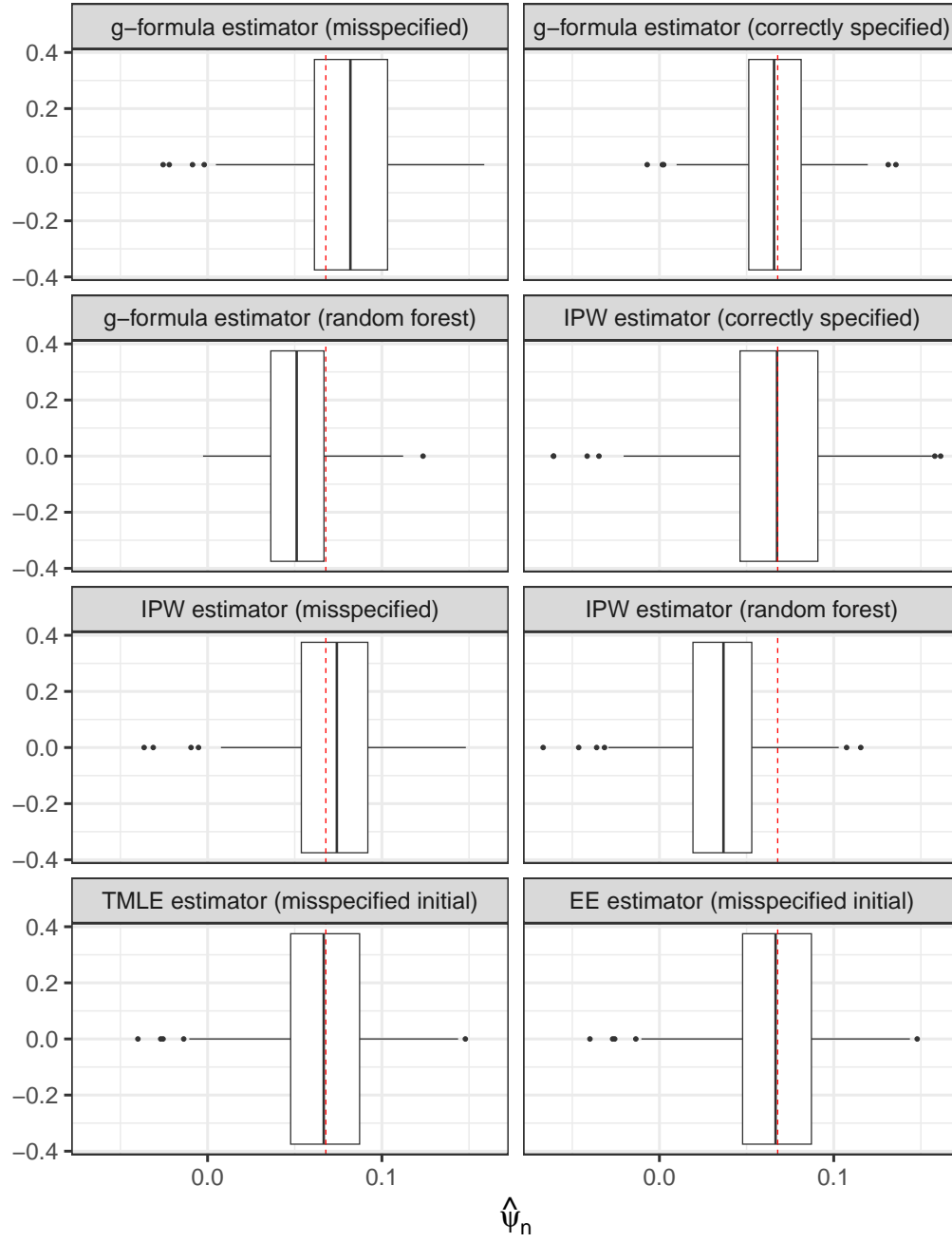


Figure 1

## 4.1 Extra: Tuning the random forest

```
setwd("~/Undervisning/TMLE/TMLE-course/practicals/day1/")
library(ggplot2)
library(data.table)
library(randomForestSRC)

ate.rf.models.list <- list()
V <- 5 # <- number of folds.
loss.fun <- function(Y, fit) -Y*log(fit)-(1-Y)*log(1-fit) # <- loss function.

for (m in 1:500) {

  set.seed(5+m)
  sim.data <- sim.fun(1000)
  sim.data[, id:=1:.N]

  #-- what random forests do we want to consider?
  rf.models <- list(rf=c(nodesize=7, mtry=1),
                    rf=c(nodesize=5, mtry=1),
                    rf=c(nodesize=10, mtry=2),
                    rf=c(nodesize=10, mtry=1),
                    rf=c(nodesize=7, mtry=2)
                  )

  #-- for cross-validation;
  cv.split <- matrix(sample(1:nrow(sim.data), size=nrow(sim.data)), ncol=V)

  for (kk in 1:length(rf.models)) {
    rf.model <- rf.models[[kk]]
    for (vv in 1:V) {
      test.set <- cv.split[,vv]
      train.set <- sim.data[, id][!sim.data[, id] %in% test.set]
      sim.data.train <- sim.data[id%in%train.set]
      sim.data.test <- sim.data[id%in%test.set]
      train.fit <- rfsrc(formula(paste0("Y~A+X1+X2+X3")),
                        data=sim.data.train,
                        nodesize=rf.model["nodesize"],
                        mtry=rf.model["mtry"])
      sim.data[id%in%test.set, (paste0("fit", kk)):=
        predict(train.fit,
                newdata=sim.data[id%in%test.set],
                type="response")$predicted]
    }
  }

  #-- compute cv error;
  cve.rf.models <- unlist(lapply(1:length(rf.models), function(kk) {
    sum(loss.fun(sim.data$Y, sim.data[, get(paste0("fit", kk))]))
  })))

  #-- fit all random forest models;
  for (kk in 1:length(rf.models)) {
    rf.model <- rf.models[[kk]]
    fit.rf <- rfsrc(formula(paste0("Y~A+X1+X2+X3")),
```

```

        data=sim.data,
        nodesize=rf.model["nodesize"],
        mtry=rf.model["mtry"])
sim.data[, (paste0("pred.rf.", kk, ".A1")):=predict(fit.rf, type="response",
newdata=copy(sim.data)[, A:=1])$predicted]
sim.data[, (paste0("pred.rf.", kk, ".A0")):=predict(fit.rf, type="response",
newdata=copy(sim.data)[, A:=0])$predicted]
}

cv.picked <- (1:length(rf.models))[(cve.rf.models==min(na.omit(cve.rf.models))) & !
is.na(cve.rf.models)]

ate.rf.models.list[[m]] <- c(sapply(1:length(rf.models), function(kk) sim.data[,
mean(get(paste0("pred.rf.", kk, ".A1"))-
get(paste0("pred.rf.", kk, ".A0")))]),
sim.data[, mean(get(paste0("pred.rf.", cv.picked, ".A1"))-
get(paste0("pred.rf.", cv.picked, ".A0")))]])

saveRDS(ate.rf.models.list,
file=paste0("~/Undervisning/TMLE/TMLE-course/data/sim-data-output/",
"save-est-sim-setting-RF",
".rds"))
}

```

```

library(ggplot2)
library(data.table)

table.rf.picked <- data.table(rf.picked=as.numeric(table(unlist(lapply(ate.rf.models.
list, function(rf.cve) {
rf.picked <- (1:length(rf.cve))[rf.cve==min(rf.cve)]
}))))/sum(as.numeric(table(unlist(lapply(ate.rf.models.list, function(rf.cve) {
rf.picked <- (1:length(rf.cve))[rf.cve==min(rf.cve)]
}))))), which=paste0("RF ", 1:length(ate.rf.models.list[[1]])))

dplot <- data.table(ate=unlist(lapply(1:(length(rf.models)+1), function(kk) sapply(ate
.rf.models.list, function(ate.rf) ate.rf[kk]))),
which=unlist(lapply(1:(length(rf.models)+1), function(kk) rep(paste0("RF "
, kk),
length(ate.rf.models.list))))))
dplot[which==paste0("RF ", length(rf.models)+1), which:="RF (CVE)"]

dplot <- merge(dplot, table.rf.picked, by="which", all.x=TRUE)
dplot[, rf.picked:=paste0("picked: ", round(rf.picked*100, 1), "%")]
dplot[rf.picked=="picked: NA%", rf.picked:=""]

ggplot(dplot) +
  theme_bw(base_size=25) +
  geom_boxplot(aes(x=ate)) +
  geom_vline(aes(xintercept=ATE), color="red", linetype="dashed") +
  geom_text(data=unique(dplot, by="rf.picked"), aes(x=0.11, y=-0.37, label=rf.picked
)) +
  facet_wrap(. ~ which) +
  ylab("") + xlab("ATE")

```

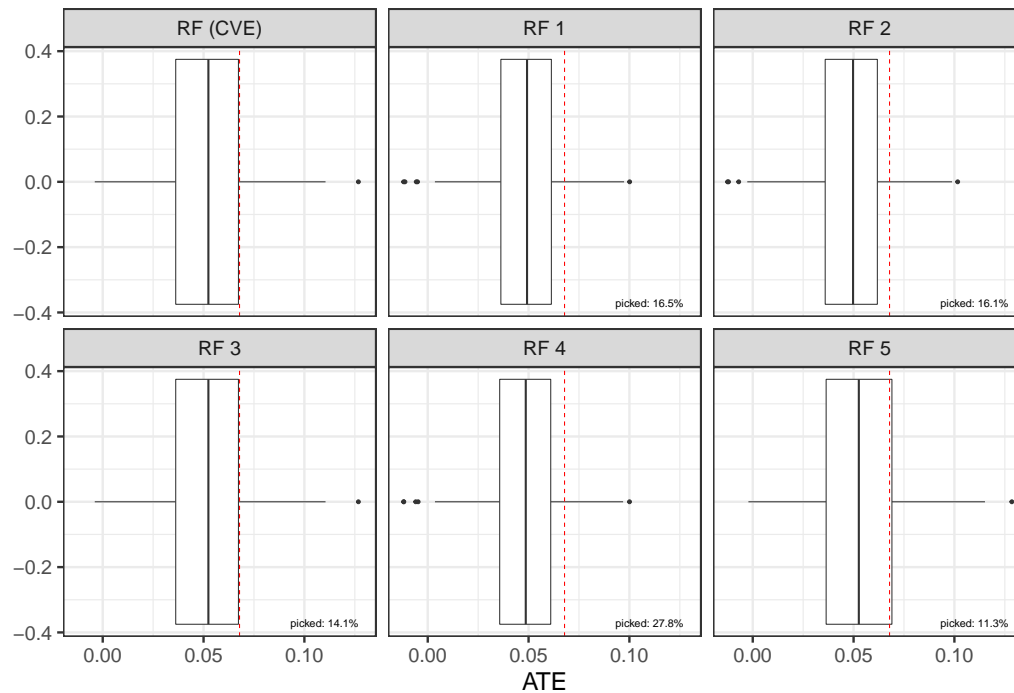


Figure 2

## 5 Changed data setting

### Task 11.

```
new.sim.fun <- function(n, a=NULL) {
  X1 <- runif(n, -2, 2)
  X2 <- rnorm(n)
  X3 <- rbinom(n, 1, 0.2)
  if (length(a)>0) {
    A <- a
  } else {
    A <- rbinom(n, 1, prob=plogis(-0.25 + 2.8*X1 + 0.25*X3))
  }
  Y <- rbinom(n, 1, prob=plogis(-0.9 + 1.9*X1^2 + 0.6*X2 + 0.5*A))
  if (length(a)>0) {
    return(mean(Y))
  } else {
    return(data.table(id=1:n, X1=X1, X2=X2, X3=X3, A=A, Y=Y))
  }
}
```

### Task (11)3.

```
set.seed(15)
head(sim.data <- new.sim.fun(1000))
```

```
id      X1      X2 X3 A Y
```

```

1:  1  0.4084562  0.38996075  0 1 1
2:  2 -1.2198243 -1.67449303  1 0 0
3:  3  1.8658349 -2.22881407  0 1 1
4:  4  0.6036221 -0.01388672  0 0 0
5:  5 -0.5317124  0.57686435  0 0 0
6:  6  1.9554368  0.15718650  0 1 1

```

```

message("fitted model for the outcome regression:")
summary(fit.f <- glm(Y~A+X1+X2+X3, family=binomial, data=sim.data))
message("-----")
message("fitted model for the propensity score:")
summary(fit.pi <- glm(A~X1+X2+X3, family=binomial, data=sim.data))

```

fitted model for the outcome regression:

Call:

```
glm(formula = Y ~ A + X1 + X2 + X3, family = binomial, data = sim.data)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.4128	-1.1028	0.6456	0.8032	1.3819

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	0.72501	0.13638	5.316	0.0000001061167 ***
A	0.72655	0.24014	3.026	0.00248 **
X1	-0.11379	0.10283	-1.107	0.26847
X2	0.51261	0.07640	6.710	0.0000000000195 ***
X3	0.09166	0.19105	0.480	0.63138

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1146.1 on 999 degrees of freedom  
Residual deviance: 1085.4 on 995 degrees of freedom  
AIC: 1095.4

Number of Fisher Scoring iterations: 4

-----  
fitted model for the propensity score:

Call:

```
glm(formula = A ~ X1 + X2 + X3, family = binomial, data = sim.data)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.6536	-0.3012	0.0803	0.3409	3.3465

Coefficients:

```

              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.26425     0.12320  -2.145   0.0320 *
X1           2.98819     0.18819  15.879   <2e-16 ***
X2          -0.07899     0.10454   -0.756   0.4499
X3           0.58430     0.28893   2.022   0.0431 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

(Dispersion parameter for binomial family taken to be 1)

```

Null deviance: 1385.72  on 999  degrees of freedom
Residual deviance:  538.79  on 996  degrees of freedom
AIC: 546.79

```

Number of Fisher Scoring iterations: 6

```

##-- g-formula;
sim.data[, pred.EY1:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.EY0:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=0])]
message(paste0("g-formula estimate = ", round(sim.data[, mean(pred.EY1 - pred.EY0)],
5)))
##-- ipw;
sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
message(paste0("ipw estimate = ", round(sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred
.pi1))], 5)))

```

```

g-formula estimate = 0.13182
ipw estimate = 0.39891

```

Note the support issues, violating the positivity assumption (see Figure 3):

```

library(ggplot2)
ggplot(sim.data) +
  theme_bw(base_size=25) +
  geom_histogram(aes(x=X1)) +
  facet_wrap(. ~ A, ncol=2) +
  xlab(expression(X[1])) + ylab("")

```

#### Task (11)4.

```

fit.f2 <- glm(Y~A+X1.squared+X2+X3, family=binomial, data=sim.data[, X1.squared:=X1
~2])
fit.pi <- glm(A~X1+X2+X3, family=binomial, data=sim.data)
##-- g-formula;
sim.data[, pred.EY1:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.EY0:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=0])]
message(paste0("g-formula estimate = ", round(sim.data[, mean(pred.EY1 - pred.EY0)],
5)))
##-- ipw;
sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
message(paste0("ipw estimate = ", round(sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred
.pi1))], 5)))

```

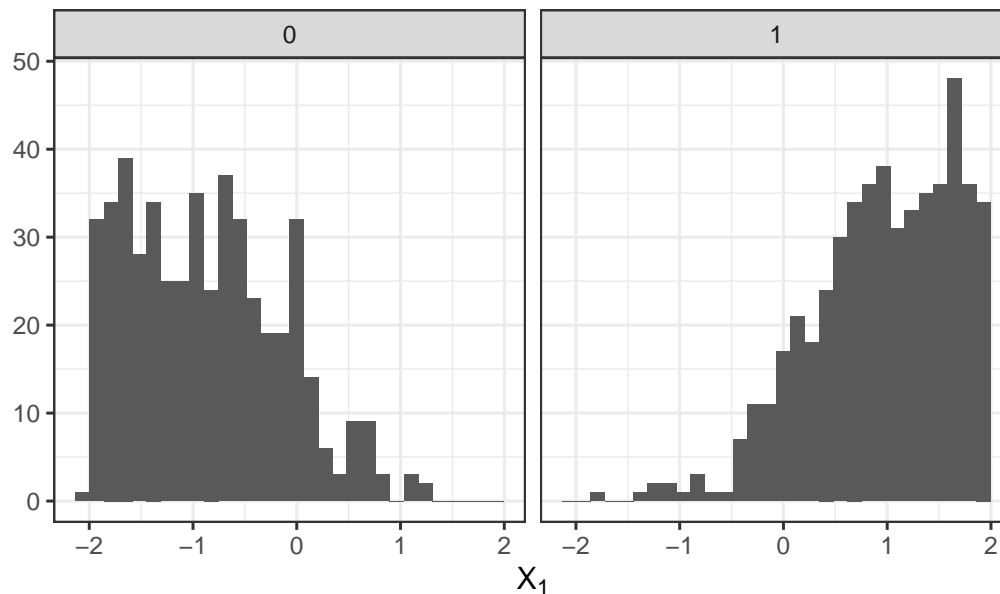


Figure 3

g-formula estimate = 0.07055  
 ipw estimate = 0.39891

Task (11)5.

```
library(randomForestSRC)
fit.rf <- rfsrc(Y~A+X1+X2+X3, data=sim.data)
##-- g-formula;
sim.data[, pred.EY1:=predict(fit.rf, type="response", newdata=copy(sim.data)[, A:=1])$
  predicted]
sim.data[, pred.EY0:=predict(fit.rf, type="response", newdata=copy(sim.data)[, A:=0])$
  predicted]
message(paste0("g-formula estimate (RF) = ", round(sim.data[, mean(pred.EY1 - pred.EY0
)], 5)))
```

g-formula estimate (RF) = 0.05446

Task (11)6.

```
sim.data[, pred.EY1:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.EY0:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=0])]
sim.data[, mean(A/pred.pi1*(Y-pred.EY1) - (1-A)/(1-pred.pi1)*(Y-pred.EY0) + pred.EY1 -
  pred.EY0)]
```

[1] 0.1742685

Task (11)7.

```
library(tmle)
tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
```

```

cbind(X1=sim.data$X1,X2=sim.data$X2,X3=sim.data$X3),
gform=A~X1+X2+X3, ## treatment model
Qform=Y~A+X1+X2+X3, ## outcome model
family="binomial",
cvQinit=FALSE)
#-- get the ATE estimate:
tmle.fit$estimates$ATE$psi

```

[1] 0.1401536

How much weight truncation is going on?

```

summary(tmle.fit$g$g1W)
tmle.fit$gbound
mean(tmle.fit$g$g1W<tmle.fit$gbound)

```

```

      Min.   1st Qu.   Median     Mean 3rd Qu.    Max.
0.001739 0.049518 0.533751 0.512000 0.952438 0.998164
[1] 0.02288933 1.00000000
[1] 0.601

```

Changing the gbound argument to avoid truncation of weights:

```

library(tmle)
tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
  cbind(X1=sim.data$X1,X2=sim.data$X2,X3=sim.data$X3),
  gform=A~X1+X2+X3, ## treatment model
  Qform=Y~A+X1+X2+X3, ## outcome model
  family="binomial",
  gbound=c(0,1),
  cvQinit=FALSE)
#-- get the ATE estimate:
tmle.fit$estimates$ATE$psi

```

[1] 0.1776457

## Task (11)8.

Without changing the weight truncation:

```

tmle.fit2 <- tmle(Y=sim.data$Y, A=sim.data$A,
  cbind(X1=sim.data$X1, X1.squared=sim.data$X1^2,X2=sim.data$X2,X3=sim.data$X3
),
  gform=A~X1+X2+X3, ## treatment model
  Qform=Y~A+X1.squared+X2+X3, ## outcome model
  family="binomial",
  cvQinit=FALSE)
#-- get the ATE estimate:
tmle.fit2$estimates$ATE$psi

```

[1] 0.08152813

Changing the gbound argument to avoid truncation of weights:



```

tmle.fit2 <- tmle(Y=sim.data$Y, A=sim.data$A,
  cbind(X1=sim.data$X1, X1.squared=sim.data$X1^2,X2=sim.data$X2,X3=sim.data$X3
),
  gform=A~X1+X2+X3, ## treatment model
  Qform=Y~A+X1.squared+X2+X3, ## outcome model
  family="binomial",
  gbound=c(0,1),
  cvQinit=FALSE)
#-- get the ATE estimate:
tmle.fit2$estimates$ATE$psi

```

[1] 0.08330816

For comparison, the estimating equation (EE) estimator is:

```

sim.data[, pred.EY1:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.EY0:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=0])]
sim.data[, mean(A/pred.pi1*(Y-pred.EY1) - (1-A)/(1-pred.pi1)*(Y-pred.EY0) + pred.EY1 -
  pred.EY0)]

```

[1] 0.08379974

## 5.1 Simulation study for changed data setting

Results are shown in Figure 4.

```

fit.g.glm1 <- list()
fit.g.glm2 <- list()
fit.g.rf <- list()
fit.ipw <- list()
fit.ipw2 <- list()
fit.ipw.rf <- list()
fit.tmle <- list()
fit.tmle.var <- list()
fit.wt.tmle <- list()
fit.wt.tmle.var <- list()
fit.ee <- list()
fit.ee.var <- list()
fit.tmle2 <- list()
fit.tmle2.var <- list()
fit.wt.tmle2 <- list()
fit.wt.tmle2.var <- list()
fit.ee2 <- list()
fit.ee2.var <- list()

for (m in 1:500) {

  set.seed(m+110)
  sim.data <- new.sim.fun(1000)

  fit.f <- glm(Y~A+X1+X2+X3, family=binomial, data=sim.data)
  fit.f2 <- glm(Y~A+X1.squared+X2+X3, family=binomial, data=sim.data[, X1.squared:=X1
    ^2])
  fit.rf <- rfsrc(Y~A+X1+X2+X3, data=sim.data)

```

```

fit.pi <- glm(A~X1+X2+X3, family=binomial, data=sim.data)
fit.pi2 <- glm(A~X2+X3, family=binomial, data=sim.data)
fit.rf.pi <- rfsrc(A~X1+X2+X3, data=sim.data)

##-- g-formula (misspecified);
sim.data[, pred.f1:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.f0:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=0])]
fit.g.glm1[[m]] <- sim.data[, mean(pred.f1 - pred.f0)]

##-- ipw (correctly specified);
sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
fit.ipw[[m]] <- sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred.pi1))]

##-- tmle (misspecified f);
tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
  cbind(X1=sim.data$X1,X2=sim.data$X2,X3=sim.data$X3),
  gform=A~X1+X2+X3, ## treatment model
  Qform=Y~A+X1+X2+X3, ## outcome model
  gbound=c(0,1),
  family="binomial",
  cvQinit=FALSE)
fit.tmle[[m]] <- tmle.fit$estimates$ATE$psi
fit.tmle.var[[m]] <- tmle.fit$estimates$ATE$var

##-- *with* weight truncation:
tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
  cbind(X1=sim.data$X1,X2=sim.data$X2,X3=sim.data$X3),
  gform=A~X1+X2+X3, ## treatment model
  Qform=Y~A+X1+X2+X3, ## outcome model
  family="binomial",
  cvQinit=FALSE)
fit.wt.tmle[[m]] <- tmle.fit$estimates$ATE$psi
fit.wt.tmle.var[[m]] <- tmle.fit$estimates$ATE$var

##-- estimating equation estimator (misspecified f);
sim.data[, pred.f1:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=1])]
sim.data[, pred.f0:=predict(fit.f, type="response", newdata=copy(sim.data)[, A:=0])]
sim.data[, pred.f:=predict(fit.f, type="response", newdata=sim.data)]
sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
fit.ee[[m]] <- sim.data[, mean((A/pred.pi1 - (1-A)/(1-pred.pi1))*(Y - pred.f) +
  pred.f1 - pred.f0)]
fit.ee.var[[m]] <- sim.data[, mean((A/pred.pi1*(Y-pred.f1) - (1-A)/(1-pred.pi1)*(Y-
  pred.f0) + pred.f1 - pred.f0)^2)/nrow(sim.data)]

##-- g-formula (correctly specified);
sim.data[, pred.f1:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=1])
]
sim.data[, pred.f0:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=0])
]
fit.g.glm2[[m]] <- sim.data[, mean(pred.f1 - pred.f0)]

##-- tmle (correctly specified f);
tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
  cbind(X1=sim.data$X1, X1.squared=sim.data$X1^2,X2=sim.data$X2,X3=sim.data$
X3),

```

```

      gform=A~X1+X2+X3, ## treatment model
      Qform=Y~A+X1.squared+X2+X3, ## outcome model
      gbound=c(0,1),
      family="binomial",
      cvQinit=FALSE)
fit.tmle2[[m]] <- tmle.fit$estimates$ATE$psi
fit.tmle2.var[[m]] <- tmle.fit$estimates$ATE$var

##-- *with* weight truncation:
tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
  cbind(X1=sim.data$X1, X1.squared=sim.data$X1^2,X2=sim.data$X2,X3=sim.data$
    X3),
  gform=A~X1+X2+X3, ## treatment model
  Qform=Y~A+X1.squared+X2+X3, ## outcome model
  family="binomial",
  cvQinit=FALSE)
fit.wt.tmle2[[m]] <- tmle.fit$estimates$ATE$psi
fit.wt.tmle2.var[[m]] <- tmle.fit$estimates$ATE$var

##-- estimating equation estimator (correctly specified f);
sim.data[, pred.f1:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=1])
]
sim.data[, pred.f0:=predict(fit.f2, type="response", newdata=copy(sim.data)[, A:=0])
]
sim.data[, pred.f:=predict(fit.f2, type="response", newdata=sim.data)]
sim.data[, pred.pi1:=predict(fit.pi, type="response", newdata=sim.data)]
fit.tee2[[m]] <- sim.data[, mean((A/pred.pi1 - (1-A)/(1-pred.pi1))*(Y - pred.f) +
  pred.f1 - pred.f0)]
fit.tee2.var[[m]] <- sim.data[, mean((A/pred.pi1*(Y-pred.f1) - (1-A)/(1-pred.pi1)*(Y-
  pred.f0) + pred.f1 - pred.f0)^2)/nrow(sim.data)]

##-- g-formula based on RF;
sim.data[, pred.f1:=predict(fit.rf, type="response", newdata=copy(sim.data)[, A:=1])
$predicted]
sim.data[, pred.f0:=predict(fit.rf, type="response", newdata=copy(sim.data)[, A:=0])
$predicted]
fit.g.rf[[m]] <- sim.data[, mean(pred.f1 - pred.f0)]

##-- ipw (misspecified);
sim.data[, pred.pi1:=predict(fit.pi2, type="response", newdata=sim.data)]
fit.ipw2[[m]] <- sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred.pi1))]

##-- rf (misspecified);
sim.data[, pred.pi1:=predict(fit.rf.pi, type="response", newdata=sim.data)$predicted
]
fit.ipw.rf[[m]] <- sim.data[, mean(A*Y/pred.pi1 - (1-A)*Y/(1-pred.pi1))]

saveRDS(list(fit.g.glm1 = fit.g.glm1,
  fit.g.glm2 = fit.g.glm2,
  fit.g.rf = fit.g.rf,
  fit.ipw = fit.ipw,
  fit.ipw2 = fit.ipw2,
  fit.ipw.rf = fit.ipw.rf,
  fit.tmle = fit.tmle,

```

```

fit.tmle.var = fit.tmle.var,
fit.wt.tmle = fit.wt.tmle,
fit.wt.tmle.var = fit.wt.tmle.var,
fit.ee = fit.ee,
fit.ee.var = fit.ee.var,
fit.tmle2 = fit.tmle2,
fit.tmle2.var = fit.tmle2.var,
fit.wt.tmle2 = fit.wt.tmle2,
fit.wt.tmle2.var = fit.wt.tmle2.var,
fit.ee2 = fit.ee2,
fit.ee2.var = fit.ee2.var),
file=paste0(here(), "/data/sim-data-output/",
            "save-est-sim-setting-2",
            ".rds"))
}

```

```

library(ggplot2)

estimator.list <- readRDS(file=paste0(here(), "/TMLE-course/data/sim-data-output/",
                                       "save-est-sim-setting-2",
                                       ".rds"))

pdat <- data.table(estimator=c(rep("g-formula estimator (misspecified)",
                                  length(estimator.list$fit.g.glm1)),
                              rep("g-formula estimator (correctly specified)",
                                  length(estimator.list$fit.g.glm2)),
                              rep("g-formula estimator (random forest)",
                                  length(estimator.list$fit.g.rf)),
                              rep("IPW estimator (correctly specified)",
                                  length(estimator.list$fit.ipw)),
                              rep("IPW estimator (misspecified)",
                                  length(estimator.list$fit.ipw2)),
                              rep("IPW estimator (random forest)",
                                  length(estimator.list$fit.ipw.rf)),
                              rep("TMLE estimator (misspecified initial)",
                                  length(estimator.list$fit.tmle)),
                              rep("EE estimator (misspecified initial)",
                                  length(estimator.list$fit.ee))),
                  est=c(unlist(estimator.list$fit.g.glm1),
                        unlist(estimator.list$fit.g.glm2),
                        unlist(estimator.list$fit.g.rf),
                        unlist(estimator.list$fit.ipw),
                        unlist(estimator.list$fit.ipw2),
                        unlist(estimator.list$fit.ipw.rf),
                        unlist(estimator.list$fit.tmle),
                        unlist(estimator.list$fit.ee)))

pdat[, estimator.factor := factor(estimator, levels=pdat[, unique(estimator)], order=
  TRUE)]

ggplot(pdat) +
  theme_bw(base_size=25) +
  geom_boxplot(aes(x=est)) +
  facet_wrap(. ~ estimator.factor, ncol=2) +

```

```
geom_vline(aes(xintercept=ATE), linetype="dashed", color="red") +  
xlab(expression(hat(psi)[n])) + ylab("")
```

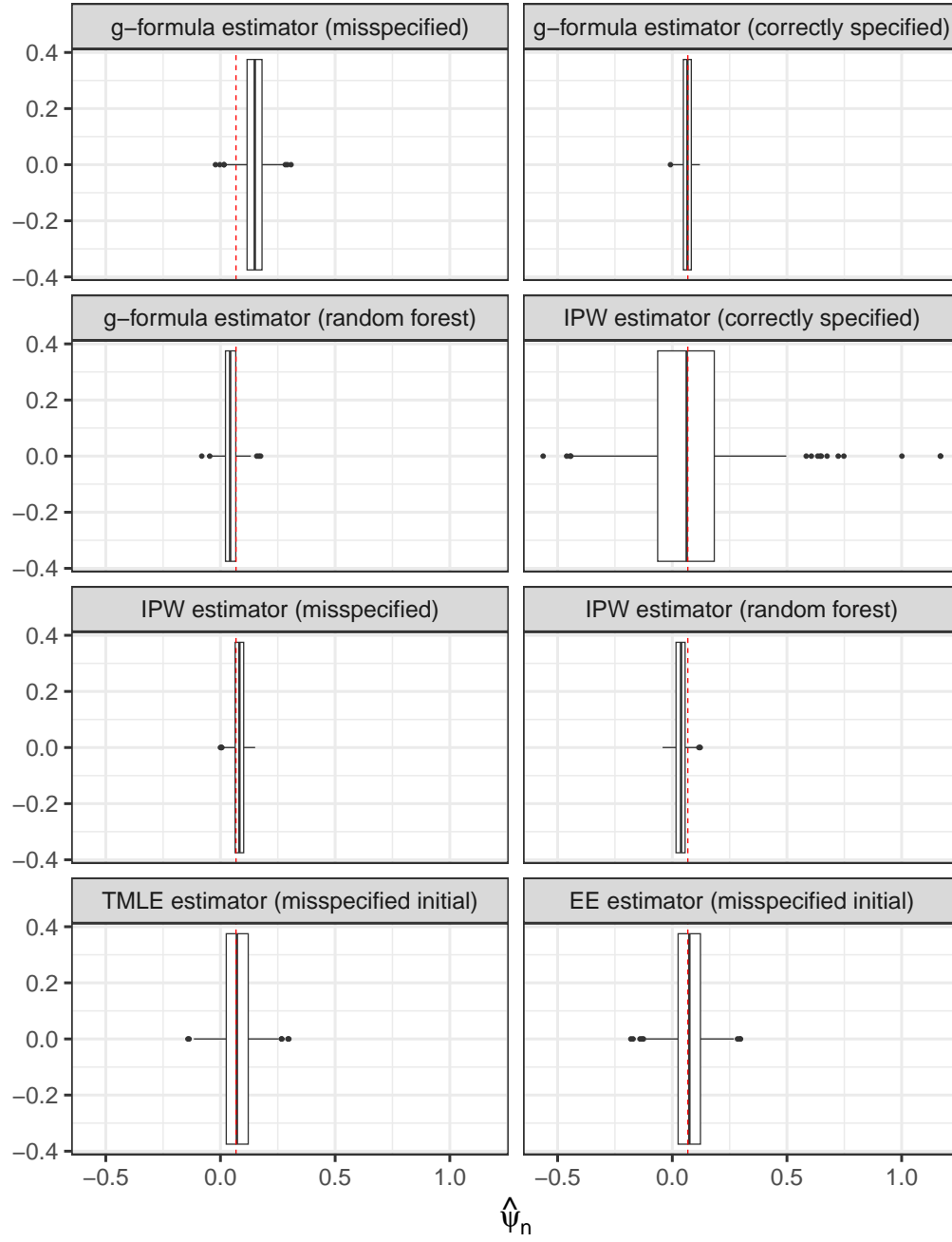


Figure 4